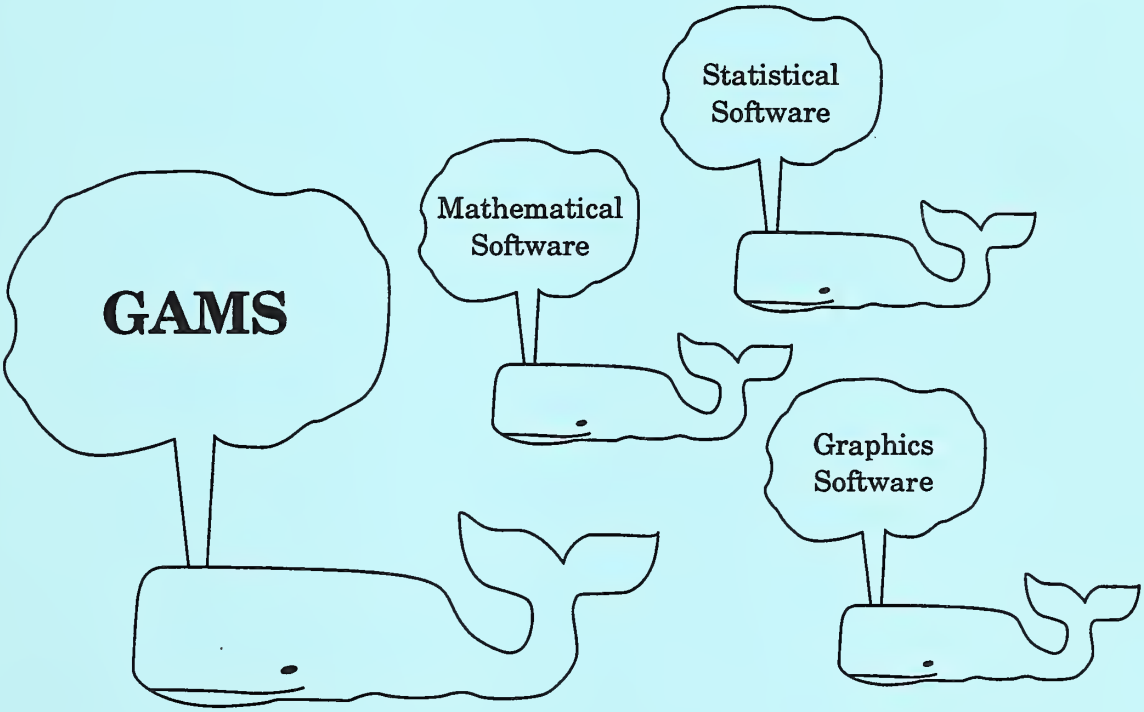
NAT'L INST. OF STAND & TECH R.I.C.
A11104 061676

NATIONAL INSTITUTE OF STANDARDS &
TECHNOLOGY
Research Information Center
Gaithersburg, MD 20899

NAT'L INST. OF STAND & TECH R.I.C.
A11103 389102
NIST
PUBLICATIONS

NISTIR 90-4237

# Guide to Available Mathematical Software




U.S. DEPARTMENT OF COMMERCE

National Institute of Standards and Technology
Center for Computing and Applied Mathematics
Gaithersburg, MD 20899

March 1990

QC
100
.U56
90-4237
1990
C.2

NATIONAL INSTITUTE OF STANDARDS & TECHNOLOGY
Research Information Center
Gaithersburg, MD 20899

DATE DUE

| | | | |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Demco, Inc. 38-293

NISTIR 90-4237

# Guide to Available Mathematical Software

Ronald F. Boisvert
Sally E. Howe
David K. Kahaner
Jeanne L. Springmann

Center for Computing and Applied Mathematics
National Institute of Standards and Technology
Gaithersburg, MD 20899

Issued March 1990

# Table of Contents

# Preface

The Guide to Available Mathematical Software (GAMS) is a catalog of general-purpose mathematical and statistical software in use at National Institute of Standards and Technology (NIST) laboratories in Gaithersburg, MD and Boulder, CO. This is the third edition of GAMS. In it, some 5000 software modules organized in 40 application packages available on six different computer systems at NIST are cataloged. This edition also incorporates a major revision of the GAMS Classification Scheme for mathematical and statistical software originally presented in [BHK85].

The document itself was developed and produced using computer hardware and software tools. Most pages are the output of computer programs which query a database of information about locally-available mathematical and statistical software developed for this purpose [BHS89,Boi89]. The document itself was typeset using LaTeX [Lam86].

This document could not have been produced without the cooperation of many. In particular we wish to thank Paul Boggs, David E. Brown, John Brown, Joyce Conlon, Lina DeLeonibus, James Filliben, John Koontz, Daniel Lozier, Marjorie McClain, and the staff of the NIST Statistical Engineering Division. We also appreciate the continued support for research in mathematical and statistical software provided by NIST and its Center for Computing and Applied Mathematics.

# References

[BHK85] R. F. Boisvert, S. E. Howe, and D. K. Kahaner, GAMS — a framework for the management of scientific software, *ACM Transactions on Mathematical Software*, 11:313–355, 1985.

[BHS89] R. F. Boisvert, S. E. Howe, and J. L. Springmann, *Internal Structure of the Guide to Available Mathematical Software*, NISTIR 89-4042, National Institute of Standards and Technology, (also available as PB89-170864/AS, National Technical Information Service, Springfield, VA 22161), March 1989.

[Boi89] R. F. Boisvert, The Guide to Available Mathematical Software advisory system, *Mathematics and Computers in Simulation*, 31:453–463, 1989.

[Lam86] L. Lamport, LaTeX *User's Guide & Reference Manual*, Addison-Wesley, Reading, MA, 1986.

# Disclaimer

Certain commercial products are identified in this report in order to adequately document the computing environment at NIST. Identification of these products does not imply recommendation or endorsement by NIST, nor does it imply that the identified products are necessarily the best available for the purpose.

# How To Use GAMS

This document is organized so that one may easily locate software that solves a given mathematical or statistical problem. Given such a problem, one should

1. Turn to *GAMS Problem Classes* to find the classification which most clearly identifies the problem. A bullet (•) to the left of the classification indicates that there is software available in that class.
2. Turn to *Modules By Class* to find the list of software for the problem class located above. For each software module there is a short description and the name of the application package which contains it.
3. (Optional) Further information about each software module may be found in the *Module Dictionary* which is organized in alphabetical order by module name. Included is a list of computers on which the module is available and notes pertaining to the implementation, such as its precision.
4. Turn to *Application Package Reference* to find information about how to access the software packages on each computer on which the software is available. Included is a reference for the package and a method for obtaining detailed on-line documentation on each computer.

At the beginning of each major section of GAMS you will find further details on the organization and terminology used in that section.

The following tables contain information about the computer systems on which the software is available.

**Computer Systems with GAMS Software**

| *Name* | *Computer* | *Location* |
|---|---|---|
| 855NOS | Cyber 180/855 (NOS) | Gaithersburg Central Facility |
| 855VE | Cyber 180/855 (NOS/VE) | Gaithersburg Central Facility |
| 205 | Cyber 200/205 (VSOS) | Gaithersburg Central Facility |
| 840NOS | Cyber 180/840 (NOS) | Boulder Central Facility |
| CAMVAX | VAX 11/785 (VMS) | Gaithersburg CCAM† |
| PC | PCs/clones in various configurations | Gaithersburg Micro-Resource Site |

† NIST Center for Computing and Applied Mathematics. Access restricted.

| **New Accounts** | | | |
|---|---|---|---|
| In Gaithersburg | Betty Cantrell | 301-975-2883 (FTS-879-2883) | A221 Admin |
| CAMVAX | Darcy Barnett | 301-975-3830 (FTS-879-3830) | B146 Tech |
| In Boulder | Karen Ohm | 303-497-5850 (FTS-320-5850) | 4549 – 1 |
| **User assistance** | | | |
| In Gaithersburg | Consultants | 301-975-2968 (FTS-879-2968) | B 08 Tech |
| CAMVAX | Darcy Barnett | 301-975-3830 (FTS-879-3830) | B146 Tech |
| PC | Marjorie McClain | 301-975-3837 (FTS-879-3837) | A151 Tech |
| In Boulder | Consultants | 303-497-3566 (FTS-320-3566) | 3550 – 1 |

# GAMS Interactive Consultant

The GAMS Interactive Consultant is an on-line version of this catalog. Using the Consultant one may browse through the problem classification scheme, viewing short descriptions of software modules available at selected problem classes. This interactive system is available on the Cyber 180/855 (NOS and NOS/VE), as well as on the CCAM Vax 11/785. To run it, type the following:

| | |
|---|---|
| On the Cyber 180/855 (NOS): | `INVOKE,GAMS` |
| On the Cyber 180/855 (NOS/VE): | `GAMS` |
| On the VAX 11/785 (VMS): | `GAMS` |

# Problem Classifications

*To find software for a given problem one must first describe the problem precisely. For that purpose this section contains a taxonomy for mathematical and statistical problems. The taxonomy, or problem classification scheme, is divided into 20 major problem classes. Many of these are also divided into subclasses which may then be subdivided further. The classification scheme is presented in the form of an indented outline on the following pages.*

*In computer science terminology, the scheme is a tree, with "Mathematical and Statistical Software" being the root. The higher levels of the tree correspond to general problem areas and the lower levels correspond to more specific problems. The tree has a maximum depth of seven levels. Tree levels correspond to indentation levels in the listing that follows.*

*Each problem class has a label. Problem classes at the highest level of the tree are labeled with single letters, such as J (Integral Transforms). Subclasses are identified by alternating letter-and-number combinations; thus, for example, the subclasses of J1 (Trigonometric transforms, including fast Fourier transforms) are labeled J1a, J1b, etc. The longer the label, the lower the level of the tree, and the more detailed the problem specification.*

*To find software for a given problem one must first find the problem class which most closely describes the problem. Problem classes which contain software cataloged in GAMS have a bullet (•) to the left of their labels. Note also that the problem classification scheme has comments embedded within to suggest related problem classes which might also have appropriate software.*

*In general, try to select the most specific problem class (lowest level, longest label) that is appropriate. Some software modules solve a large number of problems, and are classified at less specific problem classes (higher levels, shorter labels); users should also look for such classes.*

## A. Arithmetic, error analysis

- A1. Integer
- A2. Rational
- A3. Real
- • A3a. Standard precision
- • A3c. Extended precision
- • A3d. Extended range
- • A4. Complex
- • A4a. Standard precision
- A4c. Extended precision
- A4d. Extended range
- A5. Interval
- • A6. Change of representation
- • A6a. Type conversion
- • A6b. Base conversion
- • A6c. Decomposition, construction
- • A7. Sequences (e.g., convergence acceleration)

## • B. Number theory

## C. Elementary and special functions *(search also class L5)*

- • C1. Integer-valued functions (e.g., factorial, binomial coefficient, permutations, combinations, floor, ceiling)
- • C2. Powers, roots, reciprocals
- C3. Polynomials
- • C3a. Orthogonal
- • C3a1. Trigonometric
- • C3a2. Chebyshev, Legendre
- C3a3. Laguerre
- C3a4. Hermite
- • C3b. Non-orthogonal
- • C4. Elementary transcendental functions
- • C4a. Trigonometric, inverse trigonometric
- • C4b. Exponential, logarithmic
- • C4c. Hyperbolic, inverse hyperbolic
- C4d. Integrals of elementary transcendental functions
- • C5. Exponential and logarithmic integrals
- • C6. Cosine and sine integrals
- C7. Gamma
- • C7a. Gamma, log gamma, reciprocal gamma
- • C7b. Beta, log beta
- • C7c. Psi function
- C7d. Polygamma function
- • C7e. Incomplete gamma
- • C7f. Incomplete beta
- C7g. Riemann zeta
- C8. Error functions
- • C8a. Error functions, their inverses, integrals, including the normal distribution function
- • C8b. Fresnel integrals

- • **C8c.** Dawson's integral
- • **C9.** Legendre functions
- **C10.** Bessel functions
- • **C10a.** $J$, $Y$, $H_1$, $H_2$
- • **C10a1.** Real argument, integer order
- • **C10a2.** Complex argument, integer order
- • **C10a3.** Real argument, real order
- • **C10a4.** Complex argument, real order
- **C10a5.** Complex argument, complex order
- • **C10b.** $I$, $K$
- • **C10b1.** Real argument, integer order
- • **C10b2.** Complex argument, integer order
- • **C10b3.** Real argument, real order
- • **C10b4.** Complex argument, real order
- **C10b5.** Complex argument, complex order
- • **C10c.** Kelvin functions
- • **C10d.** Airy and Scorer functions
- • **C10e.** Struve, Anger, and Weber functions
- • **C10f.** Integrals of Bessel functions
- • **C11.** Confluent hypergeometric functions
- • **C12.** Coulomb wave functions
- **C13.** Jacobian elliptic functions, theta functions
- • **C14.** Elliptic integrals
- • **C15.** Weierstrass elliptic functions
- **C16.** Parabolic cylinder functions
- • **C17.** Mathieu functions
- **C18.** Spheroidal wave functions
- • **C19.** Other special functions

## • D. Linear Algebra

- • **D1.** Elementary vector and matrix operations
- • **D1a.** Elementary vector operations
- • **D1a1.** Set to constant
- • **D1a2.** Minimum and maximum components
- **D1a3.** Norm
- • **D1a3a.** $L_1$ (sum of magnitudes)
- • **D1a3b.** $L_2$ (Euclidean norm)
- • **D1a3c.** $L_\infty$ (maximum magnitude)
- • **D1a4.** Dot product (inner product)
- • **D1a5.** Copy or exchange (swap)
- • **D1a6.** Multiplication by scalar
- • **D1a7.** Triad ($\alpha x + y$ for vectors $x$, $y$ and scalar $\alpha$)
- • **D1a8.** Elementary rotation (Givens transformation)
- • **D1a9.** Elementary reflection (Householder transformation)
- • **D1a10.** Convolutions
- • **D1a11.** Other vector operations
- • **D1b.** Elementary matrix operations
- • **D1b1.** Initialize (e.g., to zero or identity)
- • **D1b2.** Norm
- • **D1b3.** Transpose
- • **D1b4.** Multiplication by vector

- • D1b5. Addition, subtraction
- • D1b6. Multiplication
- • D1b7. Matrix polynomial
- • D1b8. Copy
- • D1b9. Storage mode conversion
- • D1b10. Elementary rotation (Givens transformation)
- • D1b11. Elementary reflection (Householder transformation)
- D2. Solution of systems of linear equations (including inversion, *LU* and related decompositions)
- • D2a. Real nonsymmetric matrices
- • D2a1. General
- • D2a2. Banded
- • D2a2a. Tridiagonal
- • D2a3. Triangular
- • D2a4. Sparse
- D2b. Real symmetric matrices
- D2b1. General
- • D2b1a. Indefinite
- • D2b1b. Positive definite
- • D2b2. Positive definite banded
- • D2b2a. Tridiagonal
- • D2b4. Sparse
- D2c. Complex non-Hermitian matrices
- • D2c1. General
- • D2c2. Banded
- • D2c2a. Tridiagonal
- • D2c3. Triangular
- D2c4. Sparse
- D2d. Complex Hermitian matrices
- D2d1. General
- • D2d1a. Indefinite
- • D2d1b. Positive definite
- • D2d2. Positive definite banded
- • D2d2a. Tridiagonal
- D2d4. Sparse
- • D2e. Associated operations (e.g., matrix reorderings)
- D3. Determinants
- D3a. Real nonsymmetric matrices
- • D3a1. General
- • D3a2. Banded
- D3a2a. Tridiagonal
- • D3a3. Triangular
- D3a4. Sparse
- D3b. Real symmetric matrices
- D3b1. General
- • D3b1a. Indefinite
- • D3b1b. Positive definite
- • D3b2. Positive definite banded
- D3b2a. Tridiagonal
- D3b4. Sparse
- D3c. Complex non-Hermitian matrices
- • D3c1. General
- • D3c2. Banded

| | | |
|---|---|---|
| | **D3c2a.** | Tridiagonal |
| • | **D3c3.** | Triangular |
| | **D3c4.** | Sparse |
| | **D3d.** | Complex Hermitian matrices |
| | **D3d1.** | General |
| • | **D3d1a.** | Indefinite |
| • | **D3d1b.** | Positive definite |
| • | **D3d2.** | Positive definite banded |
| | **D3d2a.** | Tridiagonal |
| | **D3d4.** | Sparse |
| | **D4.** | Eigenvalues, eigenvectors |
| | **D4a.** | Ordinary eigenvalue problems ($Ax = \lambda x$) |
| • | **D4a1.** | Real symmetric |
| • | **D4a2.** | Real nonsymmetric |
| • | **D4a3.** | Complex Hermitian |
| • | **D4a4.** | Complex non-Hermitian |
| • | **D4a5.** | Tridiagonal |
| • | **D4a6.** | Banded |
| • | **D4a7.** | Sparse |
| | **D4b.** | Generalized eigenvalue problems (e.g., $Ax = \lambda Bx$) |
| • | **D4b1.** | Real symmetric |
| • | **D4b2.** | Real general |
| | **D4b3.** | Complex Hermitian |
| • | **D4b4.** | Complex general |
| • | **D4b5.** | Banded |
| • | **D4c.** | Associated operations |
| | **D4c1.** | Transform problem |
| • | **D4c1a.** | Balance matrix |
| | **D4c1b.** | Reduce to compact form |
| • | **D4c1b1.** | Tridiagonal |
| • | **D4c1b2.** | Hessenberg |
| • | **D4c1b3.** | Other |
| • | **D4c1c.** | Standardize problem |
| | **D4c2.** | Compute eigenvalues of matrix in compact form |
| • | **D4c2a.** | Tridiagonal |
| • | **D4c2b.** | Hessenberg |
| • | **D4c2c.** | Other |
| • | **D4c3.** | Form eigenvectors from eigenvalues |
| • | **D4c4.** | Back transform eigenvectors |
| • | **D4c5.** | Determine Jordan normal form |
| • | **D5.** | $QR$ decomposition, Gram-Schmidt orthogonalization |
| • | **D6.** | Singular value decomposition |
| | **D7.** | Update matrix decompositions |
| | **D7a.** | $LU$ |
| • | **D7b.** | Cholesky |
| • | **D7c.** | $QR$ |
| | **D7d.** | Singular value |
| • | **D8.** | Other matrix equations (e.g., $AX + XB = C$) |
| | **D9.** | Singular, overdetermined or underdetermined systems of linear equations, generalized inverses |
| | **D9a.** | Unconstrained |
| • | **D9a1.** | Least squares ($L_2$) solution |
| • | **D9a2.** | Chebyshev ($L_\infty$) solution |

• contains software

- **D9a3.** Least absolute value ($L_1$) solution
- **D9a4.** Other

**D9b.** Constrained
- **D9b1.** Least squares ($L_2$) solution
- **D9b2.** Chebyshev ($L_\infty$) solution
- **D9b3.** Least absolute value ($L_1$)

**D9b4.** Other
- **D9c.** Generalized inverses

## E. Interpolation

**E1.** Univariate data (curve fitting)
- **E1a.** Polynomial splines (piecewise polynomials)
- **E1b.** Polynomials
- **E1c.** Other functions (e.g., rational, trigonometric)

**E2.** Multivariate data (surface fitting)
- **E2a.** Gridded
- **E2b.** Scattered

**E3.** Service routines for interpolation

**E3a.** Evaluation of fitted functions, including quadrature
- **E3a1.** Function evaluation
- **E3a2.** Derivative evaluation
- **E3a3.** Quadrature
- **E3b.** Grid or knot generation
- **E3c.** Manipulation of basis functions (e.g., evaluation, change of basis)
- **E3d.** Other

## F. Solution of nonlinear equations

**F1.** Single equation

**F1a.** Polynomial
- **F1a1.** Real coefficients
- **F1a2.** Complex coefficients
- **F1b.** Nonpolynomial
- **F2.** System of equations
- **F3.** Service routines (e.g., check user-supplied derivatives)

## G. Optimization *(search also classes K, L8)*

**G1.** Unconstrained

**G1a.** Univariate

**G1a1.** Smooth function
- **G1a1a.** User provides no derivatives
- **G1a1b.** User provides first derivatives

**G1a1c.** User provides first and second derivatives
- **G1a2.** General function (no smoothness assumed)

**G1b.** Multivariate

**G1b1.** Smooth function
- **G1b1a.** User provides no derivatives
- **G1b1b.** User provides first derivatives

- • **G1b1c.** User provides first and second derivatives
- • **G1b2.** General function (no smoothness assumed)
- **G2.** Constrained
- **G2a.** Linear programming
- • **G2a1.** Dense matrix of constraints
- • **G2a2.** Sparse matrix of constraints
- • **G2b.** Transportation and assignments problem
- **G2c.** Integer programming
- **G2c1.** Zero/one
- **G2c2.** Covering and packing problems
- • **G2c3.** Knapsack problems
- **G2c4.** Matching problems
- • **G2c5.** Routing, scheduling, location problems
- • **G2c6.** Pure integer programming
- **G2c7.** Mixed integer programming
- **G2d.** Network (for network reliability search class M)
- • **G2d1.** Shortest path
- • **G2d2.** Minimum spanning tree
- **G2d3.** Maximum flow
- **G2d3a.** Generalized networks
- **G2d3b.** Networks with side constraints
- **G2d4.** Test problem generation
- **G2e.** Quadratic programming
- • **G2e1.** Positive definite Hessian (i.e., convex problem)
- • **G2e2.** Indefinite Hessian
- **G2f.** Geometric programming
- **G2g.** Dynamic programming
- **G2h.** General nonlinear programming
- **G2h1.** Simple bounds
- **G2h1a.** Smooth function
- • **G2h1a1.** User provides no derivatives
- • **G2h1a2.** User provides first derivatives
- • **G2h1a3.** User provides first and second derivatives
- • **G2h1b.** General function (no smoothness assumed)
- **G2h2.** Linear equality or inequality constraints
- **G2h2a.** Smooth function
- • **G2h2a1.** User provides no derivatives
- • **G2h2a2.** User provides first derivatives
- **G2h2a3.** User provides first and second derivatives
- **G2h2b.** General function (no smoothness assumed)
- **G2h3.** Nonlinear constraints
- **G2h3a.** Equality constraints only
- • **G2h3a1.** Smooth function and constraints
- **G2h3a1a.** User provides no derivatives
- **G2h3a1b.** User provides first derivatives of function and constraints
- **G2h3a1c.** User provides first and second derivatives of function and constraints
- **G2h3a2.** General function and constraints (no smoothness assumed)
- **G2h3b.** Equality and inequality constraints
- **G2h3b1.** Smooth function and constraints
- • **G2h3b1a.** User provides no derivatives
- • **G2h3b1b.** User provides first derivatives of function and constraints
- **G2h3b1c.** User provides first and second derivatives of function and constraints

• contains software

G2h3b2. General function and constraints (no smoothness assumed)
• G2i. Global solution to nonconvex problems
• G3. Optimal control
G4. Service routines
G4a. Problem input (e.g., matrix generation)
G4b. Problem scaling
• G4c. Check user-supplied derivatives
• G4d. Find feasible point
G4e. Check for redundancy
• G4f. Other

## H. Differentiation, integration

• H1. Numerical differentiation
H2. Quadrature (numerical evaluation of definite integrals)
H2a. One-dimensional integrals
H2a1. Finite interval (general integrand)
H2a1a. Integrand available via user-defined procedure
• H2a1a1. Automatic (user need only specify required accuracy)
• H2a1a2. Nonautomatic
H2a1b. Integrand available only on grid
H2a1b1. Automatic (user need only specify required accuracy)
• H2a1b2. Nonautomatic
H2a2. Finite interval (specific or special type integrand including weight functions, oscillating and singular integrands, principal value integrals, splines, etc.)
H2a2a. Integrand available via user-defined procedure
• H2a2a1. Automatic (user need only specify required accuracy)
• H2a2a2. Nonautomatic
H2a2b. Integrand available only on grid
• H2a2b1. Automatic (user need only specify required accuracy)
H2a2b2. Nonautomatic
H2a3. Semi-infinite interval (including exp $-x$ weight function)
H2a3a. Integrand available via user-defined procedure
• H2a3a1. Automatic (user need only specify required accuracy)
• H2a3a2. Nonautomatic
H2a4. Infinite interval (including exp $-x^2$ weight function)
H2a4a. Integrand available via user-defined procedure
• H2a4a1. Automatic (user need only specify required accuracy)
• H2a4a2. Nonautomatic
H2b. Multidimensional integrals
H2b1. One or more hyper-rectangular regions (includes iterated integrals)
H2b1a. Integrand available via user-defined procedure
• H2b1a1. Automatic (user need only specify required accuracy)
• H2b1a2. Nonautomatic
H2b1b. Integrand available only on grid
H2b1b1. Automatic (user need only specify required accuracy)
• H2b1b2. Nonautomatic
H2b2. n-dimensional quadrature on a nonrectangular region
H2b2a. Integrand available via user-defined procedure
• H2b2a1. Automatic (user need only specify required accuracy)
• H2b2a2. Nonautomatic
H2b2b. Integrand available only on grid

| | | |
|---|---|---|
| | **H2b2b1.** | Automatic (user need only specify required accuracy) |
| | **H2b2b2.** | Nonautomatic |
| • | **H2c.** | Service routines (e.g., compute weights and nodes for quadrature formulas) |

## I. Differential and integral equations

| | | |
|---|---|---|
| | **I1.** | Ordinary differential equations (ODE's) |
| • | **I1a.** | Initial value problems |
| | **I1a1.** | General, nonstiff or mildly stiff |
| • | **I1a1a.** | One-step methods (e.g., Runge-Kutta) |
| • | **I1a1b.** | Multistep methods (e.g., Adams predictor-corrector) |
| • | **I1a1c.** | Extrapolation methods (e.g., Bulirsch-Stoer) |
| • | **I1a2.** | Stiff and mixed algebraic- differential equations |
| | **I1b.** | Multipoint boundary value problems |
| • | **I1b1.** | Linear |
| • | **I1b2.** | Nonlinear |
| • | **I1b3.** | Eigenvalue (e.g., Sturm-Liouville) |
| • | **I1c.** | Service routines (e.g., interpolation of solutions, error handling, test programs) |
| | **I2.** | Partial differential equations |
| | **I2a.** | Initial boundary value problems |
| • | **I2a1.** | Parabolic |
| • | **I2a1a.** | One spatial dimension |
| • | **I2a1b.** | Two or more spatial dimensions |
| • | **I2a2.** | Hyperbolic |
| | **I2b.** | Elliptic boundary value problems |
| | **I2b1.** | Linear |
| • | **I2b1a.** | Second order |
| | **I2b1a1.** | Poisson (Laplace) or Helmholtz equation |
| • | **I2b1a1a.** | Rectangular domain (or topologically rectangular in the coordinate system) |
| • | **I2b1a1b.** | Nonrectangular domain |
| • | **I2b1a2.** | Other separable problems |
| • | **I2b1a3.** | Nonseparable problems |
| | **I2b1c.** | Higher order equations (e.g., biharmonic) |
| | **I2b2.** | Nonlinear |
| | **I2b3.** | Eigenvalue |
| • | **I2b4.** | Service routines |
| • | **I2b4a.** | Domain triangulation *(search also class P)* |
| • | **I2b4b.** | Solution of discretized elliptic equations |
| • | **I3.** | Integral equations |

## • J. Integral transforms

| | | |
|---|---|---|
| | **J1.** | Trigonometric transforms including fast Fourier transforms |
| | **J1a.** | One-dimensional |
| • | **J1a1.** | Real |
| • | **J1a2.** | Complex |
| • | **J1a3.** | Sine and cosine transforms |
| • | **J1b.** | Multidimensional |
| • | **J2.** | Convolutions |
| • | **J3.** | Laplace transforms |
| • | **J4.** | Hilbert transforms |

• contains software

## K. Approximation *(search also class L8)*

K1. Least squares ($L_2$) approximation
K1a. Linear least squares *(search also classes D5, D6, D9)*
K1a1. Unconstrained
K1a1a. Univariate data (curve fitting)
• K1a1a1. Polynomial splines (piecewise polynomials)
• K1a1a2. Polynomials
• K1a1a3. Other functions (e.g., trigonometric, user-specified)
• K1a1b. Multivariate data (surface fitting)
K1a2. Constrained
• K1a2a. Linear constraints
K1a2b. Nonlinear constraints
K1b. Nonlinear least squares
K1b1. Unconstrained
K1b1a. Smooth functions
• K1b1a1. User provides no derivatives
• K1b1a2. User provides first derivatives
• K1b1a3. User provides first and second derivatives
K1b1b. General functions
K1b2. Constrained
• K1b2a. Linear constraints
K1b2b. Nonlinear constraints
• K2. Minimax ($L_\infty$) approximation
• K3. Least absolute value ($L_1$) approximation
• K4. Other analytic approximations (e.g., Taylor polynomial, Pade)
• K5. Smoothing
K6. Service routines for approximation
K6a. Evaluation of fitted functions, including quadrature
• K6a1. Function evaluation
• K6a2. Derivative evaluation
• K6a3. Quadrature
• K6b. Grid or knot generation
• K6c. Manipulation of basis functions (e.g., evaluation, change of basis)
• K6d. Other

## L. Statistics, probability

L1. Data summarization
L1a. One-dimensional data
• L1a1. Raw data
• L1a1a. Location
• L1a1b. Dispersion
• L1a1c. Shape
• L1a1d. Frequency, cumulative frequency
• L1a1e. Ties
• L1a3. Grouped data
• L1b. Two dimensional data *(search also class L1c)*
L1c. Multi-dimensional data
• L1c1. Raw data

- • **L1c1b.** Covariance, correlation
- • **L1c1d.** Frequency, cumulative frequency
- • **L1c2.** Raw data containing missing values *(search also class L1c1)*
- • **L2.** Data manipulation
- • **L2a.** Transform *(search also classes L10a1, N6, and N8)*
- • **L2b.** Tally
- • **L2c.** Subset
- • **L2d.** Merge *(search also class N7)*
- • **L2e.** Construct new variables (e.g., indicator variables)
- **L3.** Elementary statistical graphics *(search also class Q)*
- **L3a.** One-dimensional data
- • **L3a1.** Histograms
- • **L3a2.** Frequency, cumulative frequency, percentile plots
- • **L3a3.** EDA (e.g., box-plots)
- • **L3a4.** Bar charts
- • **L3a5.** Pie charts
- • **L3a6.** $X_i$ vs. $i$ (including symbol plots)
- • **L3a7.** Lag plots (e.g., plots of $X_i$ vs. $X_{i-1}$)
- **L3b.** Two-dimensional data *(search also class L3e)*
- • **L3b1.** Histograms (superimposed and bivariate)
- • **L3b2.** Frequency, cumulative frequency
- **L3b3.** Scatter diagrams
- • **L3b3a.** Y vs. X
- • **L3b3b.** Symbol plots
- • **L3b3c.** Lag plots (i.e., plots of $X_i$ vs. $Y_{i-j}$)
- • **L3b4.** EDA
- • **L3c.** Three-dimensional data *(search also class L3e)*
- **L3e.** Multi-dimensional data
- • **L3e1.** Histograms
- • **L3e2.** Frequency, cumulative frequency, percentile plots
- **L3e3.** Scatter diagrams
- • **L3e3a.** Superimposed $Y$ vs. $X$
- • **L3e3c.** Superimposed $X_i$ vs. $i$
- • **L3e3d.** Matrices of bivariate scatter diagrams
- • **L3e4.** EDA
- **L4.** Elementary data analysis
- **L4a.** One-dimensional data
- **L4a1.** Raw data
- **L4a1a.** Parametric analysis
- • **L4a1a1.** Plots of empirical and theoretical density and distribution functions
- **L4a1a2.** Probability plots
- • **L4a1a2b.** Beta, binomial
- • **L4a1a2c.** Cauchy, chi-squared
- • **L4a1a2d.** Double exponential
- • **L4a1a2e.** Exponential, extreme value
- • **L4a1a2f.** F distribution
- • **L4a1a2g.** Gamma, geometric
- • **L4a1a2h.** Halfnormal
- • **L4a1a2l.** Lambda, logistic, lognormal
- • **L4a1a2n.** Negative binomial, normal
- • **L4a1a2p.** Pareto, Poisson
- • **L4a1a2s.** Semicircular

• contains software

- **L4a1a2t.** t distribution, triangular
- **L4a1a2u.** Uniform
- **L4a1a2w.** Weibull

**L4a1a3.** Probability plot correlation coefficient plots
- **L4a1a3c.** Chi-squared
- **L4a1a3e.** Extreme value
- **L4a1a3g.** Gamma, geometric
- **L4a1a3l.** Lambda
- **L4a1a3n.** Normal
- **L4a1a3p.** Pareto, Poisson
- **L4a1a3t.** t distribution
- **L4a1a3w.** Weibull

**L4a1a4.** Parameter estimates and tests
- **L4a1a4b.** Binomial
- **L4a1a4e.** Extreme value
- **L4a1a4n.** Normal
- **L4a1a4p.** Poisson
- **L4a1a4u.** Uniform
- **L4a1a4w.** Weibull
- **L4a1a5.** Transformation selection (e.g., for normality)
- **L4a1a6.** Tail and outlier analysis
- **L4a1a7.** Tolerance limits

**L4a1b.** Nonparametric analysis
- **L4a1b1.** Estimates and tests regarding location (e.g., median), dispersion, and shape
- **L4a1b2.** Density function estimation
- **L4a1c.** Goodness-of-fit tests
- **L4a1d.** Analysis of a sequence of numbers *(search also class L10a)*
- **L4a3.** Grouped and/or censored data
- **L4a4.** Data sampled from a finite population
- **L4a5.** Categorical data

**L4b.** Two dimensional data *(search also class L4c)*

**L4b1.** Pairwise independent data

**L4b1a.** Parametric analysis
- **L4b1a1.** Plots of empirical and theoretical density and distribution functions
- **L4b1a4.** Parameter estimates and hypothesis tests
- **L4b1b.** Nonparametric analysis (e.g., rank tests)
- **L4b1c.** Goodness-of-fit tests
- **L4b3.** Pairwise dependent data
- **L4b4.** Pairwise dependent grouped data
- **L4b5.** Data sampled from a finite population

**L4c.** Multi-dimensional data *(search also classes L4b and L7a1)*

**L4c1.** Independent data
- **L4c1a.** Parametric analysis
- **L4c1b.** Nonparametric analysis
- **L4e.** Multiple multi-dimensional data sets

**L5.** Function evaluation *(search also class C)*

**L5a.** Univariate

**L5a1.** Cumulative distribution functions, probability density functions
- **L5a1b.** Beta, binomial
- **L5a1c.** Cauchy, chi-squared
- **L5a1d.** Double exponential
- **L5a1e.** Error function, exponential, extreme value

- **L5a1f.** F distribution
- **L5a1g.** Gamma, general, geometric
- **L5a1h.** Halfnormal, hypergeometric
- **L5a1k.** Kendall F statistic, Kolmogorov-Smirnov
- **L5a1l.** Lambda, logistic, lognormal
- **L5a1n.** Negative binomial, normal
- **L5a1p.** Pareto, Poisson
- **L5a1t.** t distribution
- **L5a1u.** Uniform
- **L5a1v.** Von Mises
- **L5a1w.** Weibull

**L5a2.** Inverse distribution functions, sparsity functions

- **L5a2b.** Beta, binomial
- **L5a2c.** Cauchy, chi-squared
- **L5a2d.** Double exponential
- **L5a2e.** Error function, exponential, extreme value
- **L5a2f.** F distribution
- **L5a2g.** Gamma, general, geometric
- **L5a2h.** Halfnormal
- **L5a2l.** Lambda, logistic, lognormal
- **L5a2n.** Negative binomial, normal, normal order statistics
- **L5a2p.** Pareto, Poisson
- **L5a2t.** t distribution
- **L5a2u.** Uniform
- **L5a2w.** Weibull

**L5b.** Multivariate

**L5b1.** Cumulative multivariate distribution functions, probability density functions

- **L5b1n.** Normal

**L5b2.** Inverse cumulative distribution functions

- **L5b2n.** Normal

**L6.** Random number generation

**L6a.** Univariate

- **L6a2.** Beta, binomial, Boolean
- **L6a3.** Cauchy, chi-squared
- **L6a4.** Double exponential
- **L6a5.** Exponential, extreme value
- **L6a6.** F distribution
- **L6a7.** Gamma, general (continuous, discrete), geometric
- **L6a8.** Halfnormal, hypergeometric
- **L6a12.** Lambda, logistic, lognormal
- **L6a14.** Negative binomial, normal, normal order statistics
- **L6a16.** Pareto, Pascal, permutations, Poisson
- **L6a19.** Samples, stable distribution
- **L6a20.** t distribution, time series, triangular
- **L6a21.** Uniform (continuous, discrete), uniform order statistics
- **L6a22.** Von Mises
- **L6a23.** Weibull

**L6b.** Multivariate

- **L6b3.** Contingency table, correlation matrix
- **L6b5.** Experimental designs
- **L6b12.** Linear $L_1$ (least absolute value) approximation
- **L6b13.** Multinomial

• contains software

- **L6b14.** Normal
- **L6b15.** Orthogonal matrix
- **L6b21.** Uniform
- **L6c.** Service routines (e.g., seed)

**L7.** Analysis of variance (including analysis of covariance)
- **L7a.** One-way
- **L7a1.** Parametric
- **L7a2.** Nonparametric
- **L7b.** Two-way *(search also class L7d)*
- **L7c.** Three-way (e.g., Latin squares) *(search also class L7d)*
- **L7d.** Multi-way
- **L7d1.** Balanced complete data (e.g., factorial designs)
- **L7d2.** Balanced incomplete data
- **L7d3.** General linear models (unbalanced data)
- **L7e.** Multivariate
- **L7f.** Generate experimental designs
- **L7g.** Service routines

**L8.** Regression *(search also classes D5, D6, D9, G, K)*

**L8a.** Simple linear (i.e., $y = b_0 + b_1 x$) *(search also class L8h)*

**L8a1.** Ordinary least squares
- **L8a1a.** Parameter estimation
- **L8a1a1.** Unweighted data
- **L8a1a2.** Weighted data
- **L8a1d.** Inference (e.g., calibration) *(search also class L8a1a)*
- **L8a2.** $L_p$ for $p$ different from 2 (e.g., least absolute value, minimax)
- **L8a3.** Robust
- **L8a4.** Errors in variables

**L8b.** Polynomial (e.g., $y = b_0 + b_1 x + b_2 x^2$) *(search also class L8c)*

**L8b1.** Ordinary least squares
- **L8b1a.** Degree determination

**L8b1b.** Parameter estimation
- **L8b1b1.** Not using orthogonal polynomials
- **L8b1b2.** Using orthogonal polynomials
- **L8b1c.** Analysis *(search also class L8b1b)*
- **L8b1d.** Inference *(search also class L8b1b)*

**L8c.** Multiple linear (i.e., $y = b_0 + b_1 x_1 + \ldots + b_p x_p$)

**L8c1.** Ordinary least squares

**L8c1a.** Variable selection
- **L8c1a1.** Using raw data
- **L8c1a2.** Using correlation or covariance data
- **L8c1a3.** Using other data
- **L8c1b.** Parameter estimation *(search also class L8c1a)*
- **L8c1b1.** Using raw data
- **L8c1b2.** Using correlation data
- **L8c1c.** Analysis *(search also classes L8c1a and L8c1b)*
- **L8c1d.** Inference *(search also classes L8c1a and L8c1b)*
- **L8c2.** Several regressions
- **L8c3.** $L_p$ for $p$ different from 2
- **L8c4.** Robust
- **L8c5.** Measurement error models
- **L8c6.** Models based on ranks
- **L8d.** Polynomial in several variables

- **L8e.** Nonlinear (i.e., $y = F(X, b)$) *(search also class L8h)*
- • **L8e1.** Ordinary least squares
- • **L8e1a.** Variable selection
- **L8e1b.** Parameter estimation *(search also class L8e1a)*
- • **L8e1b1.** Unweighted data, user provides no derivatives
- • **L8e1b2.** Unweighted data, user provides derivatives
- • **L8e1b3.** Weighted data, user provides no derivatives
- • **L8e1b4.** Weighted data, user provides derivatives
- • **L8e2.** Ridge
- • **L8e5.** Measurement error models
- • **L8f.** Simultaneous (i.e., $Y = Xb$)
- • **L8g.** Spline (i.e., piecewise polynomial)
- • **L8h.** EDA (e.g., smoothing)
- • **L8i.** Service routines (e.g., matrix manipulation for variable selection)
- • **L9.** Categorical data analysis
- • **L9a.** 2-by-2 tables
- • **L9b.** Two-way tables *(search also class L9d)*
- • **L9c.** Log-linear model
- • **L9d.** EDA (e.g., median polish)
- **L10.** Time series analysis *(search also class J)*
- **L10a.** Univariate *(search also classes L3a6 and L3a7)*
- **L10a1.** Transformations
- • **L10a1a.** Elementary *(search also class L2a)*
- • **L10a1b.** Stationarity *(search also class L8a1)*
- **L10a1c.** Filters *(search also class K5)*
- • **L10a1c1.** Difference
- • **L10a1c2.** Symmetric linear (e.g., moving averages)
- • **L10a1c3.** Autoregressive linear
- • **L10a1c4.** Other
- • **L10a1d.** Taper
- • **L10a2.** Time domain analysis
- • **L10a2a.** Summary statistics
- • **L10a2a1.** Autocorrelations and autocovariances
- • **L10a2a2.** Partial autocorrelations
- • **L10a2b.** Stationarity analysis *(search also class L10a2a)*
- • **L10a2c.** Autoregressive models
- • **L10a2c1.** Model identification
- • **L10a2c2.** Parameter estimation
- • **L10a2d.** ARMA and ARIMA models (including Box-Jenkins methods)
- • **L10a2d1.** Model identification
- • **L10a2d2.** Parameter estimation
- • **L10a2d3.** Forecasting
- • **L10a2e.** State-space analysis (e.g., Kalman filtering)
- • **L10a2f.** Analysis of a locally stationary series
- • **L10a3.** Frequency domain analysis *(search also class J1)*
- **L10a3a.** Spectral analysis
- • **L10a3a1.** Pilot analysis
- • **L10a3a2.** Periodogram analysis
- • **L10a3a3.** Spectrum estimation using the periodogram
- • **L10a3a4.** Spectrum estimation using the Fourier transform of the autocorrelation function
- • **L10a3a5.** Spectrum estimation using autoregressive models
- • **L10a3a6.** Spectral windows

• contains software

- • **L10a3b.** Complex demodulation
- **L10b.** Two time series *(search also classes L3b3c, L10c, and L10d)*
- • **L10b2.** Time domain analysis
- • **L10b2a.** Summary statistics (e.g., cross-correlations)
- • **L10b2b.** Transfer function models
- • **L10b3.** Frequency domain analysis *(search also class J1)*
- **L10b3a.** Cross-spectral analysis
- • **L10b3a2.** Cross-periodogram analysis
- • **L10b3a3.** Cross-spectrum estimation using the cross-periodogram
- • **L10b3a4.** Cross-spectrum estimation using the Fourier transform of the cross-correlation or cross-covariance function
- • **L10b3a6.** Spectral functions
- • **L10c.** Multivariate time series *(search also classes J1, L3e3 and L10b)*
- • **L10d.** Two multi-channel time series
- • **L11.** Correlation analysis *(search also classes L4 and L13c)*
- • **L12.** Discriminant analysis
- **L13.** Covariance structure models
- • **L13a.** Factor analysis
- • **L13b.** Principal components analysis
- • **L13c.** Canonical correlation
- **L14.** Cluster analysis
- **L14a.** One-way
- **L14a1.** Unconstrained
- **L14a1a.** Nested
- • **L14a1a1.** Joining (e.g., single link)
- • **L14a1a2.** Divisive
- • **L14a1a3.** Switching
- • **L14a1a4.** Predict missing values
- • **L14a1b.** Non-nested (e.g., K means)
- • **L14a2.** Constrained
- • **L14b.** Two-way
- • **L14c.** Display
- • **L14d.** Service routines (e.g., compute distance matrix)
- • **L15.** Life testing, survival analysis
- • **L16.** Multidimensional scaling
- • **L17.** Statistical data sets

## M. **Simulation, stochastic modeling** *(search also classes L6 and L10)*

- M1. Simulation
- M1a. Discrete
- M1b. Continuous (Markov models)
- M2. Queueing
- M3. Reliability
- M3a. Quality control
- M3b. Electrical network
- M4. Project optimization (e.g., PERT)

## N. **Data handling** *(search also class L2)*

- • N1. Input, output

- • **N2.** Bit manipulation
- • **N3.** Character manipulation
- • **N4.** Storage management (e.g., stacks, heaps, trees)
- **N5.** Searching
  - • **N5a.** Extreme value
  - • **N5b.** Insertion position
  - • **N5c.** On a key
- **N6.** Sorting
  - **N6a.** Internal
    - • **N6a1.** Passive (i.e. construct pointer array, rank)
      - • **N6a1a.** Integer
      - • **N6a1b.** Real
      - • **N6a1c.** Character
    - • **N6a2.** Active
      - • **N6a2a.** Integer
      - • **N6a2b.** Real
      - • **N6a2c.** Character
  - **N6b.** External
- • **N7.** Merging
- • **N8.** Permuting

## O. Symbolic computation

## • P. Computational geometry *(search also classes G and Q)*

## • Q. Graphics *(search also class L3)*

## • R. Service routines

- • **R1.** Machine-dependent constants
- • **R2.** Error checking (e.g., check monotonicity)
- • **R3.** Error handling
  - • **R3a.** Set criteria for fatal errors
  - • **R3b.** Set unit number for error messages
  - • **R3c.** Other utilities
- • **R4.** Documentation retrieval

## S. Software development tools

- • **S1.** Program transformation tools
- • **S2.** Static program analysis tools
- • **S3.** Dynamic program analysis tools

## • Z. Other

• contains software

# Modules by Class

*This section is organized in the same way as the GAMS Classification Scheme. For each class there is a list of names and brief descriptions, often provided by the vendor, of software modules which solve the given problem. This list is organized alphabetically by application package and module name. Classes for which there is no software in GAMS are not listed.*

*In some cases, a module has a bullet (•) to the left of its name. This indicates that the module is an easy-to-use driver for other software modules in that package.*

*Introductory remarks are included at the highest levels of the classification scheme. These may address the computational issues associated with the software in that class and its subclasses, or describe conventions used in classifying the software. References to literature in the technical subject area may also be provided; most of these are available at the NIST-Gaithersburg Research Information Center (Administration Building, Room E120) and at the Department of Commerce Boulder Laboratories Main Library (Radio Building, Room 1202).*

# A: Arithmetic, Error Analysis

Modern scientific computers provide hardware to perform basic arithmetic on both fixed and floating-point numbers. Programming languages make these hardware facilities directly accessible through their implementations of integer and real arithmetic. Other arithmetic systems are also used in scientific computation. Complex arithmetic is the prime example; it is available in major scientific programming languages, and many software libraries make use of it. Other important examples are multiple precision integer and real arithmetic, rational arithmetic, and interval arithmetic. Software implementing non-standard, though elementary, arithmetic operations on a variety of data types is classified in this class.

Conversion from one data type to another is another elementary operation for which software exists. The PORT library, for example, provides a set of Fortran subprograms for performing standard type conversion, including the construction of machine-base numbers given a base-10 mantissa and exponent, and the inverse operation.

Finally, software for convergence acceleration is also listed in this class.

## A3a : Standard precision real arithmetic

*Collected Algorithms of the ACM*

**A532** A software package for roundoff analysis of noniterative numerical methods. (See W. Miller and D. Spooner, ACM TOMS 4 (1978) pp. 388-390.)

**A594** A Fortran program which performs an automatic roundoff error analysis of numerical algorithms. (See J.L. Larson, M.E. Pasternak, and J.A. Wisniewski, ACM TOMS 9 (1983) pp. 125-130.)

*NAG Subprogram Library*

**F06BLF** Quotient of real scalars, with overflow flag.

## A3c : Extended precision real arithmetic

*Collected Algorithms of the ACM*

**A524** MP: Fortran subroutines for performing multiple precision floating-point arithmetic and evaluating elementary and special functions. The subroutines are machine-independent and the precision is arbitrary, subject to storage limitations. (See R.P. Brent, ACM TOMS 4 (1978) pp. 71-81.)

*IMSL MATH/LIBRARY Subprogram Library*

**DQADD** Add a double precision scalar to the accumulator in extended precision.

**DQINI** Initialize an extended-precision accumulator with a double precision scalar.

**DQMUL** Multiply double precision scalars in extended precision.

**DQSTO** Store a double precision approximation to an extended-precision scalar.

## A3d : Extended range real arithmetic

*Collected Algorithms of the ACM*

**A567** NORMP: a Fortran subroutine for computing values of normalized Legendre polynomials of varying order and of fixed argument and degree. Also included is a package of six Fortran subroutines to facilitate the use of a special form of computer floating-point arithmetic called extended-range arithmetic. (See D.W. Lozier and J.M. Smith, ACM TOMS 7 (1981) pp. 141-146.)

## A4 : Complex arithmetic

*PORT Subprogram Library*

**CDADD** Sum of double precision complex numbers. Each is represented by a double precision array of two elements.

**CDDIV** Quotient of double precision complex numbers. Each is represented by a double precision array of two elements.

**CDMUL** Product of double precision complex numbers. Each is represented by a double precision array of two elements.

**CDSUB** Subtract double precision complex numbers. Each is represented by a double precision array of two elements.

*Scientific Desk PC Subprogram Library*

**A4BMD** Complex multiply.

**A4BMOD** Complex modulus.

**AZA4D** Complex divide.

## A4a : Standard precision complex arithmetic

*CMLIB Library (FNLIB Sublibrary)*

**CARG** Argument of a complex number.

*IMSL SFUN/LIBRARY Subprogram Library*

**CABS** Absolute value of a complex number.

**CARG** Argument of a complex number.

*NAG Subprogram Library*

**A02ABF** Modulus of a complex number.

**A02ACF** Quotient of complex numbers.

**C06GBF** Forms the complex conjugate of a Hermitian sequence of N data values.

**C06GCF** Forms the complex conjugate of a sequence of N data values.

**C06GQF** Forms the complex conjugates of M Hermitian sequences, each containing N data values.

**C06GSF** Takes M Hermitian sequences, each containing N data values, and forms the real and imaginary parts of the M corresponding complex sequences.

**F06CLF** Quotient of complex scalars, with overflow flag.

## A6 : Change of numeric representation

*MAGEV Cyber 205 Subprogram Library*

**CV64T60** To convert one or several Cyber 200 64-bit numbers to Cyber 170 60-bit format. The data type of the input numbers may be either all floating-point, all integer, or a mixture thereof. The data types will be preserved in the translation.

**CVF32TI2** To convert one or several half precision (32-bit floating-point) numbers to 16-bit two's complement integers, i.e., to the IBM data type INTEGER*2.

**CVF32TR4** To convert one or several half precision (32-bit floating-point) numbers to IBM 32-bit floating-point format, also known as REAL*4.

**CVF64VR4** To convert one or several Cyber 200 64-bit floating-point numbers to VAX 32-bit floating-point format.

**CVI2TF32** To convert one or several 16-bit two's complement integers, corresponding to IBM INTEGER*2 format, to Cyber 200 half precision format, i.e., into 32-bit floating-point numbers.

**CVR4TF32** To convert IBM 32-bit floating-point numbers, also known as REAL*4, to Cyber 200 half precision format, i.e., into 32-bit floating-point numbers.

**CVVR4F64** To convert one or several VAX 32-bit floating-point numbers to Cyber 200 64-bit floating-point format.

## A6a : Numeric type conversion

*PORT Subprogram Library*

**CNVBDC** Converts double precision vector into complex vector (backward loop).

**CNVBDI** Converts double precision vector into integer vector (backward loop).

**CNVBDR** Converts double precision vector into real vector (backward loop).

**CNVBIC** Converts integer vector into complex vector (backward loop).

**CNVBID** Converts integer vector to double precision vector (backward loop).

**CNVBIR** Converts integer vector into real vector (backward loop).

**CNVBRC** Converts real vector into complex vector (backward loop).

**CNVBRD** Converts real vector into double precision vector (backward loop).

**CNVBRI** Converts real vector into integer vector (backward loop).

**CNVFDC** Converts double precision vector into complex vector (forward loop).

**CNVFDI** Converts double precision vector into integer vector (forward loop).

**CNVFDR** Converts double precision vector into real vector (forward loop).

**CNVFIC** Converts integer vector into complex vector (forward loop).

**CNVFID** Converts integer vector into double precision vector (forward loop).

**CNVFIR** Converts integer vector into real vector (forward loop).

**CNVFRC** Converts real vector into complex vector (forward loop).

**CNVFRD** Converts real vector into double precision vector (forward loop).

**CNVFRI** Converts real vector into integer vector (forward loop).

## A6b : Numeric base conversion

*PORT Subprogram Library*

**VBTOD** Convert a machine-base mantissa and exponent into a base 10 floating-point number.

**VDTOB** Convert a base-10 mantissa and exponent into an machine-base floating-point number.

## A6c : Decomposition, construction of machine numbers

*PORT Subprogram Library*

**MKFL** Given m, the mantissa of a floating-point number, and e, its exponent, MKFL returns $MB^e$, where B is the base of the machine.

**UMKFL** Decompose a non-zero floating-point number into a mantissa and an exponent.

| **A7 :** | **Sequences (e.g., convergence acceleration)** |
|---|---|

*Collected Algorithms of the ACM*

**A585** EXTRAP: A Fortran subroutine for sequence extrapolation (based upon the E-algorithm) and generalized interpolation by a linear combination of functions forming a Chebyshev system (based upon the, Muhlbach-Neville-Aitken algorithm). Includes the Epsilon Algorithm of Wynn as a special case. (See C. Brezinski, ACM TOMS 8 (1982) pp. 290-301.)

**A602** HURRY: a Fortran subprogram for accelerating the convergence of alternating and monotone sequences and series (based on Levin's u transform). The routine estimates truncation and roundoff errors to make a near-optimal stopping decision and provide a good estimate of the accuracy. (See T. Fessler, W.F. Ford, and D.A. Smith, ACM TOMS 9 (1983) pp. 355-357.)

*CMLIB Library (QUADPKS Sublibrary)*

**EA** Given a slowly convergent sequence, this routine extrapolates nonlinearly to a better estimate of its limit, thus improving the rate of convergence. Based on the Epsilon Algorithm of P. Wynn. An estimate of the absolute error is also given.

*NAG Subprogram Library*

**C06BAF** Accelerates the convergence of a given convergent sequence to its limit.

**C06GBF** Forms the complex conjugate of a Hermitian sequence of N data values.

**C06GCF** Forms the complex conjugate of a sequence of N data values.

**C06GQF** Forms the complex conjugates of M Hermitian sequences, each containing N data values.

**C06GSF** Takes M Hermitian sequences, each containing N data values, and forms the real and imaginary parts of the M corresponding complex sequences.

# B: Number Theory

Software classified in this class performs such number-theoretic calculations as the decomposition of integers into prime factors.

| **B :** | **Number theory** |
|---|---|

*Collected Algorithms of the ACM*

**A515** COMB: a Fortran subroutine which generates a vector from a lexicographical index. That is, let $C_1$, $C_2$, ... $C_m$ be the set of combinations of n items taken p at a time arranged in lexographical order. Given an integer $i$, this routine finds $C_i$. (See B.P. Buckles and M. Lybanon, ACM TOMS 3 (1977) pp. 180-182.)

**A536** PURDY: a Fortran subroutine for evaluating Purdy's irreversible enciphering function. It serves as a machine independent model for studying the evaluation of polynomials mod P and for the implementation of more efficient machine dependent system utility programs for enciphering passwords. (See H.D. Knoble, ACM TOMS 5 (1979) pp. 108-111.)

*IMSL Subprogram Library*

**VDCPS** Decompose an integer into its prime factors.

*IMSL MATH/LIBRARY Subprogram Library*

**PRIME** Decompose an integer into its prime factors.

# C: Elementary and Special Functions

This class contains computer programs for operations on selected mathematical functions. The provision of such software is still a wide open field. Some idea of the difficulties involved is given in the remainder of this introduction. The reader who is interested only in what is currently available should refer immediately to the categories below for further specific information. For the special distribution functions of statistics see also class L5.

Some elementary functions are provided by the Fortran compiler. Compilers conforming to the ANSI standard (ANSI X3.9 — 1978) include the square root function, the exponential and logarithmic functions, the circular functions and inverses, and the hyperbolic functions in single and double precision, and in some cases complex. Double precision complex versions of some of these functions, plus the error, complementary error, gamma, and log gamma functions in single and double precision are provided by some compilers as extensions to the standard. When the same Fortran program is to be run on computers made by different manufacturers, it is wise not to rely on extensions to the language standard. Therefore, portable Fortran programs for mathematical functions, other than the ones required by the Fortran standard are of great interest.

The major difficulty in implementating portable special function software is how to achieve efficiently the requisite accuracy for different computer wordlengths. This can be seen by considering how a special function is computed. Some form of approximation is always used, often derived from a truncated infinite expansion, and thus, the more accuracy required, the more expensive the computation. An efficient portable function must be self-adaptive in the sense of being able to select the most suitable approximation from a set of possible ones.

There are algorithmic difficulties as well. Rounding errors inevitably occur in finite precision computation, and these can accumulate so that the final result is much less accurate than expected. The algorithm that is used should be stable in the sense that it should not amplify rounding errors as the computation progresses.

The accuracy of functions implemented in the compiler is generally higher than the accuracy of functions in a portable library. This is because it is more convenient and less costly to use assembly language instead of Fortran to perform the computation in extended precision.

Progress has been made in meeting these difficulties in recent years. There now exist fairly widely accepted Fortran calls that return static information about the number of bits in a computer word, the exponent range, etc., for both single and double precision. This information has been used effectively in some cases to solve the algorithm selection problem. Research continues on developing better algorithms and computer arithmetic, on improving the design of Fortran subroutines for mathematical functions, and on developing standards for programming languages and function libraries. This work will lead to mathematical function software that is easier to use, gives fuller coverage of input ranges of interest, and that has more versatile modes of recovery from faults. It will also lead to a much wider selection of mathematical functions in portable libraries.

Certain conventions have been followed in the module descriptions that follow. Unless otherwise specified, (a) the variables $x$ and $y$ are real, (b) the variables $z$, $z1$, and $z2$ are complex, and (c) angles are measured in radians.

## References

[AMS55] M. Abramowitz and I. A. Stegun, editors. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* U. S. Government Printing Office, Washington, D.C., 1964.

[Cod74] W. J. Cody. The construction of numerical subroutine libraries. *SIAM Review*, 16(1):36–46, 1974.

[CW80] W. J. Cody and W. M. Waite. *Software Manual for the Elementary Functions.* Prentice-Hall, Englewood Cliffs, NJ, 1980.

[Fik68] C. T. Fike. *Computer Evaluation of Mathematical Functions.* Prentice-Hall, Englewood Cliffs, NJ, 1968.

[HCL68] J. F. Hart, E. W. Cheney, C. L. Lawson, et al. *Computer Approximations.* John Wiley & Sons, New York, 1968.

**C1 : Integer-valued functions (e.g., factorial, binomial coefficient, permutations, combinations, floor, ceiling)**

*Collected Algorithms of the ACM*

**A515** COMB: a Fortran subroutine which generates a vector from a lexicographical index. That is, let $C_1$, $C_2$, ... $C_m$ be the set of combinations of n items taken p at a time arranged in lexographical order. Given an integer $i$, this routine finds $C_i$. (See B.P. Buckles and M. Lybanon, ACM TOMS 3 (1977) pp. 180-182.)

*CMLIB Library (FNLIB Sublibrary)*

**BINOM** Binomial coefficient. Input is integer, output is real.

**FAC** Factorial. Input is integer, output is real.

*IMSL STAT/LIBRARY Subprogram Library*

**BINOM** Binomial coefficient. Input is integer, output is real.

*IMSL SFUN/LIBRARY Subprogram Library*

**AINT** Integer part of x. Input and output are real.

**BINOM** Binomial coefficient. Input is integer, output is real.

**FAC** Factorial. Input is integer, output is real.

*PORT Subprogram Library*

**CEIL** Smallest integer greater than or equal to x. Input and output are real.

**FLR** Largest integer less than or equal to x. Input and output are real.

**ICEIL** Smallest integer greater than or equal to x. Input is real, output is integer.

**IFLR** Largest integer less than or equal to x. Input is real, output is integer.

**C2 : Powers, roots, reciprocals**

*Collected Algorithms of the ACM*

**A650** Assembly language programs for efficient square root implementation on the Motorola 68000. (See K. C. Johnson, ACM TOMS 13 (1987) pp. 138-151.)

*CMLIB Library (FNLIB Sublibrary)*

**CBRT** Cube root of a real argument.

**CCBRT** Cube root of a complex argument.

**CSQRT** Square root of a complex argument.

**SQRT** Square root of a real argument.

*IMSL MATH/LIBRARY Subprogram Library*

**HYPOT** $\sqrt{A^2 + B^2}$ without underflow or overflow.

*IMSL SFUN/LIBRARY Subprogram Library*

**CBRT** Cube root of a real argument.

**CCBRT** Cube root of a complex argument.

**CSQRT** Square root of a complex argument.

**SQRT** Square root of a real argument.

*NAG Subprogram Library*

**A02AAF** Square root of a complex argument.

*Scientific Desk PC Subprogram Library*

**C2CCUB** Cube root of a real argument.

**C2CSQD** Square root of a complex argument.

**C2CUB** Cube root of a real argument.

## C3a : Orthogonal polynomials

*IMSL Subprogram Library*

**RLPOL** Generate orthogonal polynomials with associated constants AA and BB.

*IMSL STAT/LIBRARY Subprogram Library*

**OPOLY** Generate orthogonal polynomials with respect to a specified interval and weights.

*IMSL SFUN/LIBRARY Subprogram Library*

**INITS** Returns the number of terms of a given orthogonal series needed to insure that the error is no larger than the requested accuracy.

*PORT Subprogram Library*

**ORTHP** Evaluates a polynomial expressed as a sum of general orthogonal polynomials.

## C3a1 : Trigonometric polynomials

*PORT Subprogram Library*

**TRIGP** Evaluates a trigonometric polynomial.

## C3a2 : Chebyshev, Legendre polynomials

*Collected Algorithms of the ACM*

**A567** NORMP: a Fortran subroutine for computing values of normalized Legendre polynomials of varying order and of fixed argument and degree. Also included is a package of six Fortran subroutines to facilitate the use of a special form of computer floating-point arithmetic called extended-range arithmetic. (See D.W. Lozier and J.M. Smith, ACM TOMS 7 (1981) pp. 141-146.)

*CMLIB Library (FCNPAK Sublibrary)*

**XSLEGF** Calculates a sequence of values of Legendre functions of the first or second kind of positive or negative order. The argument and either degree or order are fixed.

**XSNRMP** Calculates a sequence of values of normalized Legendre polynomials for fixed degree and argument and variable order.

*CMLIB Library (FNLIB Sublibrary)*

**CSEVL** Evaluates a Chebyshev polynomial series.

*CMLIB Library (QUADPKS Sublibrary)*

**QMOMO** Computes integral of k-th degree Chebyshev polynomial times one of a selection of functions with various singularities.

*IMSL SFUN/LIBRARY Subprogram Library*

**CSEVL** Evaluate a series of Chebyshev polynomials.

*NAG Subprogram Library*

**C06DBF** Returns the value of the sum of a Chebyshev series through the routine name.

**E02AEF** Evaluates a polynomial from its Chebyshev series representation.

**E02AHF** Determines the coefficients in the Chebyshev series representation of the derivative of a polynomial given in Chebyshev series form.

**E02AJF** Determines the coefficients in the Chebyshev series representation of the indefinite integral of a polynomial given in Chebyshev series form.

**E02AKF** Evaluates a polynomial from its Chebyshev series representation, allowing an arbitrary index increment for accessing the array of coefficients.

*PORT Subprogram Library*

**TCHBP** Evaluates a polynomial expressed as a sum of Chebyshev polynomials.

## C3b : Non-orthogonal polynomials

*Collected Algorithms of the ACM*

**A604** EXTREM: a Fortran subprogram for the calculating extremal polynomials. If L is a linear functional on polynomials of degree n or less, then p is extremal if it is a polynomial of (Chebyshev) norm one at which L takes on its norm. (See F.W. Sauer, ACM TOMS 9 (1983) pp. 381-383.)

**A628** GROEB: a Fortran program for constructing canonical (or Groebner) bases of polynomial ideals. (See F. Winkler et al., ACM TOMS 11 (1985) pp. 66-78.)

*Scientific Desk PC Subprogram Library*

**F1A1E** Evaluates a complex polynomial and its derivatives, with optional error bounds.

## C4 : Elementary transcendental functions

*IMSL Subprogram Library*

**MMLINC** Evaluates an elementary integral from which inverse circular functions, logarithms or inverse hyperbolic functions may be computed.

## C4a : Trigonometric, inverse trigonometric functions

*CMLIB Library (FNLIB Sublibrary)*

**CACOS** arccos(z).

**CASIN** arcsin(z).

**CATAN** arctan(z).

**CATAN2** arctan(z1/z2).

**CCOS** cos(z).

**CCOT** cot(z).

**COS** cos(x).

**COSDG** cos(x), x in degrees.

| | |
|---|---|
| **COT** | cot(x). |
| **CSIN** | sin(z). |
| **CTAN** | tan(z). |
| **DACOS** | arccos(x), for double precision x. |
| **DASIN** | arcsin(x), for double precision x. |
| **DATAN** | arctan(x), for double precision x. |
| **DATAN2** | arctan(x/y), for double precision x, y. |
| **SIN** | sin(x). |
| **SINDG** | sin(x), x in degrees. |
| **TAN** | tan(x). |

*IMSL SFUN/LIBRARY Subprogram Library*

| | |
|---|---|
| **ACOS** | arccos(x). |
| **ACOSH** | arccosh(x). |
| **ASIN** | arcsin(x). |
| **ASINH** | arcsinh(x). |
| **ATAN** | arctan(x). |
| **ATAN2** | arctan(x/y). |
| **ATANH** | arctanh(x). |
| **CACOS** | arccos(z). |
| **CACOSH** | arccosh(z). |
| **CASIN** | arcsin(z). |
| **CASINH** | arcsinh(z). |
| **CATAN** | arctan(z). |
| **CATAN2** | arctan(z1/z2). |
| **CATANH** | arctanh(z). |
| **CCOS** | cos(z). |
| **CCOSH** | cosh(z). |
| **CCOT** | cotan(z). |
| **COS** | cos(x). |
| **COSDG** | cos(x), x in degrees. |
| **COSH** | cosh(x). |
| **COT** | cotan(x). |
| **CSIN** | sin(z). |
| **CSINH** | sinh(z). |
| **CTAN** | tan(z). |
| **CTANH** | tanh(z). |
| **SIN** | sin(x). |
| **SINDG** | sin(x), x in degrees. |
| **SINH** | sinh(x). |
| **TAN** | tan(x). |
| **TANH** | tanh(x). |

*NAG Subprogram Library*

| | |
|---|---|
| **F06BCF** | Recover real cosine c and read sine s from given real tangent t, i.e., $c = 1/(1+t^2)^{1/2}$ and s = tc. |
| **F06CCF** | Recover real cosine c and complex sine s from given complex tangent t, i.e., $c = 1/(1+\|t\|^2)^{1/2}$ and s = tc. |
| **F06CDF** | Recover complex cosine c and real sine s from given complex tangent t, i.e., $c = (\mathrm{sign}(\mathrm{real}(t))\|t\|)/(t(1+\|t\|^2)^{1/2})$ and s = tc. |
| **S07AAF** | tan(x). |
| **S09AAF** | arcsin(x). |
| **S09ABF** | arccos(x). |

*PORT Subprogram Library*

| | |
|---|---|
| **ARCOS** | arccos(x). |
| **ARSIN** | arcsin(x). |
| **TAN** | tan(x). |

*Scientific Desk PC Subprogram Library*

| | |
|---|---|
| **C4ACCO** | arccos(z). |
| **C4ACCT** | cotan(x). |
| **C4ACSN** | arcsin(z). |
| **C4ACT** | cotan(x). |
| **C4ACT2** | arctan(z1/z2). |
| **C4ACTI** | arctan(z). |
| **C4ACTN** | tan(z). |

## C4b : Exponential, logarithmic functions

*CMLIB Library (FNLIB Sublibrary)*

| | |
|---|---|
| **ALNREL** | ln(x+1). |
| **ALOG** | ln(x). |
| **ALOG10** | log(x). |
| **CEXP** | exp(z). |
| **CEXPRL** | (exp(z)-1)/z. |
| **CLNREL** | ln(1+z). |
| **CLOG** | ln(z). |
| **CLOG10** | log(z). |
| **EXP** | exp(x). |
| **EXPREL** | (exp(x)-1)/x. |

*IMSL SFUN/LIBRARY Subprogram Library*

| | |
|---|---|
| **ALNREL** | ln(1+x). |
| **ALOG** | ln(x). |
| **ALOG10** | log(x). |
| **CEXP** | exp(z). |
| **CEXPRL** | (exp(z)-1)/z. |
| **CLNREL** | ln(1+z). |

| | |
|---|---|
| **CLOG** | ln(z). |
| **CLOG10** | log(z). |
| **EXP** | exp(x). |
| **EXPRL** | (exp(x)-1)/x. |

*PORT Subprogram Library*

| | |
|---|---|
| **CDEXP** | exp(z) for complex double precision z. |
| **CDLOG** | ln(z) for complex double precision z. |

*Scientific Desk PC Subprogram Library*

| | |
|---|---|
| **C4BALR** | ln(1+x). |
| **C4BCLD** | ln(z). |
| **C4BCLR** | ln(1+x). |
| **C4BCXD** | exp(z). |
| **C4BCXP** | (exp(z)-1)/z. |
| **C4BEXP** | (exp(x)-1)/x. |

## C4c : Hyperbolic, inverse hyperbolic functions

*CMLIB Library (FNLIB Sublibrary)*

| | |
|---|---|
| **ACOSH** | arccosh(x). |
| **ASINH** | arcsinh(x). |
| **ATANH** | arctanh(x). |
| **CACOSH** | arccosh(z). |
| **CASINH** | arcsinh(z). |
| **CATANH** | arctanh(z). |
| **CCOSH** | cosh(z). |
| **COSH** | cosh(x) . |
| **CSINH** | sinh(z). |
| **CTANH** | tanh(z). |
| **SINH** | sinh(x). |
| **TANH** | tanh(x). |

*NAG Subprogram Library*

| | |
|---|---|
| **S10AAF** | tanh(x). |
| **S10ABF** | sinh(x). |
| **S10ACF** | cosh(x). |
| **S11AAF** | arctanh(x). |
| **S11ABF** | arcsinh(x). |
| **S11ACF** | arccosh(x). |

*PORT Subprogram Library*

| | |
|---|---|
| **ACOSH** | arccosh(x). |
| **ASINH** | arcsinh(x). |
| **ATANH** | arctanh(x). |
| **COSH** | cosh(x). |
| **SINH** | sinh(x). |
| **TANH** | tanh(x). |

*Scientific Desk PC Subprogram Library*

**C4CACH** arccosh(z).
**C4CASH** arcsinh(z).
**C4CATH** arctanh(z).
**C4CCH** cosh(z).
**C4CSH** sinh(z).
**C4CSHD** Complex sinh.
**C4CTH** tanh(z).

*XMLIBV Cyber 205 Subprogram Library*

**VCOSH** cosh(x), for a real vector x.
**VSINH** sinh(x), for a real vector x.

## C5 : Exponential and logarithmic integrals

*Collected Algorithms of the ACM*

**A556** EXPINT: a Fortran subroutine for computing sequences of exponential integrals E(n+k,x), k=0,1,. . .,m-1 for n≥1, and x≥0. (See D.E. Amos, ACM TOMS 6 (1980) pp. 420-428.)

*CMLIB Library (AMOSLIB Sublibrary)*

**EXINT** Computes sequences of exponential integrals E(n+k,x) k=0,. . .,m-1 or exp(x) times same to specified tolerance.

*CMLIB Library (FNLIB Sublibrary)*

**ALI** Logarithmic integral, integral from 0 to x of 1/ln(t).
**E1** Integral from x to infinity of exp(-t)/t.
**EI** Integral from -x to infinity of -exp(-t)/t. Exponential integral for arguments greater than zero and the Cauchy principal value for arguments less than zero.
**SPENC** Integral from 0 to x of ln(|1-t|)/t.

*IMSL Subprogram Library*

**MMDEI** Exponential integrals.
**MMDEN** Exponential integrals of integer order for real argument x scaled by exp(x).

*IMSL SFUN/LIBRARY Subprogram Library*

**ALI** Logarithmic integral, integral from 0 to x of 1/ln(t).
**CHI** Hyperbolic cosine integral.
**CI** Cosine integral.
**CIN** Evaluate a function closely related to the cosine integral.
**CINH** Evaluate a function closely related to the hyperbolic cosine integral.
**E1** Exponential integral for arguments greater than zero and the Cauchy principal value of the integral for arguments less than zero.
**EI** Integral from -x to infinity of -exp(-t)/t. Exponential integral for arguments greater than zero and the Cauchy principal value for arguments less than zero.
**ENE** Exponential integral of integer order for arguments greater than zero scaled by exp(x).
**SHI** Hyperbolic sine integral.
**SI** Sine integral.
**SPENC** Evaluate Spence's integral, see [AMS55 (27.7.1)].

*NAG Subprogram Library*

**S13AAF** Exponential integral, e1(x).

*Scientific Desk PC Subprogram Library*

**C5** Computes various logarithmic and exponential integrals.

*SPECFN Software Library*

**EXPINT** Exponential integrals and scaled exponential integrals.

**SICIEI** Sine, cosine, exponential, scaled exponential, hyperbolic sine and hyperbolic cosine integrals.

## C6 : Cosine and sine integrals

*NAG Subprogram Library*

**S13ACF** Cosine integral, ci(x).

**S13ADF** Sine integral, si(x).

*SPECFN Software Library*

**SICIEI** Sine, cosine, exponential, scaled exponential, hyperbolic sine and hyperbolic cosine integrals.

## C7a : Gamma, log gamma, reciprocal gamma functions

*CMLIB Library (AMOSLIB Sublibrary)*

**GAMLN** $\ln(\Gamma(x))$, for non-negative x.

*CMLIB Library (FNLIB Sublibrary)*

**ALGAMS** $\ln(|\Gamma(x)|)$ and $\text{sign}(\Gamma(x))$.

**ALNGAM** $\ln(|\Gamma(x)|)$.

**CGAMMA** $\Gamma(z)$.

**CGAMR** $1/\Gamma(z)$.

**CLNGAM** $\ln(\Gamma(z))$.

**GAMMA** $\Gamma(x)$.

**GAMR** $1/\Gamma(x)$.

**POCH** Pochhammer's symbol. Input and output are real.

**POCH1** Pochhammer's symbol from first order. Input and output are real.

*IMSL Subprogram Library*

**ALGAMA** $\ln(|\Gamma(x)|)$.

**GAMMA** $\Gamma(x)$.

*IMSL MATH/LIBRARY, STAT/LIBRARY and SFUN/LIBRARY Subprogram Libraries*

**GAMMA** $\Gamma(x)$.

*IMSL SFUN/LIBRARY Subprogram Library*

**ALGAMS** $\ln(|\Gamma(x)|)$ and $\text{sign}(\Gamma(x))$.

**ALNGAM** $\ln(|\Gamma(x)|)$.

**CGAMMA** Complex gamma function.

**CGAMR** $1/\Gamma(z)$.

| | |
|---|---|
| **CLNGAM** | ln(Γ(z)). |
| **CPSI** | Logarithmic derivative of the gamma function for a complex argument. |
| **GAMR** | 1/Γ(x). |
| **POCH** | Generalization of Pochhammer's symbol. |
| **POCH1** | Generalization of Pochhammer's symbol starting from first order. |

*NAG Subprogram Library*

| | |
|---|---|
| **S14AAF** | Γ(x). |
| **S14ABF** | ln(Γ(x)). |

*NMS Subprogram Library*

| | |
|---|---|
| **GAMMA** | Γ(x). |

*Scientific Desk PC Subprogram Library*

| | |
|---|---|
| **C7ACLG** | Complex log of the Γ function. |
| **C7AG** | Γ(x). |
| **C7ALG** | ln(Γ(x)). |

## C7b : Beta, log beta functions

*CMLIB Library (FNLIB Sublibrary)*

| | |
|---|---|
| **ALBETA** | ln(Beta(a,b)), for positive real a,b. |
| **BETA** | Beta(a,b), for positive real a,b. |
| **CBETA** | Beta(a,b), for complex a,b (with positive real parts). |
| **CLBETA** | ln(Beta(a,b)), for complex a,b (with positive real parts). |

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

| | |
|---|---|
| **BETA** | Complete beta function. |
| **BETDF** | Beta probability distribution function. |
| **BETIN** | Inverse of the beta distribution function. |

*IMSL SFUN/LIBRARY Subprogram Library*

| | |
|---|---|
| **ALBETA** | Logarithm of the complete beta function for positive arguments. |
| **BETAI** | Incomplete beta function. |
| **CBETA** | Complex complete beta function. |
| **CLBETA** | Complex logarithm of the complete beta function. |

*Scientific Desk PC Subprogram Library*

| | |
|---|---|
| **C7BCLB** | Log of the complex beta function. |
| **C7BLB** | Log of the complete beta function. |
| **C7BTA** | Complete beta function. |

## C7c : Psi function

*Collected Algorithms of the ACM*

**A610** PSIFN: a Fortran subprogram for calculating derivatives of the psi function. A double precision version is included. (See D.E. Amos, ACM TOMS 9 (1983) pp. 494-502.)

*CMLIB Library (AMOSLIB Sublibrary)*

**PSIFN** Derivatives of the Psi function.

*CMLIB Library (FNLIB Sublibrary)*

**CPSI** Psi (digamma) of complex argument.
**PSI** Psi (digamma) of real argument.

*IMSL Subprogram Library*

**MMPSI** Logarithmic derivative of the gamma function.

*IMSL SFUN/LIBRARY Subprogram Library*

**PSI** Logarithmic derivative of the gamma function.

*Scientific Desk PC Subprogram Library*

**C7BCPS** Psi function.
**C7BPSI** Psi function.

## C7e : Incomplete gamma function

*Collected Algorithms of the ACM*

**A542** GAM: a Fortran subroutine based on Taylor's series and continued fractions for evaluating Tricomi's incomplete gamma function and the complementary incomplete gamma function. (See W. Gautschi, ACM TOMS 5 (1979) pp. 482-489.)

**A654** GRATIO and GAMINV: Fortran routines for computing the incomplete gamma function ratios and their inverse. (See A. R. DiDonato and A. H. Morris, Jr., ACM TOMS 13 (1987) pp. 318-319.)

*CMLIB Library (FNLIB Sublibrary)*

**GAMI** Incomplete gamma function.
**GAMIC** Complementary incomplete gamma function.
**GAMIT** Tricomi's incomplete gamma function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**CHIDF** Chi-squared distribution function.
**CHIIN** Inverse of the chi-squared distribution function.
**GAMDF** Gamma distribution function.

*IMSL SFUN/LIBRARY Subprogram Library*

**GAMI** Incomplete gamma function.
**GAMIC** Complementary incomplete gamma function.
**GAMIT** Tricomi's incomplete gamma function.

*Scientific Desk PC Subprogram Library*

**C7EG** Incomplete gamma function.
**C7EGC** Complementary incomplete gamma function.

| C7f : | Incomplete beta function |
|---|---|

*CMLIB Library (FNLIB Sublibrary)*

**BETAI** Incomplete Beta function.

| C8a : | Error functions, their inverses, integrals, including the normal distribution function |
|---|---|

*Collected Algorithms of the ACM*

**A521** INERFC: a Fortran subroutine for evaluating repeated integrals of the coerror function. (See W. Gautschi, ACM TOMS 3 (1977) pp. 301-302.)

*CMLIB Library (FNLIB Sublibrary)*

**ERF** Error function, $= 2/\pi^{1/2}\times$ the integral from 0 to x of exp(-$t^2$)dt.

**ERFC** Complementary error function, $= 2/\pi^{1/2}\times$ the integral from x to infinity of exp(-$t^2$)dt.

*IMSL Subprogram Library*

**ERF** Error function, $= 2/\pi^{1/2}\times$ the integral from 0 to x of exp(-$t^2$)dt.

**ERFC** Complementary error function, $= 2/\pi^{1/2}\times$ the integral from x to infinity of exp(-$t^2$)dt.

**MDNOR** Normal or Gaussian probability distribution function.

**MERFCI** Inverse complementary error function.

**MERFI** Inverse error function.

**MERRCZ** Computes approximate values of exp(-$z^2$)erfc(-iz) for complex z.

*IMSL STAT/LIBRARY Subprogram Library*

**ANORDF** Normal (Gaussian) distribution function.

**ANORIN** Inverse of the normal (Gaussian) distribution function.

*IMSL SFUN/LIBRARY Subprogram Library*

**ANORDF** Normal (Gaussian) distribution function.

**ANORIN** Inverse of the normal (Gaussian) distribution function.

**CERFE** Complex scaled complementary error function.

**ERF** Error function, $= 2/\pi^{1/2}\times$ the integral from 0 to x of exp(-$t^2$)dt.

**ERFC** Complementary error function, $= 2/\pi^{1/2}\times$ the integral from x to infinity of exp(-$t^2$)dt.

**ERFCE** Exponentially scaled complementary error function.

**ERFCI** Inverse complementary error function.

**ERFI** Inverse error function.

*NAG Subprogram Library*

**S15ABF** Cumulative normal distribution function.

**S15ACF** Complement of cumulative normal distribution function.

**S15ADF** Complement of error function, erfc(x).

**S15AEF** Error function, erf(x).

*NMS Subprogram Library*

**ERF** Error function, $= 2/\pi^{1/2}\times$ the integral from 0 to x of exp(-$t^2$)dt.

**ERFC** Complementary error function, $= 2/\pi^{1/2}\times$ the integral from x to infinity of exp(-$t^2$)dt.

*Scientific Desk PC Subprogram Library*

**C8AE** Error function.

**C8AEI** Inverse of the error function.

*SPECFN Software Library*

**ERRINT** Error function and complementary error function.

## C8b : Fresnel integrals

*NAG Subprogram Library*

**S20ACF** Fresnel integrals, s(x).

**S20ADF** Fresnel integrals, c(x).

## C8c : Dawson's integral

*CMLIB Library (FNLIB Sublibrary)*

**DAWS** Dawson's integral.

*IMSL Subprogram Library*

**MMDAS** Dawson's integral.

*IMSL SFUN/LIBRARY Subprogram Library*

**DAWS** Dawson's integral.

*NAG Subprogram Library*

**S15AFF** Dawson's integral.

*Scientific Desk PC Subprogram Library*

**C8CD** Dawson's integral.

## C9 : Legendre functions

*Collected Algorithms of the ACM*

**A567** NORMP: a Fortran subroutine for computing values of normalized Legendre polynomials of varying order and of fixed argument and degree. Also included is a package of six Fortran subroutines to facilitate the use of a special form of computer floating-point arithmetic called extended-range arithmetic. (See D.W. Lozier and J.M. Smith, ACM TOMS 7 (1981) pp. 141-146.)

*CMLIB Library (FCNPAK Sublibrary)*

**XSLEGF** Calculates a sequence of values of Legendre functions of the first or second kind of positive or negative order. The argument and either degree or order are fixed.

**XSNRMP** Calculates a sequence of values of normalized Legendre polynomials for fixed degree and argument and variable order.

## C10a : Bessel functions J, Y, $H_1$, $H_2$

*BESPAK Software Library*

**BESPAK** Subroutines for Bessel and modified Bessel functions of complex argument and real (integer or fractional) order. BESPAK consists of four main subroutines, one for each Bessel function, and four subsidiary routines.

## C10a1 : Bessel functions J, Y, and H of real argument and integer order

*CMLIB Library (FNLIB Sublibrary)*

**BESJ0** Bessel function $J_0(x)$.
**BESJ1** Bessel function $J_1(x)$.
**BESY0** Modified Bessel function $Y_0(x)$.
**BESY1** Modified Bessel function $Y_1(x)$.

*IMSL Subprogram Library*

**MMBSJ0** Bessel function $J_0(x)$.
**MMBSJ1** Bessel function $J_1(x)$.
**MMBSJN** Sequence of Bessel functions $J_n(x)$.

*IMSL SFUN/LIBRARY Subprogram Library*

**BSJ0** Bessel function $J_0(x)$.
**BSJ1** Bessel function $J_1(x)$.
**BSJNS** Sequence of Bessel functions $J_n(x)$.
**BSY0** Bessel function $Y_0(x)$.
**BSY1** Bessel function $Y_1(x)$.

*NAG Subprogram Library*

**S17ACF** Bessel functions $Y_0(x)$.
**S17ADF** Bessel functions $Y_1(x)$.
**S17AEF** Bessel functions $J_0(x)$.
**S17AFF** Bessel functions $J_1(x)$.

*PORT Subprogram Library*

**BESRJ** Bessel functions, J, of real argument and integer order.

*Scientific Desk PC Subprogram Library*

**C10AJ0** Bessel function $J_0(x)$.
**C10AJ1** Bessel function $J_1(x)$.
**C10AY0** Bessel function $Y_0$ (and $J_0$).
**C10AY1** Bessel function $Y_1$ (and $J_1$).

*XMLIBV Cyber 205 Subprogram Library*

**VBESJ0** Bessel function $J_0(x)$ for a vector of real arguments.
**VBESJ1** Bessel function $J_1(x)$ for a vector of real arguments.
**VBESY0** Bessel function $Y_0(x)$ for a vector of real arguments.
**VBESY1** Bessel function $Y_1(x)$ for a vector of real arguments.

## C10a2 : Bessel functions J, Y, and H of complex argument and integer order

*CMLIB Library (AMOSLIB Sublibrary)*

**CJYHBS** Bessel and Struve functions $J_0(z)$, $J_1(z)$, $Y_0(z)$, $Y_1(z)$, $H_0(z)$, $H_1(z)$.

*IMSL Subprogram Library*

**MMBZJN** Sequence of Bessel functions $J_n(z)$.

*IMSL SFUN/LIBRARY Subprogram Library*

**CBJNS** Sequence of Bessel functions $J_n(z)$.

*PORT Subprogram Library*

**BESCJ** Sequence of Bessel functions $J_n(z)$.

## C10a3 : Bessel functions J, Y, and H of real argument and real order

*Collected Algorithms of the ACM*

**A511** IBESS and JBESS: CDC 6600 Fortran subroutines for Bessel functions $I_v(x)$ and $J_v(x)$, for real $x \geq 0$, and real $v \geq 0$. (See D.E. Amos, S.L. Daniel, and M.K. Weston, ACM TOMS 3 (1977) pp. 93-95.)

*CMLIB Library (AMOSLIB Sublibrary)*

**BESJ** Sequence of Bessel functions, $J_{\alpha+k-1}(x)$ k=1,...,N for positive real $\alpha$, x. Uses internal double precision arithmetic.

**BESY** Sequence of Bessel functions $Y_{\alpha+k-1}(x)$, k=1,n for positive x and non-negative $\alpha$. Uses internal double precision arithmetic.

*COULOMB Software Library*

**COULFG** A subroutine to compute the regular and irregular Coulomb wave functions F and G, and their derivatives dF/dx and dG/dx, for real values of the Sommerfeld parameter eta and the angular momentum L. A single call to COULFG computes the functions for fixed positive x, fixed real eta, and a range of L-values in integer steps.

*IMSL Subprogram Library*

**MMBSJR** Sequence of Bessel functions $J_r(x)$, for r real and positive.

**MMBSYN** Sequence of Bessel functions $Y_r(x)$, for r real and positive.

*IMSL SFUN/LIBRARY Subprogram Library*

**BSJS** Sequence of Bessel functions $J_r(x)$, for r real and positive.

**BSYS** Sequence of Bessel functions $Y_r(x)$, for real nonnegative r and positive x.

*NMS Subprogram Library*

**BESJ** Sequence of Bessel functions $J_{\alpha+k-1}(x)$, k=1,...,n for non-negative $\alpha$ and x.

*Scientific Desk PC Subprogram Library*

**C10A3J** Bessel function $J_{n+a}(x)$, x and $n + a$ non-negative.

**C10AHD** Complex Bessel function $H_1$, $H_2$ (real order).

**C10AJD** Complex Bessel function J (real order).

**C10AYD** Complex Bessel function Y (real order).

## C10a4 : Bessel functions J, Y, and H of complex argument and real order

*Collected Algorithms of the ACM*

**A644** A portable Fortran package for Bessel functions of a complex argument and nonnegative order. In particular, the Bessel functions $H_1$, $H_2$, I, J, K, and Y, as well as the Airy functions Ai, Bi, and their derivatives are provided in both single and double precision. Exponential scaling and sequence generation are optional. (See D.E. Amos, ACM TOMS 12 (1986) pp. 265-273.)

## C10b : Bessel functions I, K

*BESPAK Software Library*

**BESPAK** Subroutines for Bessel and modified Bessel functions of complex argument and real (integer or fractional) order. BESPAK consists of four main subroutines, one for each Bessel function, and four subsidiary routines.

*Collected Algorithms of the ACM*

**A571** BESRAT, VKAPPA, SPHERR, CAPPA3: Fortran functions providing statistics for von Mises's and Fisher's distributions of directions (the ratio of modified Bessel functions of the first kind). (See G.W. Hill, ACM TOMS 7 (1981) pp. 233-238.)

## C10b1 : Bessel functions I and K of real argument and integer order

*Collected Algorithms of the ACM*

**A518** VMISES: a Fortran function which computes the left tail area of the Von Mises distribution, which is equal to the incomplete modified Bessel function of the first kind and zero-th order (I0). (see G.W. Hill, ACM TOMS 3 (1977)pp. 279-284.)

*CMLIB Library (FNLIB Sublibrary)*

**BESI0** Modified Bessel function $I_0(x)$.

**BESI0E** Exponentially scaled modified Bessel function $\exp(-|x|)I_0(x)$.

**BESI1** Modified Bessel function $I_1(x)$.

**BESI1E** Exponentially scaled modified Bessel function $\exp(-|x|)I_1(x)$.

**BESK0** Modified Bessel function $K_0(x)$.

**BESK0E** Exponentially scaled modified Bessel function $\exp(x)K_0(x)$.

**BESK1** Modified Bessel function $K_1(x)$.

**BESK1E** Exponentially scaled modified Bessel function $\exp(x)K_1(x)$.

*IMSL Subprogram Library*

**MMBSI0** Modified Bessel function $I_0(x)$.

**MMBSI1** Modified Bessel function $I_1(x)$.

**MMBSIN** Sequence of modified Bessel functions $I_n(x)$.

**MMBSK0** Modified Bessel function $K_0(x)$.

**MMBSK1** Modified Bessel function $K_1(x)$.

*IMSL SFUN/LIBRARY Subprogram Library*

**BSI0** Modified Bessel function $I_0(x)$.
**BSI0E** Exponentially scaled modified Bessel function $\exp(-x)I_0(x)$.
**BSI1** Modified Bessel function $I_1(x)$.
**BSI1E** Exponentially scaled modified Bessel function $\exp(-x)I_1(x)$.
**BSINS** Sequence of modified Bessel functions $I_n(x)$.
**BSK0** Bessel function $K_0(x)$.
**BSK0E** Exponentially scaled modified Bessel function $\exp(x)K_0(x)$.
**BSK1** Bessel function $K_1(x)$.
**BSK1E** Exponentially scaled modified Bessel function $\exp(x)K_1(x)$.

*NAG Subprogram Library*

**S18ACF** Modified Bessel function $K_0(x)$.
**S18ADF** Modified Bessel function $K_1(x)$.
**S18AEF** Modified Bessel function $I_0(x)$.
**S18AFF** Modified Bessel function $I_1(x)$.
**S18CCF** Exponentially scaled modified Bessel function $\exp(x)K_0(x)$.
**S18CDF** Exponentially scaled modified Bessel function $\exp(x)K_1(x)$.
**S18CEF** Exponentially scaled modified Bessel function $\exp(-|x|)I_0(x)$.
**S18CFF** Exponentially scaled modified Bessel function $\exp(-|x|)I_1(x)$.

*NMS Subprogram Library*

**BESI0** Modified Bessel function $I_0(x)$.

*PORT Subprogram Library*

**BESRI** Sequence of modified Bessel functions $I_n(x)$.

*Scientific Desk PC Subprogram Library*

**C10BI0** Bessel function $I_0(x)$.
**C10BI1** Bessel function $I_1(x)$.
**C10BK0** Bessel function $K_0(x)$.
**C10BK1** Bessel function $K_1(x)$.

*XMLIBV Cyber 205 Subprogram Library*

**VBESI0** Modified Bessel function $I_0(x)$ for a vector of real arguments.
**VBESI1** Modified Bessel function $I_1(x)$ for a vector of real arguments.
**VBESK0** Modified Bessel function $K_0(x)$ for a vector of real arguments.
**VBESK1** Modified Bessel function $K_1(x)$ for a vector of real arguments.

## C10b2 : Bessel functions I and K of complex argument and integer order

*IMSL Subprogram Library*

**MMBZIN** Sequence of modified Bessel functions $I_n(z)$.

*IMSL SFUN/LIBRARY Subprogram Library*

**CBINS** Sequence of modified Bessel functions $I_n(z)$.

*PORT Subprogram Library*

**BESCI** Sequence of modified Bessel functions $I_n(z)$.

## C10b3 : Bessel functions I and K of real argument and real order

*Collected Algorithms of the ACM*

**A511** IBESS and JBESS: CDC 6600 Fortran subroutines for Bessel functions $I_v(x)$ and $J_v(x)$, for real $x \geq 0$, and real $v \geq 0$. (See D.E. Amos, S.L. Daniel, and M.K. Weston, ACM TOMS 3 (1977) pp. 93-95.)

**A597** RIBESL: a Fortran subprogram for calculating sequences of modified Bessel functions of the first kind (real argument and real order). Optionally, the result may be exponentially scaled. (See W.J. Cody, ACM TOMS 9 (1983) pp. 242-245.)

*CMLIB Library (FNLIB Sublibrary)*

**BESKES** Sequence of exponentially scaled Bessel functions $\exp(x)K_r(x)$, for r real.

**BESKS** Sequence of Bessel functions $K_r(x)$, for $r$ real.

*IMSL Subprogram Library*

**MMBSIR** Sequence of modified Bessel functions $I_r(x)$ or $\exp(-x)I_r(x)$, for $r$ real.

**MMBSKR** Sequence of modified Bessel functions $K_r(x)$ or $\exp(x)K_r(x)$, for $r$ real.

*IMSL SFUN/LIBRARY Subprogram Library*

**BSIES** Sequence of exponentially scaled modified Bessel functions $\exp(-x)I_r(x)$, for $r$ real and x positive.

**BSIS** Sequence of modified Bessel functions $I_r(x)$, for $r$ nonnegative real and x positive.

**BSKES** Sequence of exponentially scaled modified Bessel functions $\exp(x)K_r(x)$, for real $r$.

**BSKS** Sequence of modified Bessel functions $K_r(x)$, for real $r$.

*Scientific Desk PC Subprogram Library*

**C10B3** Bessel functions $I_{n+\alpha}(x)$ for non-negative x and order $n+\alpha$.

**C10B3K** Modified Bessel functions of the second kind, $K_{n+\alpha}(x)$, for non-negative x and order $n+\alpha$.

## C10b4 : Bessel functions I and K of complex argument and real order

*Collected Algorithms of the ACM*

**A644** A portable Fortran package for Bessel functions of a complex argument and nonnegative order. In particular, the Bessel functions $H_1$, $H_2$, I, J, K, and Y, as well as the Airy functions Ai, Bi, and their derivatives are provided in both single and double precision. Exponential scaling and sequence generation are optional. (See D.E. Amos, ACM TOMS 12 (1986) pp. 265-273.)

*Scientific Desk PC Subprogram Library*

**C10BID** Complex Bessel function I (real order).

**C10BKD** Complex Bessel function K (real order).

## C10c : Kelvin functions

*IMSL Subprogram Library*

**MMKEL0** Kelvin functions $ber_0(x)$, $bei_0(x)$, $ker_0(x)$, $kei_0(x)$.

**MMKEL1** Kelvin functions $ber_1(x)$, $bei_1(x)$, $ker_1(x)$, $kei_1(x)$.

**MMKELD** Derivatives of the Kelvin functions $ber_0(x)$, $bei_0(x)$, $ker_0(x)$, and $kei_0(x)$.

*IMSL SFUN/LIBRARY Subprogram Library*

**AKEI0** Kelvin function $kei_0(x)$.

**AKEI1** Kelvin function $kei_1(x)$.

**AKEIP0** Derivative of the Kelvin function $kei_0(x)$.

**AKER0** Kelvin function $ker_0(x)$.

**AKER1** Kelvin function $ker_1(x)$.

**AKERP0** Derivative of the Kelvin function $ker_0(x)$.

**BEI0** Kelvin function $bei_0(x)$.

**BEI1** Kelvin function $bei_1(x)$.

**BEIP0** Derivative of the Kelvin function $bei_0(x)$.

**BER0** Kelvin function $ber_0(x)$.

**BER1** Kelvin function $ber_1(x)$.

**BERP0** Derivative of the Kelvin function $ber_0(x)$.

*NAG Subprogram Library*

**S19AAF** Kelvin function ber(x).

**S19ABF** Kelvin function bei(x).

**S19ACF** Kelvin function ker(x).

**S19ADF** Kelvin function kei(x).

## C10d : Airy and Scorer functions

*Collected Algorithms of the ACM*

**A498** AIRY: a subprogram to evaluate the Airy functions Ai(z), Bi(z) and their derivatives for all real values of z within computer capability. Based on Chebyshev series approximations. (See P.J. Prince, ACM TOMS 1 (1975) pp. 372-379.)

**A644** A portable Fortran package for Bessel functions of a complex argument and nonnegative order. In particular, the Bessel functions $H_1$, $H_2$, I, J, K, and Y, as well as the Airy functions Ai, Bi, and their derivatives are provided in both single and double precision. Exponential scaling and sequence generation are optional. (See D.E. Amos, ACM TOMS 12 (1986) pp. 265-273.)

*CMLIB Library (FNLIB Sublibrary)*

**AI** Airy function Ai(x).

**AIE** Exponentially scaled Airy function exp(x)Ai(x).

**BI** Airy function Bi(x).

**BIE** Exponentially scaled Airy function exp(-x)Bi(x).

*IMSL SFUN/LIBRARY Subprogram Library*

| | |
|---|---|
| **AI** | Airy function Ai(x). |
| **AID** | Derivative of the Airy function Ai(x). |
| **AIDE** | Exponentially scaled derivative of the Airy function exp(x)dAi(x)/dx. |
| **AIE** | Exponentially scaled Airy function exp(x)Ai(x). |
| **BI** | Airy function Bi(x). |
| **BID** | Derivative of the Airy function Bi(x). |
| **BIDE** | Exponentially scaled derivative of the Airy function exp(-x)dBi(x)/dx. |
| **BIE** | Exponentially scaled Airy function exp(-x)Bi(x). |

*NAG Subprogram Library*

| | |
|---|---|
| **S17AGF** | Airy function Ai(x). |
| **S17AHF** | Airy function Bi(x). |
| **S17AJF** | Derivative of the Airy function Ai(x). |
| **S17AKF** | Derivative of the Airy function Bi(x). |

*Scientific Desk PC Subprogram Library*

| | |
|---|---|
| **C10DA** | Airy function Ai(x). |
| **C10DAD** | Complex Airy function Ai(z) . |
| **C10DAE** | Exponentially scaled Airy function Ai(x). |
| **C10DAM** | Airy modulus and phase. |
| **C10DB** | Airy function Bi(x). |
| **C10DBD** | Complex Airy function Bi(z). |
| **C10DBE** | Exponentially scaled Airy function Bi(x). |

## C10e : Struve, Anger, and Weber functions

*CMLIB Library (AMOSLIB Sublibrary)*

**CJYHBS** Bessel and Struve functions $J_0(z)$, $J_1(z)$, $Y_0(z)$, $Y_1(z)$, $H_0(z)$, $H_1(z)$.

## C10f : Integrals of Bessel functions

*Collected Algorithms of the ACM*

**A588** HANKEL: a Fortran subroutine for the fast evaluation of complex Hankel transforms of orders 0 and 1 using related and lagged convolutions. (See W.L. Anderson, ACM TOMS 8 (1982) pp. 369-370.)

**A609** BSKIN: a Fortran subprogram which computes the Bickley functions Ki(n,x), repeated integrals of the $K_0$ Bessel function, for non-negative integers n and reals x. A double precision version is included. (See D.E. Amos, ACM TOMS 9 (1983) pp. 480-493.)

## C11 : Confluent hypergeometric functions

*CMLIB Library (FNLIB Sublibrary)*

**CHU** Confluent hypergeometric function, U(a,b,x).

## C12 : Coulomb wave functions

*COULOMB Software Library*

**COULFG** A subroutine to compute the regular and irregular Coulomb wave functions F and G, and their derivatives dF/dx and dG/dx, for real values of the Sommerfeld parameter eta and the angular momentum L. A single call to COULFG computes the functions for fixed positive x, fixed real eta, and a range of L-values in integer steps.

## C14 : Elliptic integrals

*Collected Algorithms of the ACM*

**A577** RC, RF, RD, RJ: Fortran functions for symmetric incomplete elliptic integrals of the first, second, and third kinds. (See B.C. Carlson and E.M. Notis, ACM TOMS 7 (1981) pp. 398-403.)

*CMLIB Library (FCNPAK Sublibrary)*

**RC** Carlson's incomplete elliptic integral RC(x,y).

**RD** Carlson's incomplete elliptic integral RD(x,y,z).

**RF** Carlson's incomplete elliptic integral RF(x,y,z).

**RJ** Carlson's incomplete elliptic integral RJ(x,y,z,p).

*IMSL Subprogram Library*

**MMDELE** Complete elliptic integral E(m), see [AMS55 (17.3.3)].

**MMDELK** Complete elliptic integral K(m), see [AMS55 (17.3.1)].

**MMLIND** Carlson's incomplete elliptic integral RD(x,y,z).

**MMLINF** Carlson's incomplete elliptic integral RF(x,y,z).

**MMLINJ** Carlson's incomplete elliptic integral RJ(x,y,z,p).

*IMSL SFUN/LIBRARY Subprogram Library*

**ELE** Complete elliptic integral E(m), see [AMS55 (17.3.3)].

**ELK** Complete elliptic integral K(m), see [AMS55 (17.3.1)].

**ELRC** Carlson's incomplete elliptic integral RC(x,y).

**ELRD** Carlson's incomplete elliptic integral RD(x,y,z).

**ELRF** Carlson's incomplete elliptic integral RF(x,y,z).

**ELRJ** Carlson's incomplete elliptic integral RJ(x,y,z,p).

*NAG Subprogram Library*

**S21BAF** Carlson's incomplete elliptic integral RC(x,y).

**S21BBF** Carlson's incomplete elliptic integral RF(x,y,z).

**S21BCF** Carlson's incomplete elliptic integral RD(x,y,z).

**S21BDF** Carlson's incomplete elliptic integral RJ(x,y,z,p).

*Scientific Desk PC Subprogram Library*

**C14RC** Carlson's incomplete elliptic integral RC(x,y).

**C14RD** Carlson's incomplete elliptic integral RD(x,y,z).

**C14RF** Carlson's incomplete elliptic integral RF(x,y,z).

**C14RJ** Carlson's incomplete elliptic integral RJ(x,y,z,p).

## C15 : Weierstrass elliptic functions

*Collected Algorithms of the ACM*

**A549** Fortran functions for evaluating Weierstrass's P-functions in the equiharmonic and lemniscatic cases. (See U. Eckhardt, ACM TOMS 6 (1980) pp. 112-120.)

*IMSL Subprogram Library*

**MMWPL** Weierstrass P-function with primitive half-periods 1/2 [AMS55 (18.1)]. The corresponding invariants are $g_2$=4L, where L=2.62205... is the Lemniscate Constant [AMS55 (18.14.7)] and $g_3$=0.

**MMWPL1** First derivative of MMWPL.

**MMWPQ** Weierstrass P-function in the equianharmonic case for complex argument with unit period parallelogram.

**MMWPQ1** First derivative of MMWPQ.

*IMSL SFUN/LIBRARY Subprogram Library*

**CWPL** Weierstrass P-function with primitive half-periods 1/2 [AMS55 (18.1)]. The corresponding invariants are $g_2$=4L, where L=2.62205... is the Lemniscate Constant [AMS55 (18.14.7)] and $g_3$=0.

**CWPLD** First derivative of CWPL.

**CWPQ** Weierstrass P-function in the equianharmonic case for complex argument with unit period parallelogram.

**CWPQD** First derivative of CWPQ.

## C17 : Mathieu functions

*Collected Algorithms of the ACM*

**A537** CHARMA: a Fortran subprogram for calculating the characteristic values of Mathieu's differential equation for odd or even solutions. (See W.R. Leeb, ACM TOMS 5 (1979) pp. 112-117.)

## C19 : Other special functions

*IMSL MATH/LIBRARY Subprogram Library*

**CONST** Various mathematical and physical constants.

**CUNIT** Convert X in units XUNITS to Y in units YUNITS.

# D: Linear Algebra

The most widely used constructs in mathematical computation are, undoubtedly, matrices and vectors. Consequently, it is not surprising to find a rich selection of software designed to perform operations such as computing solutions of linear systems, eigenvalues, determinants, and inverses. While it may seem that there are many routines that perform essentially the same task, one will find that some of these routines are designed to take advantage of special features of a given problem. Thus, to select the most appropriate routine you should determine as many attributes of your matrix as possible. Some of these are given below.

1. The entries of the matrix are either real or complex numbers.

2. The matrix elements may have certain symmetry properties. If a matrix $A$ is real, it is called *symmetric* when its elements satisfy $a_{ji} = a_{ij}$. If the matrix is complex, it is called *Hermitian* when $a_{ji} = \bar{a}_{ij}$.

3. The location and density of the non-zero entries of the matrix are important attributes. A matrix is *dense* if most of the entries are non-zero. It is called *sparse* if there are relatively few non-zero entries. The cross-over point between dense and sparse is difficult to determine in practice.

4. *Banded* matrices are sparse matrices wherein all the non-zero entries are concentrated on a band near the main diagonal. Such matrices commonly arise when differential equations are discretized. *Tridiagonal* matrices are those band matrices in which the band consists only of the main diagonal and the closest lower and upper diagonals.

5. Hermitian and symmetric matrices may also be positive definite. A real matrix $A$ is *positive definite* if $x^T Ax > 0$ for all nonzero vectors $x$. This condition is often difficult to determine in practice without some independent knowledge of the matrix. A real symmetric matrix, for example, is positive definite if and only if all its eigenvalues are positive.

## References

[Duf77] I. S. Duff. A survey of sparse matrix research. *Proceedings of the IEEE*, 64(4):500–535, 1977.

[GVL83] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 1983.

[ND77] B. Noble and J. W. Daniel. *Applied Linear Algebra.* Prentice-Hall, Englewood Cliffs, NJ, second edition, 1977.

[Rei77] J. K. Reid. Sparse matrices. In D. Jacobs, editor, *The State of the Art in Numerical Analysis*, pages 85–146. Academic Press, New York, 1977.

[Ste73] G. W. Stewart. *Introduction to Matrix Computations.* Academic Press, New York, 1973.

| D : | Linear Algebra |
|---|---|

*MATLAB Interactive Program Library*

**MATLAB** An interactive system for defining and manipulating matrices. It includes solving linear systems, linear least squares, eigenvalue and eigenvector calculation, QR decomposition, singular value decomposition and inverses.

*MINITAB Interactive System*

**MINITAB** Vector and matrix commands include COPY (vectors, vectors to matrices, and conversely), DIAGONAL (create a diagonal matrix or extract the diagonal of a matrix), TRANSPOSE, INVERSE, and EIGEN (calculate eigenvalues and eigenvectors for a symmetric matrix).

# D1: Elementary vector and matrix operations

The subprograms in this class provide automatic computation of common elementary vector and matrix operations, such as the dot (inner) product of two vectors or the product of two matrices. These are provided to reduce the time and effort required to code problems. In addition, they can also help in making a program more portable while at the same time achieving high efficiency on a wide variety of machines. For example, the Basic Linear Algebra Subprograms (BLAS) are widely distributed and are often implemented in machine language to increase their efficiency. Since these operations are often found in the innermost loops of large computations the resulting savings can often be significant. Most large libraries contain implementations of the BLAS.

## References

[DDH88] J. J. Dongarra, J. Du Croz, S. Hammarling, and Richard J. Hanson. An extended set of FORTRAN basic linear algebra subprograms. *ACM Transactions on Mathematical Software*, 14(1):1–17, 1988.

[LHK79] C. L. Lawson, R. J. Hanson, D. R. Kincaid, and Krogh F. T. Basic Linear Algebra Subprograms for Fortran usage. *ACM Transactions on Mathematical Software*, 5(3):308–323, 1979.

| D1 : | Elementary vector and matrix operations |
|---|---|

*Collected Algorithms of the ACM*

**A656** Fortran model implementation and test programs for Level 2 BLAS. (See J. J. Dongarra, J. du Croz, S. Hammarling, and R. J. Hanson, ACM TOMS 14 (1988) pp. 18-32.)

| D1a : | Elementary vector operations |
|---|---|

*Collected Algorithms of the ACM*

**A539** BLAS (Basic Linear Algebra Subprograms): a package of 38 Fortran subprograms, to perform basic operations of numerical linear algebra, including dot product, Givens transformations, vector copy and swap, vector norm, vector scaling, and determination of the index of the vector component of the largest magnitude. Includes test programs and assembly language versions for the IBM 360, CDC 6000, and Univac 1100. (See C.L. Lawson et al., ACM TOMS 5 (1979) pp. 324-325.)

**A653** PC-BLAS: Assembly language version of the Basic Linear Algebra Subprograms for Fortran usage with the INTEL 8087/80287 numeric data processor. (See R. J. Hanson and F. T. Krogh, ACM TOMS 13 (1987) pp. 311-317.)

**A663** CWI BLAS: Basic Linear Algebra Subprograms in Fortran 200 for the Cyber 205. (See M. Louter-Nool, ACM TOMS 14 (1988) pp. 177-195.)

## D1a1 : Set a vector to a constant

*DATAPAC Subprogram Library*

**DEFINE** Defines a vector of constants by setting all of the elements in the single precision vector X equal to XNEW.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CSET** Set the components of a vector to a scalar, all complex.

**ISET** Set the components of a vector to a scalar, all integer.

**SSET** Set the components of a vector to a scalar, all single precision.

*NAG Subprogram Library*

**F01CQF** Sets the elements of a vector to zero.

**F06DBF** Broadcast scalar into integer vector.

**F06FBF** Broadcast scalar into real vector.

**F06HBF** Broadcast scalar into complex vector.

*PORT Subprogram Library*

**SETC** Set a specified number of values in a complex array equal to a constant.

**SETI** Set a specified number of values in an integer array equal to a constant.

**SETL** Set a specified number of values in a logical array equal to a constant.

**SETR** Set a specified number of values in a real array equal to a constant.

## D1a2 : Minimum and maximum components of a vector

*CMLIB Library (BLAS Sublibrary)*

**ICAMAX** Find smallest index of maximum magnitude component of a complex vector.

**ISAMAX** Find smallest index of maximum magnitude component of a single precision vector.

*CMLIB Library (XBLAS Sublibrary)*

**ISAMIN** Find the smallest index of the minimum magnitude component of a real vector.

**ISMAX** Find the smallest index of the maximum component of a real vector.

**ISMIN** Find the smallest index of the minimum component of a real vector.

*DATAPAC Subprogram Library*

**MAX** Computes the sample maximum of the data in the input vector X.

**MIN** Computes the sample minimum of the data in the input vector X.

*IMSL Subprogram Library*

**USMNMX** Determination of the minimum and maximum values of a vector.

**VABMXF** Maximum absolute value of the elements of a vector or a subset of the elements of a vector.

**VABMXS** Maximum absolute value of the elements of a row or column of a matrix stored in symmetric storage mode.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**ICAMAX** Find the smallest index of the component of a complex vector having maximum magnitude.

**ICAMIN** Find the smallest index of the component of a complex vector having minimum magnitude.

**IIMAX** Find the smallest index of the maximum component of an integer vector.

**IIMIN** Find the smallest index of the minimum of an integer vector.

**ISAMAX** Find the smallest index of the component of a single-precision vector having maximum absolute value.

**ISAMIN** Find the smallest index of the component of a single-precision vector having minimum absolute value.

**ISMAX** Find the smallest index of the component of a single-precision vector having maximum value.

**ISMIN** Find the smallest index of the component of a single-precision vector having minimum value.

*NAG Subprogram Library*

**F06FLF** Elements of real vector with largest and smallest absolute value.

**F06JLF** Index, real vector element with largest absolute value.

**F06JMF** Index, complex vector element with largest absolute value.

**F06KLF** Last non-negligible element of real vector.

*PORT Subprogram Library*

**EXTRMI** Finds extremal points of an integer function defined on a mesh.

**EXTRMR** Finds extremal points of a real function defined on a mesh.

*Scientific Desk PC Subprogram Library*

**D1A3C** Finds the index of the real array element having maximum absolute value.

## D1a3a : $L_1$ (sum of magnitudes) vector norm

*CMLIB Library (BLAS Sublibrary)*

**SASUM** Compute single precision sum of absolute values of components of vector.

**SCASUM** Compute complex sum of absolute values of components of vector.

*IMSL Subprogram Library*

**VABSMF** Sum of the absolute values of the elements of a vector or a subset of a vector.

**VABSMS** Sum of the absolute values of the elements of a row (or column) of a matrix stored in symmetric storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

**DISL1** Compute the 1-norm distance between two points.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**SASUM** Sum the absolute values of the components of a single precision vector.

**SCASUM** Sum the absolute values of the real part together with the absolute values of the imaginary part of the components of a complex vector.

*NAG Subprogram Library*

**F06EKF** Sum the absolute values of real vector elements.

**F06JKF** Sum the absolute values of complex vector elements.

*Scientific Desk PC Subprogram Library*

**D1A3A** Compute single precision sum of absolute values of components of vector.

## D1a3b : $L_2$ (Euclidean) vector norm

*CMLIB Library (BLAS Sublibrary)*

**SCNRM2** Compute the Euclidean length or $L_2$ norm of a complex vector.

**SNRM2** Compute the Euclidean length or $L_2$ norm of a single precision vector, without underflow or overflow.

*IMSL MATH/LIBRARY Subprogram Library*

**DISL2** Compute the Euclidean (2-norm) distance between two points.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**SCNRM2** Compute the Euclidean norm of a complex vector.

**SNRM2** Compute the Euclidean length or $L_2$ norm of a single precision vector, without underflow or overflow.

*NAG Subprogram Library*

**F05ABF** Approximate 2-norm of a vector.

**F06BMF** Compute Euclidean norm from scaled form.

**F06BNF** Compute square root of $(a^2 + b^2)$, real a and b.

**F06EJF** Compute Euclidean norm of real vector.

**F06FJF** Update Euclidean norm of real vector in scaled form.

**F06FKF** Compute weighted Euclidean norm of real vector.

**F06JJF** Compute Euclidean norm of complex vector.

**F06KJF** Update Euclidean norm of complex vector in scaled form.

*PORT Subprogram Library*

**SNRM2** Finds the length (Euclidean norm) of a vector, without underflow or overflow.

*Scientific Desk PC Subprogram Library*

**D1A3B** Forms the square root of the sum of the squares of an array of real numbers, (Euclidean, $L_2$ norm).

## D1a3c : $L_\infty$ (maximum magnitude) vector norm

*CMLIB Library (BLAS Sublibrary)*

**ICAMAX** Find smallest index of maximum magnitude component of a complex vector.

**ISAMAX** Find smallest index of maximum magnitude component of a single precision vector.

*IMSL Subprogram Library*

**VABMXF** Maximum absolute value of the elements of a vector or a subset of the elements of a vector.

**VABMXS** Maximum absolute value of the elements of a row or column of a matrix stored in symmetric storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

**DISLI** Compute the infinity norm distance between two points.

**ISAMAX** Find the smallest index of the component of a single-precision vector having maximum absolute value.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**ICAMAX** Find the smallest index of the component of a complex vector having maximum magnitude.

*NAG Subprogram Library*

**F06FLF** Elements of real vector with largest and smallest absolute value.

**F06JLF** Index, real vector element with largest absolute value.

**F06JMF** Index, complex vector element with largest absolute value.

*Scientific Desk PC Subprogram Library*

**D1A3C** Finds the index of the real array element having maximum absolute value.

## D1a4 : Vector dot product (inner product)

*CMLIB Library (BLAS Sublibrary)*

**CDOTC** Compute complex dot product using conjugated vector components.

**CDOTU** Compute complex dot product using unconjugated vector components.

**DSDOT** Compute single precision dot product x·y using double precision accumulation.

**SDOT** Compute single precision dot product.

**SDSDOT** Compute the sum of a single-precision scalar and a single-precision dot product, a + x·y, using a double-precision accumulator.

*CMLIB Library (XBLAS Sublibrary)*

**CDCDOT** Computes complex precision dot product and adds a scalar. Uses double precision accumulation.

**DCDOT** Computes a complex precision dot product using double precision accumulation.

*IMSL Subprogram Library*

**VIPRFF** Vector inner product of two vectors or subsets of two vectors.

**VIPRSS** Vector inner product of two vectors each of which is part of some matrix stored in symmetric mode.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CDOTC** Compute the complex conjugate dot product, conjg(x)·y.

**CDOTU** Compute the complex dot product x·y.

**CZCDOT** Compute the sum of a complex scalar plus a complex conjugate dot product, a + conjg(x)·y, using a double-precision accumulator.

**CZDOTA** Compute the sum of a complex scalar, a complex dot product and the double-complex accumulator, which is set to the result ACC = ACC + a + x·y.

**CZDOTC** Compute the complex conjugate dot product, conjg(x)·y, using a double-precision accumulator.

**CZDOTI** Compute the sum of a complex scalar plus a complex dot product using a double-complex accumulator, which is set to the result ACC = a + x·y.

**CZDOTU** Compute the complex dot product x·y using a double-precision accumulator.

**CZUDOT** Compute the sum of a complex scalar plus a complex dot product, a + x·y, using a double-precision accumulator.

**DSDOT** Compute single precision dot product x·y using double precision accumulation.

**SDDOTA** Compute the sum of a single-precision scalar, a single-precision dot product and the double-precision accumulator, which is set to the result ACC = ACC + a + x·y.

**SDDOTI** Compute the sum of a single-precision scalar plus a single-precision dot product using a double-precision accumulator, which is set to the result ACC = a + x·y.

**SDOT** Compute the single-precision dot product x·y.

**SDSDOT** Compute the sum of a single-precision scalar and a single-precision dot product, a + x·y, using a double-precision accumulator.

*NAG Subprogram Library*

**F01DAF** Returns the sum of an initial value and a scalar product, using basic precision arithmetic.

**F01DBF** Returns the sum of an initial value and a scalar product, using additional precision arithmetic.

**F01DCF** Computes the value of a complex scalar product and subtracts it from a complex initial value, using basic precision arithmetic.

**F01DDF** Computes the value of a complex scalar product and subtracts it from a complex initial value, using additional precision arithmetic.

**F01DEF** Returns the value of the scalar product of two arrays of length N, using basic precision arithmetic.

**F06EAF** Dot product of two real vectors.

**F06GAF** Dot product of two complex vectors, unconjugated.

**F06GBF** Dot product of two complex vectors, conjugated.

**X03AAF** Calculates the value of a scalar product using basic or additional precision and adds it to a basic or additional precision initial value.

**X03ABF** Calculates the value of a complex scalar product using basic or additional precision and adds it to a complex initial value.

*Scientific Desk PC Subprogram Library*

**D1A41** Forms the scalar (dot) product of two real arrays using double precision accumulation and multiplication.

## D1a5 : Copy or exchange (swap) vectors

*CMLIB Library (BLAS Sublibrary)*

**CCOPY** Copy a vector X to a vector Y, both complex.

**CSWAP** Interchange vectors X and Y, both complex.

**SCOPY** Copy a vector X to a vector Y, both single precision.

**SSWAP** Interchange vectors X and Y, both single precision.

*CMLIB Library (XBLAS Sublibrary)*

**SCOPYM** Copies negative of array SX into array SY, with corresponding increments INCX and INCY.

*DATAPAC Subprogram Library*

**COPY** Copies the contents of the vector X into vector Y.

**MOVE** Copies M elements of the vector X (starting with position IX1) into the vector Y (starting with position IY1).

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CCOPY** Copy a vector X to a vector Y, both complex.

**CSWAP** Interchange vectors X and Y, both complex.

**ICOPY** Copy a vector X to a vector Y, both integer.

**ISWAP** Interchange vectors X and Y, both integer.

**SCOPY** Copy a vector X to a vector Y, both single precision.

**SSWAP** Interchange vectors X and Y, both single precision.

*MAGEV Cyber 205 Subprogram Library*

**FSTOFV** The full-word scalar A(IA) is broadcast into the full-word target-vector starting with B(IB).

**FVTOFV** The full-word vector starting with A(IA) is copied into the full-word target-vector starting with B(IB).

*NAG Subprogram Library*

**F01CNF** Copies a vector of length M into a row of a matrix.

**F01CPF** Copies the contents of a vector into a second vector.

**F06DFF** Copy integer vector.

**F06EFF** Copy real vector.

**F06EGF** Swap two real vectors.

**F06GFF** Copy complex vector.

**F06GGF** Swap two complex vectors.

**F06KFF** Copy real vector to complex vector.

*PORT Subprogram Library*

**MOVEBC** Move a complex vector using backward DO loop.

**MOVEBD** Move a double precision vector using backward DO loop.

**MOVEBI** Move an integer vector using backward DO loop.

**MOVEBL** Move a logical vector using backward DO loop.

**MOVEBR** Move a real vector using backward DO loop.

**MOVEFC** Move a complex vector using forward DO loops.

**MOVEFD** Move a double precision vector using forward DO loop.

**MOVEFI** Move an integer vector using forward DO loop.

**MOVEFL** Move a logical vector using forward DO loop.

**MOVEFR** Move a real vector using forward DO loop.

*Scientific Desk PC Subprogram Library*

**D1A5** Interchange two real vectors X and Y.

**D1A5C** Copy a real vector X to a real vector Y.

## D1a6 : Multiplication of a vector by a scalar

*CMLIB Library (BLAS Sublibrary)*

**CSCAL** Multiply a vector by a scalar, y = ay, both complex.
**CSSCAL** Multiply a complex vector by a single-precision scalar, y = ay.
**SSCAL** Multiply a vector by a scalar, y = ay, both single precision.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CSCAL** Multiply a vector by a scalar, y = ay, both complex.
**CSSCAL** Multiply a complex vector by a single-precision scalar, y = ay.
**CSVCAL** Multiply a complex vector by a single-precision scalar and store the result in another complex vector, y = ax.
**CVCAL** Multiply a vector by a scalar and store the result in another vector, y = ax, all complex.
**SSCAL** Multiply a vector by a scalar, y = ay, both single precision.
**SVCAL** Multiply a vector by a scalar and store the result in another vector, y = ax, all single precision.

*MAGEV Cyber 205 Subprogram Library*

**FSMPYFV** The full-word scalar A(IA) multiplies each array-element in the full-word vector B, a broadcast multiply. The result is stored into the full-word vector C.
**FVMPYFV** To perform vector multiplication. Corresponding elements in the full-word vectors, A and B, are multiplied and stored into the corresponding element in the full-word vector, C.

*NAG Subprogram Library*

**F06EDF** Multiply real vector by scalar.
**F06FDF** Multiply real vector by scalar, preserving input vector.
**F06FGF** Negate real vector.
**F06GDF** Multiply complex vector by complex scalar.
**F06HDF** Multiply complex vector by complex scalar, preserving input vector.
**F06HGF** Negate complex vector.
**F06JDF** Multiply complex vector by real scalar.
**F06KDF** Multiply complex vector by real scalar, preserving input vector.

*Scientific Desk PC Subprogram Library*

**D1A6** Compute a real constant times a real vector.

## D1a7 : Vector triad ($\alpha x + y$ for vectors $x$, $y$ and scalar $\alpha$)

*CMLIB Library (BLAS Sublibrary)*

**CAXPY** Compute a scalar times a vector plus a vector, y = ax + y, all complex.
**SAXPY** Compute a constant times a vector plus a vector, all single precision.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CAXPY** Compute a scalar times a vector plus a vector, y = ax + y, all complex.
**SAXPY** Compute the scalar times a vector plus a vector, y = ax + y, all single precision.

*NAG Subprogram Library*

**F06ECF** Add scalar times real vector to real vector.
**F06GCF** Add scalar times complex vector to complex vector.

*Scientific Desk PC Subprogram Library*

**D1A7** Compute a real constant times a real vector plus a real vector.

## D1a8 : Elementary rotation (Givens transformation) of a vector

*CMLIB Library (BLAS Sublibrary)*

**SROT** Apply Givens plane rotation to a single precision vector.

**SROTM** Apply modified Givens plane rotation to single precision vector.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CSROT** Apply a complex Givens plane rotation.

**CSROTM** Apply a complex modified Givens plane rotation.

**SROT** Apply Givens plane rotation to a single precision vector.

**SROTG** Construct Givens plane rotation of a single precision matrix.

**SROTM** Apply modified Givens plane rotation to single precision vector.

**SROTMG** Construct modified Givens plane rotation of a single precision matrix.

*NAG Subprogram Library*

**F06AAF** Generate real plane rotation.

**F06BAF** Generate real plane rotation, storing tangent.

**F06BEF** Generate real Jacobi plane rotation.

**F06BHF** Apply real similarity rotation to 2-by-2 symmetric matrix.

**F06CAF** Generate complex plane rotation, storing tangent, real cosine.

**F06CBF** Generate complex plane rotation, storing tangent, real sine.

**F06CHF** Apply complex similarity rotation to 2-by-2 Hermitian matrix.

**F06EPF** Apply real plane rotation.

**F06FPF** Apply real symmetric plane rotation to two vectors.

**F06FQF** Generate sequence of real plane rotations.

**F06HPF** Apply complex plane rotation.

**F06HQF** Generate sequence of complex plane rotations.

**F06KPF** Apply real plane rotation to two complex vectors.

*Scientific Desk PC Subprogram Library*

**D1A8** Applies a plane rotation to a 2-by-N matrix whose first and second row are given in the real arrays X and Y.

**D1A8G** Performs a Givens plane rotation.

## D1a9 : Elementary reflection (Householder transformation) of a vector

*IMSL Subprogram Library*

**VHS12** Real Householder transformation computation and applications.

*IMSL MATH/LIBRARY Subprogram Library*

**SHOUTR** Construct a Householder transformation in single precision.

*IMSL STAT/LIBRARY Subprogram Library*

**SHOUAP** Apply a Householder transformation in single precision.
**SHOUTR** Construct a Householder transformation in single precision.

*NAG Subprogram Library*

**F06FRF** Generate real elementary reflection, NAG style.
**F06FSF** Generate real elementary reflection, LINPACK style.
**F06FTF** Apply real elementary reflection, NAG style.
**F06FUF** Apply real elementary reflection, LINPACK style.
**F06HRF** Generate complex elementary reflection.
**F06HTF** Apply complex elementary reflection.

## D1a10 : Vector convolutions

*IMSL Subprogram Library*

**VCONVO** Vector convolution.

*IMSL MATH/LIBRARY Subprogram Library*

**VCONC** Compute the convolution of two complex vectors.
**VCONR** Compute the convolution of two real vectors.

*NAG Subprogram Library*

**C06EKF** Calculates the circular convolution or correlation of two real vectors of period N. (No extra workspace is required.)
**C06FKF** Calculates the circular convolution or correlation of two real vectors of period N (using a work array for extra speed).

## D1a11 : Other vector operations

*IMSL MATH/LIBRARY Subprogram Library*

**DISL1** Compute the 1-norm distance between two points.
**DISL2** Compute the Euclidean (2-norm) distance between two points.
**DISLI** Compute the infinity norm distance between two points.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**BLINF** Compute the bilinear mode $x^T Ay$.
**CADD** Add a scalar to each component of a vector, $x = x + a$, all complex.
**CSUB** Subtract each component of a vector from a scalar, $x = a - x$, all complex.
**IADD** Add a scalar to each component of a vector, $x = x + a$, all integer.

| | |
|---|---|
| **ISUB** | Subtract each component of a vector from a scalar, x = a − x, all integer. |
| **ISUM** | Sum the values of an integer vector. |
| **SADD** | Add a scalar to each component of a vector, x = x + a, all single precision. |
| **SHPROD** | Compute the Hadamard product of two single precision vectors. |
| **SPRDCT** | Multiply the components of a single precision vector. |
| **SSUB** | Subtract each component of a vector from a scalar, x = a − x, all single precision. |
| **SSUM** | Sum the values of a single precision vector. |
| **SXYZ** | Compute a single precision XYZ product. |

*MAGEV Cyber 205 Subprogram Library*

| | |
|---|---|
| **FSADDFV** | Broadcast add. The full-word scalar A is added to each array-element in the full-word vector B. The result is stored into the full-word vector C. |
| **FSDIVFV** | Broadcast divide. The full-word scalar A(IA) is divided with each array-element in the full-word vector B and the result is stored into the full-word vector C. |
| **FSSUBFV** | Broadcast subtract. Each array-element in the full-word vector B is subtracted from the full-word scalar A(IA). The result is stored into the full-word vector C. |
| **FVADDFV** | Vector addition, C = A + B. A, B and C are full-word vectors and addition is done elementwise. |
| **FVDIVFV** | Vector division, C = A/B. A, B and C are full-word vectors and division is done elementwise. |
| **FVSUBFV** | Vector subtraction. Each element of the full-word vector B is subtracted from the corresponding element in the full-word vector A. The result is stored in the corresponding element of the full-word vector C. |
| **HCMPRS** | Compress out selected half-word elements from a half-word source vector, storing them contiguously in memory. The hardware instruction CPSV is used. |

*NAG Subprogram Library*

| | |
|---|---|
| **F06FAF** | Compute cosine of angle between two real vectors. |
| **F06KLF** | Last non-negligible element of real vector. |

## D1b : Elementary matrix operations

*IMSL MATH/LIBRARY Subprogram Library*

| | |
|---|---|
| **HRRRR** | Compute the Hadamard product of two real matrices. |

*MAGEV Cyber 205 Subprogram Library*

| | |
|---|---|
| **MXCMP** | To compare two matrices A and B by computing the maximum relative error of the columns in B with respect to those in A. Both 2-norms and infinity-norms are computed. |
| **MXEQ** | To determine whether two matrices are identical. |

## D1b1 : Set a matrix to zero or to the identity matrix

*NAG Subprogram Library*

| | |
|---|---|
| **F01CAF** | Sets elements of an m-by-n matrix A to zero. |
| **F01CBF** | Sets the elements A(i,j) to one if i=j and zero otherwise, where $1 \le i \le m$ and $1 \le j \le n$. |

## D1b2 : Matrix norm

*IMSL Subprogram Library*

**VNRMF1** 1-norm of matrices (full storage mode).

**VNRMF2** Euclidean-norm of matrices (full storage mode).

**VNRMFI** Infinity-norm matrices (full storage mode).

**VNRMS1** 1-norm of matrices (symmetric storage mode).

**VNRMS2** Euclidean-norm of matrices (symmetric storage mode).

*IMSL MATH/LIBRARY Subprogram Library*

**NR1CB** Compute the 1-norm of a complex band matrix in band storage mode.

**NR1RB** Compute the 1-norm of a real band matrix in band storage mode.

**NR1RR** Compute the 1-norm of a real matrix.

**NR2RR** Compute the Frobenius norm of a real rectangular matrix.

**NRIRR** Compute the infinity norm of a real matrix.

*Scientific Desk PC Subprogram Library*

**D1B2** Finds the element of a real rectangular matrix with the largest absolute value, the infinity norm.

**D1B21** Finds the 1-norm of a real matrix, the maximum of the sums of the absolute values of the column elements.

**D1B22** Finds the 1-norm of a real symmetric matrix stored in packed form, the maximum of the sums of the absolute values of the column elements.

**D1B2S** Finds the element with maximum absolute value, the infinity norm, of a real symmetric matrix stored in packed form.

## D1b3 : Matrix transpose

*Collected Algorithms of the ACM*

**A513** TRANS: a Fortran subroutine for in-situ matrix transposition which makes use of the cyclic structure of the transposition mapping. This is a revision of Algorithm 380. (See E.G. Cate and D.W. Twigg, ACM TOMS 3 (1977) pp. 104-110.)

*IMSL Subprogram Library*

**VTRAN** Transpose a rectangular matrix.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**TRNRR** Transpose a rectangular matrix.

*MAGEV Cyber 205 Subprogram Library*

**TPMOV** To transpose a rectangular matrix, simultaneously moving it from one location to another. The source and target locations must not overlap.

*NAG Subprogram Library*

**F01CRF** Re-orders the elements of a vector of length mn, containing an m-by-n matrix, A, so that the new vector contains the transpose matrix.

*Scientific Desk PC Subprogram Library*

**D1B3** Transposes a real rectangular matrix.

## D1b4 : Multiplication of a matrix by a vector

*IMSL MATH/LIBRARY Subprogram Library*

**MUCBV** Multiply a complex band matrix in band storage mode by a complex vector.
**MUCRV** Multiply a complex rectangular matrix by a complex vector.
**MURBV** Multiply a real band matrix in band storage mode by a real vector.
**MURRV** Multiply a real rectangular matrix by a vector.

*NAG Subprogram Library*

**F01CSF** Forms the product c = Ab where b is a vector and A is a symmetric matrix whose lower triangle is stored by rows in a one-dimensional array.
**F06HCF** Multiply complex vector by complex diagonal matrix.
**F06KCF** Multiply complex vector by real diagonal matrix.
**F06PAF** Matrix-vector product, real general matrix.
**F06PBF** Matrix-vector product, real general band matrix.
**F06PCF** Matrix-vector product, real symmetric matrix.
**F06PDF** Matrix-vector product, real symmetric band matrix.
**F06PEF** Matrix-vector product, real symmetric packed matrix.
**F06PFF** Matrix-vector product, real triangular matrix.
**F06PGF** Matrix-vector product, real triangular band matrix.
**F06PHF** Matrix-vector product, real triangular packed matrix.
**F06SAF** Matrix-vector product, complex general matrix.
**F06SBF** Matrix-vector product, complex general band matrix.
**F06SCF** Matrix-vector product, complex Hermitian matrix.
**F06SDF** Matrix-vector product, complex Hermitian band matrix.
**F06SEF** Matrix-vector product, complex Hermitian packed matrix.
**F06SFF** Matrix-vector product, complex triangular matrix.
**F06SGF** Matrix-vector product, complex triangular band matrix.
**F06SHF** Matrix-vector product, complex triangular packed matrix.

## D1b5 : Matrix addition, subtraction

*Collected Algorithms of the ACM*

**A601** A set of eight Fortran subroutines for multiplying and adding pairs of sparse matrices in special cases, that is, in which one of the pair is full and/or a vector or an elementary matrix. Also provided are routines for transposing a sparse matrix and for multiplying two sparse matrices. (See J.M. McNamee, ACM TOMS 9 (1983) pp. 344-345.)

*IMSL Subprogram Library*

**VUABQ** Matrix addition (band + band symmetric matrices).
**VUAFB** Matrix addition (full + band matrices).
**VUAFQ** Matrix addition (full + band symmetric matrices).
**VUAFS** Matrix addition (full + symmetric matrices).
**VUASB** Matrix addition (symmetric + band matrices).
**VUASQ** Matrix addition (symmetric + band symmetric matrices).

*IMSL MATH/LIBRARY Subprogram Library*

**ACBCB** Add two complex band matrices, both in band storage mode.

**ARBRB** Add two band matrices, both in band storage mode.

*NAG Subprogram Library*

**F01CDF** Adds elements of the m-by-n matrices B and C and stores the results in elements of the matrix A.

**F01CEF** Subtracts elements of the matrix C from elements of the matrix B and stores the results in elements of the matrix A.

**F01CGF** Adds elements of the matrix B to elements in different positions in the matrix A.

**F01CHF** Subtracts elements of the matrix B from elements in a different position in the matrix A.

**F06PMF** Rank-1 update, real general matrix.

**F06PPF** Rank-1 update, real symmetric matrix.

**F06PQF** Rank-1 update, real symmetric packed matrix.

**F06PRF** Rank-2 update, real symmetric matrix .

**F06PSF** Rank-2 update, real symmetric packed matrix.

**F06SMF** Rank-1 update, complex general matrix, unconjugated vector.

**F06SNF** Rank-1 update, complex general matrix, conjugated vector.

**F06SPF** Rank-1 update, complex Hermitian matrix.

**F06SQF** Rank-1 update, complex Hermitian packed matrix.

**F06SRF** Rank-2 update, complex Hermitian matrix.

**F06SSF** Rank-2 update, complex Hermitian packed matrix.

*Scientific Desk PC Subprogram Library*

**D1B5** Adds the full storage mode n-by-n matrix A to the symmetric n-by-n matrix B stored in packed form and puts the result in the full matrix C.

## D1b6 : Matrix multiplication

*Collected Algorithms of the ACM*

**A601** A set of eight Fortran subroutines for multiplying and adding pairs of sparse matrices in special cases, that is, in which one of the pair is full and/or a vector or an elementary matrix. Also provided are routines for transposing a sparse matrix and for multiplying two sparse matrices. (See J.M. McNamee, ACM TOMS 9 (1983) pp. 344-345.)

*IMSL Subprogram Library*

**VMULBB** Matrix multiplication (band storage mode).

**VMULBF** Matrix multiplication (band by full matrices).

**VMULBS** Matrix multiplication (band by symmetric matrices).

**VMULFB** Matrix multiplication (full by band matrices).

**VMULFF** Matrix multiplication (full storage mode).

**VMULFM** Matrix multiplication of the transpose of matrix A by matrix B (full storage mode).

**VMULFP** Matrix multiplication of matrix A by the transpose of matrix B (full storage mode).

**VMULFQ** Matrix multiplication (full by band symmetric matrices).

**VMULFS** Matrix multiplication (full by symmetric matrices).

**VMULQB** Matrix multiplication (band symmetric by band matrices).

**VMULQF** Matrix multiplication (band symmetric by full matrices).
**VMULQQ** Matrix multiplication (band symmetric storage mode).
**VMULQS** Matrix multiplication (band symmetric by symmetric matrices).
**VMULSB** Matrix multiplication (symmetric by band matrices).
**VMULSF** Matrix multiplication (symmetric by full matrices).
**VMULSQ** Matrix multiplication (symmetric by band symmetric matrices).
**VMULSS** Matrix multiplication (symmetric storage mode).
**VTPROF** Transpose product of matrix (full storage mode).
**VTPROS** Transpose product of a matrix (symmetric storage mode).

*IMSL MATH/LIBRARY Subprogram Library*

**MCRCR** Multiply two complex rectangular matrices, AB.
**MXYTF** Multiply a matrix A by the transpose of a matrix B, $AB^T$.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**MRRRR** Multiply two real rectangular matrices, AB.
**MXTXF** Compute the transpose product of a matrix, $A^T A$.
**MXTYF** Multiply the transpose of matrix A by matrix B, $A^T B$.

*MAGEV Cyber 205 Subprogram Library*

**BBDMPY** To perform the matrix multiplication C = AB, where A is a rectangular banded block matrix, and B and C are rectangular matrices. A vectorized outer product method is used.
**MPYUTU** To perform the matrix multiplication S = LU, where L is the transpose of the supplied upper triangular matrix U, and S is the symmetric product matrix.
**MXMPY** To perform the matrix multiplication C = AB, where C, A and B are either all columnwise or all rowwise stored rectangular matrices of matching dimensions.
**MXMPYI** To perform the matrix multiplication C = AB, where C, A and B are either all columnwise or all rowwise stored rectangular matrices of matching dimensions. This routine uses the hardware dot product instruction. Thrashing will result if the matrices are too large.
**MXMPYT** To perform the matrix multiplication $C = A^T B$, where C, A and B are either all columnwise or all rowwise stored rectangular matrices of matching dimensions. This routine uses the hardware dot product instruction.

*NAG Subprogram Library*

**F01CKF** Returns with the result of the multiplication of two matrices B and C in the matrix A, with the option to overwrite B or C.
**F01CLF** Post-multiplies the matrix B with the transpose of the matrix C and places the result in the matrix A.
**F06FCF** Multiply real vector by diagonal matrix.

*Scientific Desk PC Subprogram Library*

**D1B61** Multiplies an m-by-n real matrix A by an n-by-n symmetric real matrix B stored in packed form and stores the result in the real m-by-n matrix C.
**D1B62** Multiplies an n-by-n real symmetric matrix A stored in packed form by an n-by-m real matrix B stores the result in the real n-by-m matrix C.
**D1B63** Forms $A^T A$, where A is a real n-by-m matrix. The result is stored in either packed or full form.
**D1B64** Forms $B=A^T A$, where A and B are real symmetric matrices stored in packed form.
**D1B65** Multiplies two symmetric real matrices stored in packed form.
**D1B6F** General matrix multiply. Multiplies real matrix A(L,M) by real matrix B(M,N) and stores the result in C(L,N).

## D1b7 : Matrix polynomial

*IMSL Subprogram Library*

**VPOLYF** Matrix polynomial (full storage mode).

*IMSL MATH/LIBRARY Subprogram Library*

**POLRG** Evaluate a real general matrix polynomial.

## D1b8 : Copy a matrix

*IMSL MATH/LIBRARY Subprogram Library*

**CCBCB** Copy a complex band matrix stored in complex band storage mode.

**CCGCG** Copy a complex general matrix.

**CRBRB** Copy a real band matrix stored in band storage mode.

**CRGRG** Copy a real general matrix.

*MAGEV Cyber 205 Subprogram Library*

**CMXMOV** To move a matrix from one location to another.

**MXMOV** To move a matrix from one location to another.

**SYMSTO** To copy a symmetric matrix, stored in one of five different modes, to another symmetric matrix, thereby optionally changing the storage mode.

*NAG Subprogram Library*

**F01CFF** Copies elements of the matrix B into different positions in the matrix A.

**F01CMF** Copies elements of one matrix into a second matrix.

**F01CNF** Copies a vector of length M into a row of a matrix.

## D1b9 : Matrix storage mode conversion

*IMSL Subprogram Library*

**VCVTBF** Storage mode conversion of matrices (band to full storage mode).

**VCVTCH** Storage mode conversion of matrices (full complex to Hermitian).

**VCVTFB** Storage mode conversion of matrices (full to band storage mode).

**VCVTFQ** Storage mode conversion (full to band symmetric storage mode).

**VCVTFS** Storage mode conversion of matrices (full to symmetric).

**VCVTHC** Storage mode conversion of matrices (Hermitian to full complex).

**VCVTQF** Storage mode conversion (band symmetric to full storage mode).

**VCVTQS** Storage mode conversion (band symmetric to symmetric storage mode).

**VCVTSF** Storage mode conversion of matrices (symmetric to full).

**VCVTSQ** Storage mode conversion (symmetric to band symmetric storage mode).

*IMSL MATH/LIBRARY Subprogram Library*

**CCBCG** Convert a complex matrix in band storage mode to a complex matrix in full storage mode.

**CCGCB** Convert a complex matrix in full storage mode to a matrix in complex band storage mode.

**CHBCB** Copy a complex Hermitian band matrix stored in band Hermitian storage mode to a complex band matrix stored in band storage mode.

**CHFCG** Extend a complex Hermitian matrix defined in its upper triangle to its lower triangle.

**CRBCB** Convert a real matrix in band storage mode to a complex matrix in band storage mode.

**CRBRG** Convert a real matrix in band storage mode to a matrix in full storage mode.

**CRGCG** Copy a real general matrix to a complex general matrix.

**CRGRB** Convert a real matrix in full storage mode to a matrix in band storage mode.

**CRRCR** Copy a real rectangular matrix to a complex rectangular matrix.

**CSBRB** Copy a real symmetric band matrix stored in band symmetric storage mode to a real band matrix stored in band storage mode.

**CSFRG** Extend a real symmetric matrix defined in its upper triangle to its lower triangle.

*MAGEV Cyber 205 Subprogram Library*

**SYMSTO** To copy a symmetric matrix, stored in one of five different modes, to another symmetric matrix, thereby optionally changing the storage mode.

*Scientific Desk PC Subprogram Library*

**D1B9** Converts a symmetric real matrix, A, stored in packed form to a full matrix, B. A and B may occupy the same memory locations.

## D1b10 : Elementary rotation (Givens transformation) of a matrix

*CMLIB Library (BLAS Sublibrary)*

**CROTG** Construct Givens plane rotation of complex matrix.

**SROTG** Construct Givens plane rotation of single precision matrix.

**SROTMG** Construct modified Givens plane rotation of single precision matrix.

*CMLIB Library (XBLAS Sublibrary)*

**CSROT** Applies Givens plane rotation to complex matrix.

## D1b11 : Elementary reflection (Householder transformation) of a matrix

*IMSL Subprogram Library*

**VHS12** Real Householder transformation computation and applications.

**VHSH2C** Complex Householder transformation to zero a single element of a matrix.

**VHSH2R** Real Householder transformation to zero a single element of a matrix.

**VHSH3R** Real Householder transformation to zero two elements of a matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**SHOUAP** Apply a Householder transformation in single precision.

# D2: Solutions of systems of linear equations

Software in this class is concerned with the solution of nonsingular systems of linear equations

$$Ax = b$$

for an unknown vector $x$. Routines for matrix inversion are also found in this section. However, matrix inversion is almost never the recommended way to solve this problem. Instead, routines specially designed for solving such systems are much to be preferred, both for speed and accuracy. The expression $A^{-1}b$ is simply a convenient way of writing "the solution of the system $Ax = b$", and solving this system is usually more efficient and accurate than than explicity forming the inverse and doing a matrix-vector multiply. One exception is the calculation of a matrix inverse in order to obtain covariances for some statistical applications. Software for solving problems where the solution is overdetermined or underdetermined, i.e., $A$ is not square or is singular, can be found in class D9.

The software in this section is classified according to the type of matrix $A$ to which it may be applied, so it is important to determine as much about the special structure of a problem as possible in order to choose the best routine.

Since operations involving matrices frequently form a major part of the computation time for a given problem, great effort has been expended to isolate the dominant part of a particular computation into a single routine in order to avoid unnecessary repetitions when possible. This has the somewhat annoying consequence of often requiring two routines to solve a single problem. However, in some cases this isolation can significantly reduce the computation time. As an example, suppose we want to solve the linear system above for many different vectors $b$. The procedure for solving would be two steps:

(a) factor $A$ as a product of two or three matrices of a special form independent of the vector $b$, and

(b) use this factorization to solve for $x$ given $b$.

If $A$ is of order $n$, then generally step (a) requires computational time proportional to $n^3$, whereas step (b) requires only time proportional to $n^2$. Hence, if one has many different right side vectors $b$, it is clearly more efficient to do step (a) only once. This situation occurs, for example, when solving time-dependent partial differential equations by implicit methods.

Most subprogram libraries have large selections of programs for solving this problem. These include routines which will perform the computation in parts, as in (a) and (b) above. Much of this software is modeled after LINPACK [DBMS79]. Most libraries include drivers which use these software parts to solve a single problem in one call.

## References

[DBMS79] J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart. *LINPACK User's Guide.* SIAM, Philadelphia, 1979.

[FM67] G. Forsythe and C. Moler. *Computer Solution of Linear Algebraic Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1967.

[GL81] A. George and J. W. H. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1981.

[GVL83] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 1983.

[HY81] L. A. Hageman and D. M. Young. *Applied Iterative Methods.* Academic Press, New York, 1981.

| D2a : | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real nonsymmetric matrices)** |
|---|---|

*IMSL MATH/LIBRARY Subprogram Library*

**LSLCC** Solve a complex circulant linear system.

**LSLTC** Solve a complex Toeplitz linear system.

**LSLTO** Solve a real Toeplitz linear system.

*MAGEV Cyber 205 Subprogram Library*

**MSWIEN** To solve one or several matrix equations of the type TX = A, where T is a Toeplitz matrix, X is a vector with its first element equal to 1.0, and A is a base vector of the form (ALFA,0, 0,...,0). The algorithm used is known to geophysicists as the spike version of the Wiener-Levenson algorithm.

| D2a1 : | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (general real nonsymmetric matrices)** |
|---|---|

*Collected Algorithms of the ACM*

**A522** ESOLVE: a Fortran subroutines for the exact solution of systems of linear equations with multiple-precision integer coefficients by congruence techniques. (See S. Cabay and T.P.L. Lam, ACM TOMS 3 (1977) pp. 404-410.)

**A576** MODGE and REFINE: Fortran subroutines for solving an n-by-n system of (possibly singular) linear algebraic equations. The algorithm consists of Gaussian elimination combined with a new pivoting strategy which is particularly well suited to problems where residuals can be made small by solving for fewer than n of the unknowns. (See I. Barrodale and G.F. Stuart, ACM TOMS 7 (1981) pp. 391-397.)

**A578** Fortran subroutines for solving real linear equations in a paged virtual store. BLCFAC performs Gaussian elimination with partial pivoting on a real square matrix A with operations on blocks of consecutive columns grouped together to minimize paging on a machine with a paged virtual memory. BLCSOL solves systems based upon A or A-transpose with multiple right-hand sides using a similar strategy. (See J.J. Du Croz et al., ACM TOMS 7 (1981) pp. 537-189.)

**A641** EXSOLG: a Fortran subprogram for the exact solution of general (m-by-n) systems of linear equations with integer coefficients. If the system is singular a least square solution is computed. (See J. Springer, ACM TOMS 12 (1986) p. 149.)

*CMLIB Library (LINDRV Sublibrary)*

**SGEFS** Factors and solves a general n-by-n single precision system of linear equations.

**SGEIR** Factors and solves a general single precision system of linear equations and estimates solution accuracy (requires n-by-n extra storage).

*CMLIB Library (LINPAKS Sublibrary)*

**SGECO** Computes LU factorization of real general matrix and estimates its condition.

**SGEDI** Uses LU factorization of real general matrix to compute its determinant and/or inverse.

**SGEFA** Computes LU factorization of real general matrix.

**SGESL** Uses LU factorization of real general matrix to solve systems.

**SQRSL** Applies the output of SQRDC to compute coordinate transformations, projections, and least squares solutions (general real matrix).

*IMSL Subprogram Library*

| | |
|---|---|
| **LEQIF** | Linear equation solution – full matrices (virtual memory version). |
| **LEQOF** | Linear equation solution – full matrices (out-of-core version). |
| **LEQT1F** | Linear equation solution – full storage mode – space economizer solution. |
| **LEQT2F** | Linear equation solution – full storage mode – high accuracy solution. |
| **LINV1F** | Inversion of a matrix – full storage mode space economizer solution. |
| **LINV2F** | Inversion of a matrix – full storage mode high accuracy solution. |
| **LINV3F** | In place inverse, equation solution, and/or determinant evaluation – full storage mode. |
| **LUDATF** | LU decomposition by the Crout algorithm with optional accuracy test. |
| **LUELMF** | Elimination part of solution of Ax=b (full storage mode). |
| **LUREFF** | Refinement of solution to linear equations full storage mode. |

*IMSL MATH/LIBRARY Subprogram Library*

| | |
|---|---|
| **LINRG** | Compute the inverse of a real general matrix. |
| **LFCRG** | Compute the LU factorization of a real general matrix and estimate its $L_1$ condition number. |
| **LFIRG** | Use iterative refinement to improve the solution of a real general system of linear equations. |
| **LFSRG** | Solve a real general system of linear equations given the LU factorization of the coefficient matrix. |
| **LFTRG** | Compute the LU factorization of a real general matrix. |
| **LSARG** | Solve a real general system of linear equations with iterative refinement. |
| **LSLRG** | Solve a real general system of linear equations without iterative refinement. |

*IMSL STAT/LIBRARY Subprogram Libraries*

| | |
|---|---|
| **LFSRG** | Solve a real general system of linear equations given the LU factorization of the coefficient matrix. |
| **LFTRG** | Compute the LU factorization of a real general matrix. |
| **LINRG** | Compute the inverse of a real general matrix. |
| **LSLRG** | Solve a real general system of linear equations without iterative refinement. |

*MAGEV Cyber 205 Subprogram Library*

| | |
|---|---|
| **GEL** | To solve a system of linear equations, i.e., the matrix equation AX = B, where A is a full matrix. |
| **MXINV** | To compute the inverses of a set of square matrices, using neither scaling nor pivoting. |

*NAG Subprogram Library*

| | |
|---|---|
| **F01AAF** | Calculates the approximate inverse of a real matrix by Crout's method. |
| **F01BTF** | Decomposes a real matrix into a product of triangular matrices LU by Gaussian elimination with partial pivoting. The block-column method used is designed for efficient working on paged virtual machines. |
| **F03AFF** | Decomposes a real matrix into triangular factor matrices LU by Crout's method and evaluates the determinant. |
| **F04AAF** | Calculates the approximate solution of a set of real linear equations with multiple right hand sides by Crout's factorisation method. |
| **F04AEF** | Calculates the accurate solution of a set of real linear equations with multiple right hand sides by Crout's factorisation method. |
| **F04AHF** | Calculates the accurate solution of a set of real linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F03AFF. |

**F04AJF** Calculates the approximate solution of a set of real linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F03AFF.

**F04ARF** Calculates the approximate solution of a set of real linear equations with a single right hand side, Ax=b, by Crout's factorisation method.

**F04ATF** Calculates the accurate solution of a set of real linear equations with a single right hand side, Ax=b, by Crout's factorisation method.

**F04AYF** Calculates the approximate solution of a set of real linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F01BTF.

*NASHLIB Subprogram Library*

**A5A6** Perform Gaussian elimination with partial pivoting on a general system of linear equations. Illustrates use of algorithms A5GE and A6BS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 62-63.)

*NMS Subprogram Library*

**SGEFS** Solves n-by-n system AX=B; returns the solutions in B, the number of correct digits, and 1/cond(A).

*PORT Subprogram Library*

**LINEQ** Solves a real system of linear equations, AX=B, where B is allowed to be a matrix or a vector.

*Scientific Desk PC Subprogram Library*

**D2A11** Factors a real matrix by Gaussian elimination. Several other routines such as D2A12, D2A13, or D2A14, can make use of the factorization to solve a system of linear equations, compute inverse, or determinant.

**D2A12** Solves the real system of equations AX=B using the factors computed by D2A11.

**D2A13** Computes the determinant and/or inverse of a real matrix using the factors computed by D2A11.

**D2A14** Estimates the condition of a real matrix, given its factors from routine D2A11 and its 1-norm from routine D1B21.

*SCRUNCH Subprogram Library*

**DECOMP** Decomposes a matrix using Gaussian elimination into a lower and upper triangular factorization and estimates the condition of the matrix. This routine is usually used in conjunction with SOLVE which takes the LU factorization produced by DECOMP to actually solve a system of linear equations. In BASIC.

## D2a2 : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real nonsymmetric banded matrices)

*Collected Algorithms of the ACM*

**A546** SOLVEBLOK: a Fortran subprogram for solving almost block diagonal linear systems. Such matrices arise naturally in piecewise polynomial interpolation or approximation and in finite element methods for two-point boundary value problems. (See C. de Boor and R. Weiss, ACM TOMS 6 (1980) pp. 88-91.)

**A603** COLROW and ARCECO: Fortran subroutines for solving certain almost block diagonal linear systems by modified alternate row and column elimination. Such systems arise when solving boundary-value problems for ordinary differential equations. COLROW is designed for systems whose blocks all have the same dimension; ARCECO is designed for systems whose blocks may have different dimensions. (See J.C. Diaz, G. Fairweather, and P. Keast, ACM TOMS 9 (1983) pp. 376-380.)

**A664** GBSOL: A Fortran implementation of a Gauss algorithm to solve systems with large banded matrices using random-access disk storage. (See G. Schrauf, ACM TOMS 14 (1988) pp. 257-260.)

*CMLIB Library (LINDRV Sublibrary)*

**SNBCO** Factors a real band matrix by Gaussian elimination and estimates condition of the matrix.

**SNBFA** Factors a single precision band matrix by elimination.

**SNBFS** Factors and solves a general nonsymmetric single precision banded system of linear equations.

**SNBIR** Factors and solves a general nonsymmetric single precision banded system of equations and estimates solution accuracy (needs N*(2*ML+MU) extra storage).

**SNBSL** Solves a general nonsymmetric single precision banded system of linear equations using factors computed previously.

*CMLIB Library (LINPAKS Sublibrary)*

**SGBCO** Computes LU factorization of real band matrix and estimates its condition.

**SGBFA** Computes LU factorization of real band matrix.

**SGBSL** Uses LU factorization of real band matrix to solve systems.

*CMLIB Library (SLVBLK Sublibrary)*

**SLVBLK** Solves Ax=b where A is an almost block diagonal matrix. These arise in finite element or piecewise polynomial approximation.

*IMSL Subprogram Library*

**LEQT1B** Linear equation solution – band storage mode – space economizer solution.

**LEQT2B** Linear equation solution – band storage mode – high accuracy solution.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCRB** Compute the LU factorization of a real matrix in band storage mode and estimate its $L_1$ condition number.

**LFIRB** Use iterative refinement to improve the solution of a real system of linear equations in band storage mode.

**LFSRB** Solve a real system of linear equations given the LU factorization of the coefficient matrix in band storage mode.

**LFTRB** Compute the LU factorization of a real matrix in band storage mode.

**LSARB** Solve a real system of linear equations in band storage mode with iterative refinement.

**LSLRB** Solve a real system of linear equations in band storage mode without iterative refinement.

*MAGEV Cyber 205 Subprogram Library*

**BBDGEL** To solve the linear system of equations AX = B, where A is a banded block matrix, and X and B are rectangular matrices.

**BDGEL** To solve the linear system of equations AX = B, where A is a banded matrix, and X and B are rectangular matrices.

*NAG Subprogram Library*

**F01LBF** Decomposes a general real band matrix of order N, with M1 sub-diagonals and M2 super-diagonals, where M1+M2+1 is much less than N, into triangular matrices using Gaussian elimination with partial pivoting.

**F04LDF** Calculates the approximate solution of a set of general real band linear equations with multiple right-hand sides AX=B where A has been decomposed into triangular matrices using F01LBF.

*Scientific Desk PC Subprogram Library*

**D2A21** Factors a real band matrix by elimination. Routine D2A22 can then be used to solve a banded system of linear equations.

**D2A22** Solves the real band system AX=B or $A^T X=B$ using the factors computed by D2A21.

## D2a2a : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real nonsymmetric tridiagonal matrices)

*CMLIB Library (LINPAKS Sublibrary)*

**SGTSL** Factors a real tridiagonal matrix and simultaneously solves a system.

*MAGEV Cyber 205 Subprogram Library*

**TRID** To solve the linear system of equations AX = B where A is a tridiagonal square coefficient matrix supplied as real 64-bit (TRID) or real 32-bit (HTRID) numbers in a packed format. Special logic is included to significantly increase the efficiency when A is made up of many smaller, equally sized tridiagonal matrices.

*NAG Subprogram Library*

**F01LEF** Factorises a real tridiagonal matrix into triangular factors, $T - \lambda I = PLU$, using Gaussian elimination with partial pivoting.

**F04EAF** Calculates the approximate solution of a set of real tridiagonal linear equations with a single right hand side, Tx = b.

**F04LEF** Solves a system of tridiagonal equations following the factorisation by F01LEF. This routine is intended for applications such as inverse iteration as well as straightforward linear equation applications.

*Scientific Desk PC Subprogram Library*

**D2A2A** Solves a system of linear equations where the coefficient matrix is tridiagonal.

## D2a3 : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real nonsymmetric triangular matrices)

*CMLIB Library (LINPAKS Sublibrary)*

**STRCO** Estimates the condition of real triangular matrix.

**STRDI** Computes determinant and/or inverse of real triangular matrix.

**STRSL** Solves systems with real triangular matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCRT** Estimate the condition number of a real triangular matrix.

**LINRT** Compute the inverse of a real triangular matrix.

**LSLRT** Solve a real triangular system of linear equations.

*MAGEV Cyber 205 Subprogram Library*

**INVU** To compute the upper triangular inverse R of a non-singular upper triangular matrix U.

*NAG Subprogram Library*

**F01LZF** Factorises a real upper triangular matrix A as $QCP^T$ where Q and P are orthogonal and C is upper bidiagonal.

**F06PJF** Solve system of equations, real triangular matrix.

**F06PKF** Solve system of equations, real triangular band matrix.

**F06PLF** Solve system of equations, real triangular packed matrix.

**D2a4 : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real nonsymmetric sparse matrices)**

*Collected Algorithms of the ACM*

**A533** NSPIV: a Fortran subroutine for solving a sparse systems of linear equations by sparse Gaussian elimination with partial pivoting. (See A.H. Sherman, ACM TOMS 4 (1978) pp. 391-398.)

**A589** SICEDR: a Fortran subroutine for improving the accuracy of computed real matrix eigenvalues and improving or computing the associated eigenvector. (See J.J. Dongarra, ACM TOMS 8 (1982) pp. 371-375.)

*CMLIB Library (FISHPAK Sublibrary)*

**POIS3D** Solves block tridiagonal linear systems of algebraic equations arising from the discretization of separable elliptic partial differential equations in 3D.

**POISTG** Solves block tridiagonal linear systems of algebraic equations arising from the discretization of separable elliptic partial differential equations.

*MAGEV Cyber 205 Subprogram Library*

**D4GEL** To solve a linear system of equations, where the coefficient matrix is a D4-ordered block matrix.

*NAG Subprogram Library*

**F01BRF** Decomposes a real sparse matrix. The routine either forms the LU-decomposition of a permutation of the entire matrix, or, optionally, first permutes the matrix to block lower triangular form and then only decomposes the diagonal blocks.

**F01BSF** Decomposes a real sparse matrix using the pivotal sequence previously obtained by F01BRF when a matrix of the same sparsity pattern was decomposed.

**F04AXF** Calculates the approximate solution of a set of real sparse linear equations with a single right hand side, $Ax = b$ or $A^T x = b$, where A has been decomposed by F01BRF or F01BSF.

**F04QAF** Solves sparse unsymmetric equations, sparse linear least squares problems and sparse damped linear least squares problems using a Lanczos algorithm.

**D2b1a : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real symmetric indefinite matrices)**

*CMLIB Library (LINPAKS Sublibrary)*

**SSICO** Computes factorization of real symmetric indefinite matrix and estimates its condition.

**SSIDI** Uses factorization of real symmetric indefinite matrix to compute its determinant and/or inverse.

**SSIFA** Computes factorization of real symmetric indefinite matrix.

**SSISL** Uses factorization of real symmetric indefinite matrix to solve systems.

**SSPCO** Computes factorization of real symmetric indefinite matrix stored in packed form and estimates its condition.

**SSPDI** Uses factorization of real symmetric indefinite matrix stored in packed form to compute its determinant and/or inverse.

**SSPFA** Computes factorization of real symmetric indefinite matrix stored in packed form.

**SSPSL** Uses factorization of real symmetric indefinite matrix stored in packed form to solve systems.

*IMSL Subprogram Library*

**LEQ1S** Linear equation solution – indefinite matrix – symmetric storage mode – space economizer solution.

**LEQ2S** Linear equation solution – indefinite matrix – symmetric storage mode – high accuracy solution.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCSF** Compute the $UDU^T$ factorization of a real symmetric matrix and estimate its $L_1$ condition number.

**LFISF** Use iterative refinement to improve the solution of a real symmetric system of linear equations.

**LFSSF** Solve a real symmetric system of linear equations given the $UDU^T$ factorization of the coefficient matrix.

**LFTSF** Compute the $UDU^T$ factorization of a real symmetric matrix.

**LSASF** Solve a real symmetric system of linear equations with iterative refinement.

**LSLSF** Solve a real symmetric system of linear equations without iterative refinement.

*Scientific Desk PC Subprogram Library*

**D2B1A** Factors a real symmetric matrix stored in packed form by elimination with symmetric pivoting. The factors can then be used by routine D2B1A2 to solve a symmetric system of linear equations, routine D2B1A3 can be used to form the inverse or compute the determinant of the matrix, or routine D2B1A4 can be used to compute the condition number of the matrix.

**D2B1A2** Solves the real symmetric system AX=B using the factors computed by D2B1A1.

**D2B1A3** Computes the determinant, inertia, or inverse of a real symmetric matrix using the factors from D2B1A1.

**D2B1A4** Estimates the condition of a real symmetric matrix A stored in packed form from the factors computed by D2B1A1.

## D2b1b : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real symmetric positive definite matrices)

*CMLIB Library (LINDRV Sublibrary)*

**SPOFS** Factors and solves a symmetric positive definite single precision system of linear equations.

**SPOIR** Factors and solves a symmetric positive definite single precision system of equations and estimates solution accuracy (needs n-by-n extra storage).

*CMLIB Library (LINPAKS Sublibrary)*

**SCHDC** Compute Cholesky decomposition of real positive definite matrix with optional pivoting.

**SPOCO** Uses Cholesky algorithm to factor real positive definite matrix and estimate its condition.

**SPODI** Uses factorization of real positive definite matrix to compute its determinant and/or inverse.

**SPOFA** Uses Cholesky algorithm to factor real positive definite matrix.

**SPOSL** Uses factorization of real positive definite matrix to solve systems.

**SPPCO** Uses Cholesky algorithm to factor real positive definite matrix stored in packed form and estimate its condition.

**SPPDI** Uses factorization of real positive definite matrix stored in packed form to compute its determinant and/or inverse.

**SPPFA** Uses Cholesky algorithm to factor real positive definite matrix stored in packed form.

**SPPSL** Uses factorization of real positive definite matrix stored in packed form to solve systems.

*IMSL Subprogram Library*

**LEQT1P** Linear equation solution – positive definite matrix – symmetric storage mode – space economizer solution.

**LEQT2P** Linear equations solution – positive definite matrix – symmetric storage mode – high accuracy solution.

**LINV1P** Inversion of matrix – positive definite symmetric storage mode – space economizer solution.

**LINV2P** Inversion of a matrix – positive definite symmetric storage mode – high accuracy solution.

**LINV3P** In place inverse, equation solution, positive definite matrix – symmetric storage mode.

**LUDECP** Decomposition of a positive definite matrix symmetric storage mode.

**LUELMP** Elimination part of the solution of Ax=b positive definite matrix – symmetric storage mode.

**LUREFP** Refinement of solution to linear equations positive definite matrix – symmetric storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

**LCHRG** Compute the Cholesky decomposition of a symmetric positive semidefinite matrix with optional column pivoting.

**LFCDS** Compute the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix and estimate its $L_1$ condition number.

**LFIDS** Use iterative refinement to improve the solution of a real symmetric positive definite system of linear equations.

**LFSDS** Solve a real symmetric positive definite system of linear equations given the $R^T R$ Cholesky factorization of the coefficient matrix.

**LFTDS** Compute the $R^T R$ Cholesky factorization of a real symmetric positive definite matrix.

**LINDS** Compute the inverse of a real symmetric positive definite matrix.

**LSADS** Solve a real symmetric positive definite system of linear equations with iterative refinement.

**LSLDS** Solve a real symmetric positive definite system of linear equations without iterative refinement.

*IMSL STAT/LIBRARY Subprogram Library*

**CHFAC** Compute an upper triangular factorization of a real symmetric nonnegative definite matrix.

*MAGEV Cyber 205 Subprogram Library*

**INVSPD** To compute the inverse of a positive definite symmetric matrix.

**SYMUUT** To perform the Cholesky decomposition A = UL of the positive definite symmetric matrix A. U here denotes a non-singular upper triangular matrix, and L stands for its lower triangular transpose. Note that the order of the matrices U and L is reversed from that in the conventional LU decomposition.

*NAG Subprogram Library*

• **F01ABF** Calculates the accurate inverse of a real symmetric positive definite matrix by Cholesky's method and iterative refinement. (Simplified parameter list).

**F01ACF** Calculates the accurate inverse of a real symmetric positive definite matrix by Cholesky's method and iterative refinement.

**F01ADF** Calculates the approximate inverse of a real symmetric positive definite matrix by Cholesky's method.

**F01BQF** Forms the Cholesky decomposition of a real symmetric matrix G whose lower triangle only is stored. If G is not positive definite, the routine forms the Cholesky decomposition of G + E where E is a diagonal matrix.

**F01BUF** Decomposes a symmetric positive definite band matrix into the form $ULDL^T U^T$ where U is a unit upper triangular matrix, L is unit lower triangular and D is diagonal. It is specifically designed to precede F01BVF.

• **easy-to-use version**

**F01BXF** Performs the Cholesky factorization $U^T U$ of a real symmetric positive definite matrix A.

**F03AEF** Decomposes a real symmetric positive definite matrix A into triangular matrices $LL^T$ using Cholesky's method, and evaluates the determinant.

**F04ABF** Calculates the accurate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, by Cholesky's decomposition method.

**F04AFF** Calculates the accurate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices $A = LL^T$ using F03AEF.

**F04AGF** Calculates the approximate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices $A = LL^T$ using F03AEF.

**F04AQF** Calculates the approximate solution of a set of real symmetric positive definite linear equations with a single right hand side Ax=b where A has been decomposed into $LDL^T$ using F01BQF. (Economical storage).

**F04ASF** Calculates the accurate solution of a set of real symmetric positive definite linear equations with a single right hand side, Ax=b, by Cholesky's decomposition method.

**F04AZF** Calculates the approximate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F01BXF.

*NASHLIB Subprogram Library*

**A24** Use conjugate gradient algorithm to find the solution of a set of linear equations with nonnegative definite coefficient matrix. Illustrates use of algorithm A24CG. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 197-198.)

**A7A8** Use Cholesky algorithm to solve a symmetric system of linear equations. Illustrates use of algorithms A7CH and A8CS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 74-75.)

*Scientific Desk PC Subprogram Library*

**D2B11** Factors a real symmetric positive definite matrix stored in packed form. Then routines such as D2B12, D2B13, or D2B14 can be used to solve a system of linear equations, or find the determinant, condition number, or inverse of the matrix.

**D2B12** Solves the real symmetric positive definite system AX=B using the factorization obtained from D2B11.

**D2B13** Finds the determinant and/or the inverse of a real symmetric positive definite matrix using the factorization obtained from D2B11.

**D2B14** Finds the condition number of a real symmetric positive definite matrix A stored in packed form using the factorization obtained from D2B11 and the 1-norm of A obtained from D1B22.

## D2b2 : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real symmetric positive definite banded matrices)

*Collected Algorithms of the ACM*

**A512** FACTOR, RHS, and SOLVE: Fortran subroutines for solving symmetric positive definite periodic quindiagonal systems of linear equations. (See A. Benson, and D.J. Evans, ACM TOMS 3 (1977) pp. 96-103.)

*CMLIB Library (LINPAKS Sublibrary)*

**SPBCO** Uses Cholesky algorithm to compute factorization of real positive definite band matrix and estimates its condition.

**SPBFA** Uses Cholesky algorithm to compute factorization of real positive definite band matrix.

**SPBSL** Uses factorization of real positive definite band matrix to solve systems.

*IMSL Subprogram Library*

**LEQ1PB** Linear equation solution – positive definite symmetric band matrix – band symmetric storage mode – space economizer solution.

**LEQ2PB** Linear equation solution – positive definite band symmetric matrix – band symmetric storage mode – high accuracy solution.

**LIN1PB** Inversion of a matrix – positive definite band symmetric matrix – band symmetric storage mode – space economizer solution.

**LIN2PB** Inversion of matrix – positive definite band symmetric matrix - band symmetric storage mode – high accuracy solution.

**LUDAPB** Decomposition of a positive definite band symmetric matrix – band symmetric storage mode.

**LUELPB** Elimination part of solution of Ax=b positive definite band symmetric matrix – band symmetric storage mode.

**LUREPB** Refinement of solution to linear equations positive definite band symmetric matrix – band symmetric storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCQS** Compute the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode and estimate its $L_1$ condition number.

**LFDQS** Compute the determinant of a real symmetric positive definite matrix in band symmetric storage mode given its Cholesky factorization.

**LFIQS** Use iterative refinement to improve the solution of a real symmetric positive definite system of linear equations in band symmetric storage mode.

**LFSQS** Solve a real symmetric positive definite system of linear equations given the factorization of the coefficient matrix in band symmetric storage mode.

**LFTQS** Compute the Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode.

**LSAQS** Solve a real symmetric positive definite system of linear equations in band symmetric storage mode with iterative refinement.

**LSLQS** Solve a real symmetric positive definite system of linear equations in band symmetric storage mode without iterative refinement.

*NAG Subprogram Library*

**F01MCF** Computes the Cholesky factorisation of a symmetric, positive-definite, variable-bandwidth matrix.

**F03AGF** Decomposes a real symmetric positive definite band matrix into triangular matrices using Cholesky's method and evaluates the determinant.

**F04ACF** Calculates the approximate solution of a set of real symmetric positive definite band equations with multiple right hand sides by Cholesky's decomposition method.

**F04ALF** Calculates the approximate solution of a set of real symmetric positive definite band linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F03AGF.

**F04MCF** Computes the approximate solution of a system of real linear equations with multiple right hand sides, Ax = B, where A is a symmetric positive-definite variable-bandwidth matrix, which has previously been factorised by F01MCF. Related systems may also be solved.

| | |
|---|---|
| **D2b2a :** | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real symmetric tridiagonal matrices)** |

*CMLIB Library (LINPAKS Sublibrary)*

SPTSL Decomposes real symmetric positive definite tridiagonal matrix and simultaneously solves a system.

*NAG Subprogram Library*

F04FAF Calculates the approximate solution of a set of real symmetric positive definite tridiagonal linear equations.

*Scientific Desk PC Subprogram Library*

D2B2S Solves a system of linear equations where the coefficient matrix is symmetric positive definite tridiagonal.

| | |
|---|---|
| **D2b4 :** | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (real symmetric sparse matrices)** |

*Collected Algorithms of the ACM*

A586 ITPACK 2C: a set of seven Fortran subprograms (JCG, JSI, SOR, SSORCG, SSORSI, RSCG, and RSSI) for solving large sparse linear systems by adaptive accelerated iterative methods. (See D.R. Kincaid et al., ACM TOMS 8 (1982) pp. 302-322.)

*IMSL MATH/LIBRARY Subprogram Library*

JCGRC Solve a real symmetric definite linear system using the Jacobi-preconditioned conjugate gradient method with reverse communication.

PCGRC Solve a real symmetric definite linear system using a preconditioned conjugate gradient method with reverse communication.

*NAG Subprogram Library*

F01MAF Find an incomplete Cholesky factorisation of a sparse symmetric positive definite matrix A.

F04MAF Solves a sparse symmetric positive definite system of linear equations, Ax = b, using a preconditioned conjugate gradient method, following a factorization of A by F01MAF.

F04MBF Solves a system of sparse symmetric linear equations $(A-\lambda I)x=b$, using a Lanczos algorithm.

| | |
|---|---|
| **D2c1 :** | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (general complex non-Hermitian matrices)** |

*CMLIB Library (LINDRV Sublibrary)*

CGEFS Factors and solves a general complex system of linear equations.

CGEIR Factors and solves a general complex system of linear equations and provides estimate of accuracy of the solution (needs n-by-n extra storage).

*CMLIB Library (LINPAKC Sublibrary)*

CGECO Compute LU factorization of general complex matrix and estimate its condition.

CGEDI Compute determinant and/or inverse of general complex matrix from its LU factors.

CGEFA Compute LU factorization of general complex matrix.

CGESL Use LU factorization of general complex matrix to solve systems.

**CQRSL** Applies the output of CQRDC to compute coordinate transformations, projections, and least squares solutions (general complex matrix).

*IMSL Subprogram Library*

**LEQ2C** Linear equation solution – complex matrix high accuracy solution.

**LEQT1C** Matrix decomposition, linear equation solution – space economizer solution complex matrices.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCCG** Compute the LU factorization of a complex general matrix and estimate its $L_1$ condition number.

**LFICG** Use iterative refinement to improve the solution of a complex general system of linear equations.

**LFSCG** Solve a complex general system of linear equations given the LU factorization of the coefficient matrix.

**LFTCG** Compute the LU factorization of a complex general matrix.

**LINCG** Compute the inverse of a complex general matrix.

**LSACG** Solve a complex general system of linear equations with iterative refinement.

**LSLCG** Solve a complex general system of linear equations without iterative refinement.

*MAGEV Cyber 205 Subprogram Library*

**CGEL** To solve a system of linear equations, i.e., the matrix equation AX = B, where A is a full matrix.

*NAG Subprogram Library*

**F03AHF** Decomposes a complex matrix into triangular factor matrices LU by Crout's method and evaluates the determinant.

**F04ADF** Calculates the approximate solution of a set of complex linear equations with multiple right hand sides by Crout's factorisation method.

**F04AKF** Calculates the approximate solution of a set of complex linear equations with multiple right hand sides AX=B where A has been decomposed into triangular matrices using F03AHF.

*PORT Subprogram Library*

**CLINQ** Solves a complex system of linear equations. Coefficient matrix must be input as two real matrices.

## D2c2 : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex non-Hermitian banded matrices)

*CMLIB Library (LINDRV Sublibrary)*

**CNBCO** Factors a complex band matrix by Gaussian elimination and estimates its condition number.

**CNBFA** Factors a non-symmetric complex band matrix by elimination.

**CNBFS** Factors and solves a general complex band matrix system of linear equations.

**CNBIR** Factors and solves a general nonsymmetric complex band system of equations and estimates accuracy of the solution (requires N*(2*ML+MU) extra storage).

**CNBSL** Solves the nonsymmetric complex band system of equations using factors previously computed.

*CMLIB Library (LINPAKC Sublibrary)*

**CGBCO** Compute LU factorization of complex band matrix and estimate its condition.

**CGBFA** Compute LU factorization of general complex band matrix.

**CGBSL** Uses LU factorization of complex band matrix to solve systems.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCCB** Compute the LU factorization of a complex matrix in band storage mode and estimate its $L_1$ condition number.

**LFICB** Use iterative refinement to improve the solution of a complex system of linear equations in band storage mode.

**LFSCB** Solve a complex system of linear equations given the LU factorization of the coefficient matrix in band storage mode.

**LFTCB** Compute the LU factorization of a complex matrix in band storage mode.

**LSACB** Solve a complex system of linear equations in band storage mode with iterative refinement.

**LSLCB** Solve a complex system of linear equations in band storage mode without iterative refinement.

*NAG Subprogram Library*

**F01NAF** Finds the LU factorisation of a complex band matrix by Gaussian elimination with partial pivoting.

**F04NAF** Solves a system of complex band equations following factorisation by F01NAF. This routine is intended for applications such as inverse iteration as well as straightforward linear equation applications.

## D2c2a : **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex non-Hermitian tridiagonal matrices)**

*CMLIB Library (LINPAKC Sublibrary)*

**CGTSL** Solves systems with general complex tridiagonal matrix.

## D2c3 : **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex non-Hermitian triangular matrices)**

*CMLIB Library (LINPAKC Sublibrary)*

**CTRCO** Estimates condition of complex triangular matrix.

**CTRDI** Computes determinant and/or inverse of complex triangular matrix.

**CTRSL** Solves systems with complex triangular matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCCT** Estimate the condition number of a complex triangular matrix.

**LINCT** Compute the inverse of a complex triangular matrix.

**LSLCT** Solve a complex triangular system of linear equations.

*NAG Subprogram Library*

**F06SJF** Solve system of equations, complex triangular matrix.

**F06SKF** Solve system of equations, complex triangular band matrix.

**F06SLF** Solve system of equations, complex triangular packed matrix.

**D2d1a : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex Hermitian indefinite matrices)**

*CMLIB Library (LINPAKC Sublibrary)*

**CHICO** Computes factorization of complex Hermitian indefinite matrix and estimates its condition.

**CHIDI** Uses factorization of complex Hermitian indefinite matrix to compute its inertia, determinant, and/or inverse.

**CHIFA** Computes factorization of complex Hermitian indefinite matrix.

**CHISL** Uses factorization of complex Hermitian indefinite matrix to solve systems.

**CHPCO** Computes factorization of complex Hermitian indefinite matrix stored in packed form and estimates its condition.

**CHPDI** Uses factorization of complex Hermitian indefinite matrix stored in packed form to compute its inertia, determinant, and inverse.

**CHPFA** Computes factorization of complex Hermitian indefinite matrix stored in packed form.

**CHPSL** Uses factorization of complex Hermitian indefinite matrix stored in packed form to solve systems.

**CSICO** Computes factorization of complex symmetric indefinite matrix and estimates its condition.

**CSIDI** Uses factorization of complex symmetric indefinite matrix to compute its determinant and/or inverse.

**CSIFA** Computes factorization of complex symmetric indefinite matrix.

**CSISL** Uses factorization of complex symmetric indefinite matrix to solve systems.

**CSPCO** Computes factorization of complex symmetric indefinite matrix stored in packed form and computes its condition.

**CSPDI** Uses factorization of complex symmetric indefinite matrix stored in packed form to compute its determinant and/or inverse.

**CSPFA** Computes factorization of complex symmetric indefinite matrix stored in packed form.

**CSPSL** Uses factorization of complex symmetric indefinite matrix stored in packed form to solve systems.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCHF** Compute the $UDU^H$ factorization of a complex Hermitian matrix and estimate its $L_1$ condition number.

**LFDHF** Compute the determinant of a complex Hermitian matrix given the $UDU^H$ factorization of the matrix.

**LFIHF** Use iterative refinement to improve the solution of a complex Hermitian system of linear equations.

**LFSHF** Solve a complex Hermitian system of linear equations given the $UDU^H$ factorization of the coefficient matrix.

**LFTHF** Compute the $UDU^H$ factorization of a complex Hermitian matrix.

**LSAHF** Solve a complex Hermitian system of linear equations with iterative refinement.

**LSLHF** Solve a complex Hermitian system of linear equations without iterative refinement.

**D2d1b : Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex Hermitian positive definite matrices)**

*CMLIB Library (LINDRV Sublibrary)*

**CPOFS** Factors and solves positive definite symmetric complex system of linear equations.

**CPOIR** Solves positive definite Hermitian complex system of linear equations and estimates the accuracy of the solution (requires n-by-n extra storage).

*CMLIB Library (LINPAKC Sublibrary)*

**CCHDC** Compute Cholesky decomposition of complex positive definite matrix with optional pivoting.

**CPOCO** Uses Cholesky algorithm to compute factorization of complex positive definite matrix and estimates its condition.

**CPODI** Uses factorization of complex positive definite matrix to compute its determinant and/or inverse.

**CPOFA** Uses Cholesky algorithm to compute factorization of complex positive definite matrix.

**CPOSL** Uses factorization of complex positive definite matrix to solve systems.

**CPPCO** Uses Cholesky algorithm to factor complex positive definite matrix stored in packed form.

**CPPDI** Uses factorization of complex positive definite matrix stored in packed form to compute determinant and/or inverse.

**CPPFA** Uses Cholesky algorithm to factor complex positive definite matrix stored in packed form.

**CPPSL** Uses factorization of complex positive definite matrix stored in packed form to solve systems.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCDH** Compute the $R^H R$ factorization of a complex Hermitian positive definite matrix and estimate its $L_1$ condition number.

**LFIDH** Use iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations.

**LFSDH** Solve a complex Hermitian positive definite system of linear equations given the $R^H R$ factorization of the coefficient matrix.

**LFTDH** Compute the $R^H R$ factorization of a complex Hermitian positive definite matrix.

**LSADH** Solve a Hermitian positive definite system of linear equations with iterative refinement.

**LSLDH** Solve a complex Hermitian positive definite system of linear equations without iterative refinement.

*NAG Subprogram Library*

**F01BNF** Performs the Cholesky decomposition of a complex positive definite Hermitian matrix, given the lower triangle of the matrix.

**F01BPF** Determines the inverse of a complex positive definite Hermitian matrix, given the lower triangle of the matrix.

**F04AWF** Calculates the approximate solutions of a set of complex linear equations with multiple right hand sides, AX=B, where A is positive definite Hermitian, following the Cholesky decomposition of A by F01BNF.

| | |
|---|---|
| **D2d2 :** | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex Hermitian positive definite banded matrices)** |

*CMLIB Library (LINPAKC Sublibrary)*

**CPBCO** Uses Cholesky algorithm to compute factorization of complex positive definite band matrix and estimates its condition.

**CPBFA** Uses Cholesky algorithm to compute factorization of complex positive definite band matrix.

**CPBSL** Uses factorization of complex positive definite band matrix to solve systems.

*IMSL MATH/LIBRARY Subprogram Library*

**LFCQH** Compute the $R^H R$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode and estimate its $L_1$ condition number.

**LFIQH** Use iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations in band Hermitian storage mode.

**LFSQH** Solve a complex Hermitian positive definite system of linear equations given the factorization of the coefficient matrix in band Hermitian storage mode.

**LFTQH** Compute the $R^H R$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode.

**LSAQH** Solve a complex Hermitian positive definite system of linear equations in band Hermitian storage mode with iterative refinement.

**LSLQH** Solve a complex Hermitian positive definite system of linear equations in band Hermitian storage mode without iterative refinement.

| | |
|---|---|
| **D2d2a :** | **Solution of systems of linear equations, including matrix inverses, and LU and related factorizations (complex Hermitian tridiagonal matrices)** |

*CMLIB Library (LINPAKC Sublibrary)*

**CPTSL** Solves systems with complex positive definite tridiagonal matrix.

| | |
|---|---|
| **D2e :** | **Operations associated with the solution of systems of linear equations, including matrix inverses, and LU and related factorizations (e.g., matrix reorderings)** |

*Collected Algorithms of the ACM*

**A508** REDUCE: a Fortran subprogram for reducing the bandwidth and profile of sparse symmetric matrices using row and column permutations. (See H.L. Crane et al.,ACM TOMS 2 (1976) pp. 375-377.)

**A509** A hybrid algorithm for reducing the bandwidth and profile of sparse symmetric matrices. Presented as a modification of algorithm 508 (Fortran subprogram REDUCE). (See N.E. Gibbs, ACM TOMS 2 (1976) pp. 378-387.)

**A529** MC13D: a Fortran subroutine for finding symmetric permutations to block triangular form. That is, given the column numbers of the nonzeros in each row of a sparse matrix, this subroutine finds a symmetric permutation that makes the matrix block lower triangular. (See I.S. Duff and J.K. Reid, ACM TOMS 4 (1978) pp. 189-192.)

**A575** MC21A: a subroutine for finding a row permutation for a zero-free diagonal. That is, given the pattern of nonzeros of a sparse matrix, this routine attempts to find a permutation of its rows that makes the matrix have no zeros on its diagonal. (See I.S. Duff, ACM TOMS 7 (1981) pp. 387-390.)

A582 GPSKCA: Fortran subroutine for bandwidth or profile reduction of structurally symmetric sparse matrices. (See J.G. Lewis, ACM TOMS 8 (1982) pp. 190-194.)

A604 EXTREM: a Fortran subprogram for the calculating extremal polynomials. If L is a linear functional on polynomials of degree n or less, then p is extremal if it is a polynomial of (Chebyshev) norm one at which L takes on its norm. (See F.W. Sauer, ACM TOMS 9 (1983) pp. 381-383.)

# D3: Determinants

Although the determinant of a square matrix is a very useful tool in mathematical theory, its value is strongly dependent upon the scaling of matrix elements, and hence is an unreliable indicator of the numerical difficulties which might arise in trying to solve some problem. Casual users should exercise caution in using determinants in the formulation and solution of problems. Classified here are a number of modules which calculate determinants carefully; they are organized according to matrix type as in class D2.

## References


[DBMS79] J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart. *LINPACK User's Guide.* SIAM, Philadelphia, 1979.

[FM67] G. Forsythe and C. Moler. *Computer Solution of Linear Algebraic Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1967.


**D3a1 : Determinants of general real nonsymmetric matrices**

*CMLIB Library (LINPAKS Sublibrary)*

SGEDI Uses LU factorization of real general matrix to compute its determinant and/or inverse.

*IMSL Subprogram Library*

LINV3F In place inverse, equation solution, and/or determinant evaluation – full storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

LFDCG Compute the determinant of a complex general matrix given the LU factorization of the matrix.

LFDDS Compute the determinant of a real symmetric positive definite matrix given the $R^TR$ Cholesky factorization of the matrix.

LFDRB Compute the determinant of a real matrix in band storage mode given the LU factorization of the matrix.

LFDSF Compute the determinant of a real symmetric matrix given the $UDU^T$ factorization of the matrix.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

LFDRG Compute the determinant of a real general matrix given the LU factorization of the matrix.

*NAG Subprogram Library*

• F03AAF Calculates the determinant of a real matrix using the factorisation method of Crout.

F03AFF Decomposes a real matrix into triangular factor matrices LU by Crout's method and evaluates the determinant.

*Scientific Desk PC Subprogram Library*

**D2A13** Computes the determinant and/or inverse of a real matrix using the factors computed by D2A11.

**D3a2 : Determinants of real nonsymmetric banded matrices**

*CMLIB Library (LINDRV Sublibrary)*

**SNBDI** Computes the determinant of a single precision band matrix using factors previously computed.

*CMLIB Library (LINPAKS Sublibrary)*

**SGBDI** Uses LU factorization of real band matrix to compute its determinant. (No provision for computing matrix inverse.)

**D3a3 : Determinants of real nonsymmetric triangular matrices**

*CMLIB Library (LINPAKS Sublibrary)*

**STRDI** Computes determinant and/or inverse of real triangular matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**LFDRT** Compute the determinant of a real triangular matrix.

**D3b1a : Determinants of real symmetric indefinite matrices**

*CMLIB Library (LINPAKS Sublibrary)*

**SSIDI** Uses factorization of real symmetric indefinite matrix to compute its determinant and/or inverse.

**SSPDI** Uses factorization of real symmetric indefinite matrix stored in packed form to compute its determinant and/or inverse.

*Scientific Desk PC Subprogram Library*

**D2B1A3** Computes the determinant, inertia, or inverse of a real symmetric matrix using the factors from D2B1A1.

**D3b1b : Determinants of real symmetric positive definite matrices**

*CMLIB Library (LINPAKS Sublibrary)*

**SPODI** Uses factorization of real positive definite matrix to compute its determinant and/or inverse.

**SPPDI** Uses factorization of real positive definite matrix stored in packed form to compute its determinant and/or inverse.

*NAG Subprogram Library*

• **F03ABF** Calculates the determinant of a real symmetric positive definite matrix using Cholesky decomposition.

**F03AEF** Decomposes a real symmetric positive definite matrix A into triangular matrices $LL^T$ using Cholesky's method, and evaluates the determinant.

• **easy-to-use version**

*Scientific Desk PC Subprogram Library*

**D2B13** Finds the determinant and/or the inverse of a real symmetric positive definite matrix using the factorization obtained from D2B11.

## D3b2 : Determinants of real symmetric positive definite banded matrices

*CMLIB Library (LINPAKS Sublibrary)*

**SPBDI** Uses factorization of real positive definite band matrix to compute its determinant. (No provision for matrix inverse.)

*NAG Subprogram Library*

• **F03ACF** Calculates the determinant of a real symmetric positive definite band matrix using Cholesky decomposition.

**F03AGF** Decomposes a real symmetric positive definite band matrix into triangular matrices using Cholesky's method and evaluates the determinant.

## D3c1 : Determinants of general complex non-Hermitian matrices

*CMLIB Library (LINPAKC Sublibrary)*

**CGEDI** Compute determinant and/or inverse of general complex matrix from its LU factors.

*NAG Subprogram Library*

• **F03ADF** Calculates the determinant of a complex matrix using the factorisation method of Crout.

**F03AHF** Decomposes a complex matrix into triangular factor matrices LU by Crout's method and evaluates the determinant.

## D3c2 : Determinants of complex non-Hermitian banded matrices

*CMLIB Library (LINDRV Sublibrary)*

**CNBDI** Computes the determinant of a complex band matrix from previously computed factors.

*CMLIB Library (LINPAKC Sublibrary)*

**CGBDI** Compute determinant of complex band matrix from its LU factors. (No provision for computing inverse directly.)

*IMSL MATH/LIBRARY Subprogram Library*

**LFDCB** Compute the determinant of a complex matrix given the LU factorization of the matrix in band storage mode.

## D3c3 : Determinants of complex non-Hermitian triangular matrices

*CMLIB Library (LINPAKC Sublibrary)*

**CTRDI** Computes determinant and/or inverse of complex triangular matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**LFDCT** Compute the determinant of a complex triangular matrix.

| D3d1a : | Determinants of complex Hermitian indefinite matrices |
|---|---|

*CMLIB Library (LINPAKC Sublibrary)*

**CHIDI** Uses factorization of complex Hermitian indefinite matrix to compute its inertia, determinant, and/or inverse.

**CHPDI** Uses factorization of complex Hermitian indefinite matrix stored in packed form to compute its inertia, determinant, and inverse.

**CSIDI** Uses factorization of complex symmetric indefinite matrix to compute its determinant and/or inverse.

**CSPDI** Uses factorization of complex symmetric indefinite matrix stored in packed form to compute its determinant and/or inverse.

| D3d1b : | Determinants of complex Hermitian positive definite matrices |
|---|---|

*CMLIB Library (LINPAKC Sublibrary)*

**CPODI** Uses factorization of complex positive definite matrix to compute its determinant and/or inverse.

**CPPDI** Uses factorization of complex positive definite matrix stored in packed form to compute determinant and/or inverse.

*IMSL MATH/LIBRARY Subprogram Library*

**LFDDH** Compute the determinant of a complex Hermitian positive definite matrix given the $R^TR$ Cholesky factorization of the matrix.

*NAG Subprogram Library*

**F03AMF** Finds the determinant of a complex positive definite Hermitian matrix A, following a Cholesky decomposition of A as given by F01BNF.

| D3d2 : | Determinants of complex Hermitian positive definite banded matrices |
|---|---|

*CMLIB Library (LINPAKC Sublibrary)*

**CPBDI** Uses factorization of complex positive definite band matrix to compute determinant. (No provision for computing inverse.)

*IMSL MATH/LIBRARY Subprogram Library*

**LFDQH** Compute the determinant of a complex Hermitian positive definite matrix given the $R^TR$ Cholesky factorization in band Hermitian storage mode.

# D4: Eigenvalues, eigenvectors

The calculation of the eigenvalues and eigenvectors of a matrix has received much attention. It was for this class of problems that the first modern mathematical software package, EISPACK [SBD76,GBD77, SBD76,GBD77], was produced, and many of the routines in this class originated in the EISPACK project.

Most linear eigenvalue problems can be written in the form

$$Ax = \lambda Bx,$$

where the matrices $A$ and $B$ are given and appropriate scalars $\lambda$ (eigenvalues) and vectors $x$ (eigenvectors) are to be determined. The special case when $B$ is the identity matrix, $Ax = \lambda x$, is called the *ordinary* eigenvalue problem; otherwise the problem is referred to as a *generalized* eigenvalue problem. Software for the former is found in class D4a, while software for the latter is found in D4b. This software is classified according to the type of matrices $A$ and $B$, so one should determine as many special properties of these matrices as possible in order to find the best routine. In some cases one may also have a choice of computing only the first few (largest or smallest) eigenvalues, or the eigenvalues in a given range. The computation of eigenvectors is sometimes optional.

Class D4c contains routines for performing many of the intermediate steps in standard algorithms for computing eigenvalues and eigenvectors. These are generally not called directly by users.

## References

[GBD77] B. S. Garbow, J. M. Boyle, J. J. Dongarra, et al. *EISPACK Guide Extension.* Springer-Verlag, New York, 1977.

[GVL83] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 1983.

[Par80] B. N. Parlett. *The Symmetric Eigenvalue Problem.* Prentice-Hall, Englewood Cliffs, NJ, 1980.

[SBD76] B. T. Smith, J. M. Boyle, J. J. Dongarra, et al. *EISPACK Guide.* Springer-Verlag, New York, second edition, 1976.

[Wil65] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, London, 1965.

### D4a1 : Real symmetric matrix eigenvalue problems

*Collected Algorithms of the ACM*

**A530** TRIZD, IMZD, and TBAKZD: FORTRAN subroutines for computing the eigenvalues and eigenvectors of real skew-symmetric matrices or real tridiagonal symmetric matrices with constant diagonals. Based upon orthogonal similarity transformations. (See R.C. Ward and L.J. Gray, ACM TOMS 4 (1978) pp. 286-289.)

*CMLIB Library (EISPACK Sublibrary)*

**RS** Computes eigenvalues and, optionally, eigenvectors of a real symmetric matrix.

**RSP** Compute eigenvalues and, optionally, eigenvectors of a real symmetric matrix packed into a one dimensional array.

*CMLIB Library (LICEPAK Sublibrary)*

**SSIEV** Computes the eigenvalues and, optionally, the eigenvectors of a real symmetric matrix.

**SSPEV** Computes eigenvalues and, optionally eigenvectors of real symmetric matrix stored in packed form.

*IMSL Subprogram Library*

**EIGRS** Computes the eigenvalues and, optionally, the eigenvectors of a real symmetric matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**EVASF** Compute the largest or smallest eigenvalues of a real symmetric matrix.

**EVBSF** Compute the eigenvalues in a given range of a real symmetric matrix.

**EVESF** Compute the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix.

**EVFHF** Compute the eigenvalues in a given range and the corresponding eigenvectors of a complex Hermitian matrix.

**EVFSF** Compute the eigenvalues in a given range and the corresponding eigenvectors of a real symmetric matrix.

**EVLSF** Compute all of the eigenvalues of a real symmetric matrix.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**EVCSF** Compute all of the eigenvalues and eigenvectors of a real symmetric matrix.

*MAGEV Cyber 205 Subprogram Library*

**IMTQL** To find all eigenvalues and, optionally, all eigenvectors of a symmetric tridiagonal matrix. The complete eigensystem of a full symmetric matrix can also be determined, provided the matrix has first been made tridiagonal by calling HTRED2 or TRED2.

*NAG Subprogram Library*

**F02AAF** Calculates all the eigenvalues of a real symmetric matrix by Householder reduction and the QL algorithm.

**F02ABF** Calculates all the eigenvalues and eigenvectors of a real symmetric matrix by Householder reduction and the QL algorithm.

**F02BBF** Calculates selected eigenvalues and eigenvectors of a real symmetric matrix by Householder reduction, by the method of bisection and by inverse iteration, where the selected eigenvalues lie between two given values.

**F06BPF** Compute eigenvalue of 2-by-2 real symmetric matrix.

*NASHLIB Subprogram Library*

**A13** Use the singular value decomposition to find the eigenvalues and eigenvectors of a real symmetric matrix. Illustrates use of algorithm A13ESV. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 101-103.)

**A14A15** Use the Jacobi algorithm to find the eigenvalues and eigenvectors of a real symmetric matrix. Use the Jacobi algorithm to solve the generalized eigenvalue problem for real symmetric matrices. Illustrates use of algorithms A14JE and A15GSE. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 106-115.)

**A25** Use the conjugate gradient algorithm to find the minimum of a Rayleigh quotient. Illustrates use of algorithm A25RQM. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 206-207.)

*Scientific Desk PC Subprogram Library*

**D4A11** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real symmetric packed matrix.

*SCRUNCH Subprogram Library*

**SYMEIG** Computes the eigenvalues and, optionally, the eigenvectors of a real symmectric matrix. In BASIC.

| D4a2 : | **Real nonsymmetric matrix eigenvalue problems** |
|---|---|

*Collected Algorithms of the ACM*

**A530** TRIZD, IMZD, and TBAKZD: FORTRAN subroutines for computing the eigenvalues and eigenvectors of real skew-symmetric matrices or real tridiagonal symmetric matrices with constant diagonals. Based upon orthogonal similarity transformations. (See R.C. Ward and L.J. Gray, ACM TOMS 4 (1978) pp. 286-289.)

*CMLIB Library (EISPACK Sublibrary)*

**RG** Computes eigenvalues and, optionally, eigenvectors of a real general matrix.

*CMLIB Library (LICEPAK Sublibrary)*

**SGEEV** Computes the eigenvalues and, optionally, the eigenvectors of a general real matrix.

*IMSL Subprogram Library*

**EIGRF** Eigenvalues and (optionally) eigenvectors of a real general matrix in full storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

**EVCRG** Compute all of the eigenvalues and eigenvectors of a real matrix.

**EVLRG** Compute all of the eigenvalues of a real matrix.

*NAG Subprogram Library*

**F02AFF** Calculates all the eigenvalues of a real unsymmetric matrix by reduction to Hessenberg form and the QR algorithm.

**F02AGF** Calculates all the eigenvalues and eigenvectors of a real unsymmetric matrix by reduction to Hessenberg form and the QR algorithm.

**F02BCF** Calculates selected eigenvalues and eigenvectors of a real unsymmetric matrix by reduction to Hessenberg form, the QR algorithm and inverse iteration, where the moduli of the selected eigenvalues lie between two given values.

*NASHLIB Subprogram Library*

**A10** Use inverse interation via Gaussian elimination. Illustrates use of algorithm A10GII. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 90-91.)

**A11A12** To standardize a complex vector and compute the residuals of a complex eigenvalue and eigenvector. Illustrates use of algorithms A11VS and A12CVR. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 94-95.)

**A9** Use modification of Gauss Jordan algorithm to perform the Bauer Reinsch inversion of positive definite symmetric matrix. Illustrates use of algorithm A9GJ. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, page 84.)

*PORT Subprogram Library*

**EIGEN** Finds all eigenvalues and eigenvectors of a real matrix. Output consists of pairs of real arrays.

*Scientific Desk PC Subprogram Library*

**D** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real square (general) matrix. Orthogonal transformations are used. Real and imaginary components of output values are returned in separate real arrays. Estimates of the number of correct significant digits in the eigenvalue estimates can also be obtained.

**D4A21** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real square (general) matrix. Elementary transformations are used. Real and imaginary components of output values are returned in separate real arrays.

**D4A22D** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real square (general) matrix. Orthogonal transformations are used. Real and imaginary components of output values are returned in separate real arrays. Estimates of the number of correct significant digits in the eigenvalue estimates can also be obtained.

**D4C2C** Finds the eigenvalues and eigenvectors of a real upper Hessenberg matrix by the QR method. The eigenvectors of a real general matrix can also be found if D4C1B and D4C23 have been used to reduce this general matrix to upper Hessenberg form and to accumulate the similarity transformations.

## D4a3 : Complex Hermitian matrix eigenvalue problems

*CMLIB Library (EISPACK Sublibrary)*

**CH** Computes the eigenvalues and, optionally, eigenvectors of a complex Hermitian matrix.

*CMLIB Library (LICEPAK Sublibrary)*

**CHIEV** Computes the eigenvalues and, optionally, the eigenvectors of a complex Hermitian matrix.

*IMSL Subprogram Library*

**EIGCH** Eigenvalues and (optionally) eigenvectors of a complex Hermitian matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**EVAHF** Compute the largest or smallest eigenvalues of a complex Hermitian matrix.

**EVBHF** Compute the eigenvalues in a given range of a complex Hermitian matrix.

**EVCHF** Compute all of the eigenvalues and eigenvectors of a complex Hermitian matrix.

**EVEHF** Compute the largest or smallest eigenvalues and the corresponding eigenvectors of a complex Hermitian matrix.

**EVLHF** Compute all of the eigenvalues of a complex Hermitian matrix.

*NAG Subprogram Library*

**F02AWF** Calculates all the eigenvalues of a complex Hermitian matrix by reduction to real symmetric tridiagonal form and the QL algorithm.

**F02AXF** Calculates all the eigenvalues and eigenvectors of a complex Hermitian matrix by reduction to real symmetric tridiagonal form and the QL algorithm.

## D4a4 : Complex non-Hermitian matrix eigenvalue problems

*CMLIB Library (EISPACK Sublibrary)*

**CG** Computes the eigenvalues and, optionally, the eigenvectors of a complex general matrix.

*CMLIB Library (LICEPAK Sublibrary)*

**CGEEV** Computes the eigenvalues and, optionally, the eigenvectors of a general complex matrix.

*IMSL Subprogram Library*

**EIGCC** Eigenvalues and (optionally) eigenvectors of a complex general matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**EVCCG** Compute all of the eigenvalues and eigenvectors of a complex matrix.

**EVLCG** Compute all of the eigenvalues of a complex matrix.

*NAG Subprogram Library*

**F02AJF** Calculates all the eigenvalues of a complex matrix by reduction to upper Hessenberg form and the LR algorithm.

**F02AKF** Calculates all the eigenvalues and eigenvectors of a complex matrix by reduction to upper Hessenberg form and the LR algorithm.

**F02BDF** Calculates selected eigenvalues and eigenvectors of a complex matrix by reduction to Hessenberg form, the LR algorithm and inverse iteration, where the moduli of the selected eigenvalues lie between two given values.

## D4a5 : Tridiagonal matrix eigenvalue problems

*Collected Algorithms of the ACM*

**A530** TRIZD, IMZD, and TBAKZD: FORTRAN subroutines for computing the eigenvalues and eigenvectors of real skew-symmetric matrices or real tridiagonal symmetric matrices with constant diagonals. Based upon orthogonal similarity transformations. (See R.C. Ward and L.J. Gray, ACM TOMS 4 (1978) pp. 286-289.)

*CMLIB Library (EISPACK Sublibrary)*

**BISECT** Compute eigenvalues of symmetric tridiagonal matrix in given interval using Sturm sequencing.

**IMTQL1** Computes eigenvalues of symmetric tridiagonal matrix using implicit QL method.

**IMTQL2** Computes eigenvalues and eigenvectors of symmetric tridiagonal matrix using implicit QL method.

**IMTQLV** Computes eigenvalues of symmetric tridiagonal matrix using implicit QL method. Eigenvectors may be computed later.

**RATQR** Computes largest or smallest eigenvalues of symmetric tridiagonal matrix using rational QR method with Newton correction.

• **RST** Compute eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix.

**RT** Compute eigenvalues and eigenvectors of a special real tridiagonal matrix.

**TQL1** Compute eigenvalues of symmetric tridiagonal matrix by QL method.

**TQL2** Compute eigenvalues and eigenvectors of symmetric tridiagonal matrix.

**TQLRAT** Computes eigenvalues of symmetric tridiagonal matrix using a rational variant of the QL method.

**TRIDIB** Computes eigenvalues of symmetric tridiagonal matrix in given interval using Sturm sequencing.

**TSTURM** Computes eigenvalues of symmetric tridiagonal matrix in given interval and eigenvectors by Sturm sequencing.

*IMSL Subprogram Library*

**EQRT1S** Smallest or largest m eigenvalues of a symmetric tridiagonal matrix.

**EQRT2S** Eigenvalues and (optionally) eigenvectors of a symmetric tridiagonal matrix using the QL method.

**EQRT3S** The smallest (or largest) eigenvalues of a tridiagonal matrix in algebraic value whose sum exceeds a given value.

*MAGEV Cyber 205 Subprogram Library*

**IMTQL** To find all eigenvalues and, optionally, all eigenvectors of a symmetric tridiagonal matrix. The complete eigensystem of a full symmetric matrix can also be determined, provided the matrix has first been made tridiagonal by calling HTRED2 or TRED2.

*NAG Subprogram Library*

**F02AVF** Calculates all the eigenvalues of a real symmetric tridiagonal matrix.

**F02BEF** Calculates selected eigenvalues and eigenvectors of a real symmetric tridiagonal matrix, where the selected eigenvalues lie between two given values.

**F02BFF** Calculates selected eigenvalues of a real symmetric tridiagonal matrix, where, if the eigenvalues are numbered in ascending order, the numbers of the first and last eigenvalues required are given.

*Scientific Desk PC Subprogram Library*

**D4C2A** Finds the eigenvalues and eigenvectors of a symmetric tridiagonal matrix by the QL method.

## D4a6 : Banded matrix eigenvalue problems

*CMLIB Library (EISPACK Sublibrary)*

**BQR** Computes some of the eigenvalues of a real symmetric band matrix using the QR method with shifts of origin.

**RSB** Computes eigenvalues and, optionally, eigenvectors of real symmetric band matrix.

*IMSL Subprogram Library*

**EIGBS** Find some eigenvalues and (optionally) eigenvectors of a real symmetric band matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**EVASB** Compute the largest or smallest eigenvalues of a real symmetric matrix in band symmetric storage mode.

**EVBSB** Compute the eigenvalues in a given range of a real symmetric matrix stored in band symmetric storage mode.

**EVCSB** Compute all of the eigenvalues and eigenvectors of a real symmetric matrix in band symmetric storage mode.

**EVESB** Compute the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix in band symmetric storage mode.

**EVFSB** Compute the eigenvalues in a given range and the corresponding eigenvectors of a real symmetric matrix stored in band symmetric storage mode.

**EVLSB** Compute all of the eigenvalues of a real symmetric matrix in band symmetric storage mode.

## D4a7 : Sparse matrix eigenvalue problems

*Collected Algorithms of the ACM*

**A570** LOPSI: a Fortran subprogram which determines approximations to right or left eigenvectors corresponding to the dominant set of eigenvalues of a real unsymmetric matrix using the method of simultaneous iteration. (See W.J. Stewart and A. Jennings, ACM TOMS 7 (1981) pp. 230-232.)

*NAG Subprogram Library*

**F02FJF** Finds eigenvalues and eigenvectors of a real sparse symmetric or generalised symmetric eigenvalue problem.

| D4b1 : | Real symmetric generalized matrix eigenvalue problems |
|---|---|

*Collected Algorithms of the ACM*

**A538** SIMITZ: a Fortran subroutine for computing eigenvalues largest in magnitude and corresponding eigenvectors of a real matrix symmetric relative to a user-defined inner product. Based upon the simultaneous iteration algorithm. (See P.J. Nikolai, ACM TOMS 5 (1979) pp. 118-125.)

**A646** PDFIND: a Fortran subprogram for finding a positive definite linear combination of two real symmetric matrices. The algorithm is independent of the data structure used to store the matrices. The combination may be used to solve the generalized eigenproblem $Ax = \lambda Bx$ in case A and B are large and sparse, but neither is positive definite. (See C.R. Crawford, ACM TOMS 12 (1986) pp. 278-282.)

*CMLIB Library (EISPACK Sublibrary)*

**RSG** Computes eigenvalues and, optionally, eigenvectors of real symmetric generalized eigenproblem: $Ax=\lambda Bx$.

**RSGAB** Computes eigenvalues and, optionally, eigenvectors of real symmetric generalized eigenproblem: $ABx=\lambda x$.

**RSGBA** Computes eigenvalues and, optionally, eigenvectors of real symmetric generalized eigenproblem: $BAx=\lambda x$.

*IMSL Subprogram Library*

**EIGZS** Eigenvalues and (optionally) eigenvectors of the system $Ax = \lambda Bx$ where A and B are real symmetric matrices and B is positive definite.

*IMSL MATH/LIBRARY Subprogram Library*

**GVCSP** Compute all of the eigenvalues and eigenvectors of the generalized real symmetric eigenvalue problem $Az = \lambda Bz$, with B symmetric positive definite.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**GVLSP** Compute all of the eigenvalues of the generalized real symmetric eigenvalue problem $Az = \lambda Bz$, with B symmetric positive definite.

*NAG Subprogram Library*

**F02ADF** Calculates all the eigenvalues of $Ax = \lambda Bx$, where A is a real symmetric matrix and B is a real symmetric positive definite matrix, using Householder reduction and the QL algorithm.

**F02AEF** Calculates all the eigenvalues and eigenvectors of $Ax = \lambda Bx$, where A is a real symmetric matrix and B is a real symmetric positive definite matrix, using Householder reduction and the QL algorithm.

**F02FJF** Finds eigenvalues and eigenvectors of a real sparse symmetric or generalised symmetric eigenvalue problem.

*NASHLIB Subprogram Library*

**A14A15** Use the Jacobi algorithm to find the eigenvalues and eigenvectors of a real symmetric matrix. Use the Jacobi algorithm to solve the generalized eigenvalue problem for real symmetric matrices. Illustrates use of algorithms A14JE and A15GSE. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 106-115.)

## D4b2 : Real generalized matrix eigenvalue problems

*CMLIB Library (EISPACK Sublibrary)*

**RGG** Computes eigenvalues and eigenvectors for real generalized eigenproblem: $Ax=\lambda Bx$.

*IMSL Subprogram Library*

**EIGZF** Eigenvalues and (optionally) eigenvectors of the system $Ax=\lambda Bx$ where A and B are real matrices.

*IMSL MATH/LIBRARY Subprogram Library*

**GVCRG** Compute all of the eigenvalues and eigenvectors of a generalized real eigensystem $Az = \lambda Bz$.

**GVLRG** Compute all of the eigenvalues of a generalized real eigensystem $Az = \lambda Bz$.

*NAG Subprogram Library*

**F02BJF** Calculates all the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax = \lambda Bx$ where A and B are real, square matrices, using the QZ algorithm.

## D4b4 : Complex generalized matrix eigenvalue problems

*Collected Algorithms of the ACM*

**A496** LZHES/LZIT: subprograms for finding all the eigenvalues and (optionally) eigenvectors of the generalized eigenvalue problem for complex matrices. Based upon the LZ algorithm. (See L.C. Kaufman, ACM TOMS 1 (1975) pp. 271-281.)

**A535** CQZHES, CQSVEC, and CQZVAL: Fortran subroutines implementing the QZ algorithm for solving the generalized eigenvalue problem for complex matrices. (See B.S. Garbow, ACM TOMS 4 (1978) pp. 404-410.)

*IMSL Subprogram Library*

**EIGZC** Eigenvalues and (optionally) eigenvectors of the system $Ax=\lambda Bx$ where A and B are complex matrices.

*IMSL MATH/LIBRARY Subprogram Library*

**GVCCG** Compute all of the eigenvalues and eigenvectors of a generalized complex eigensystem $Az = \lambda Bz$.

**GVLCG** Compute all of the eigenvalues of a generalized complex eigensystem $Az = \lambda Bz$.

*NAG Subprogram Library*

**F02GJF** Calculates all the eigenvalues and, if required, all the eigenvectors of the complex generalized eigenproblem $Ax = \lambda Bx$ where A and B are complex, square matrices, using the QZ algorithm.

## D4b5 : Banded generalized matrix eigenvalue problems

*NAG Subprogram Library*

**F02FHF** Finds the eigenvalues of the generalised band symmetric eigenvalue problem $Ax = \lambda Bx$, where A and B are symmetric band matrices and B is positive definite.

**F02SDF** Finds the eigenvector corresponding to a given real eigenvalue for the generalised problem $Ax = \lambda Bx$, or for the standard problem $Ax = \lambda x$, where A and B are real band matrices.

## D4c : Operations associated with matrix eigenvalue problems

*Collected Algorithms of the ACM*

**A589** SICEDR: a Fortran subroutine for improving the accuracy of computed real matrix eigenvalues and improving or computing the associated eigenvector. (See J.J. Dongarra, ACM TOMS 8 (1982) pp. 371-375.)

*IMSL MATH/LIBRARY Subprogram Library*

**EPICG** Compute the performance index for a complex eigensystem.

**EPIHF** Compute the performance index for a complex Hermitian eigensystem.

**EPIRG** Compute the performance index for a real eigensystem.

**EPISB** Compute the performance index for a real symmetric eigensystem in band symmetric storage mode.

**GPICG** Compute the performance index for a generalized complex eigensystem $Az = \lambda Bz$.

**GPIRG** Compute the performance index for a generalized real eigensystem $Az = \lambda Bz$.

**GPISP** Compute the performance index for a generalized real symmetric eigensystem problem.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**EPISF** Compute the performance index for a real symmetric eigensystem.

## D4c1a : Balance matrix for eigenvalue problem

*CMLIB Library (EISPACK Sublibrary)*

**BALANC** Balances a general real matrix and isolates eigenvalues whenever possible.

**CBAL** Balances a complex general matrix and isolates eigenvalues whenever possible.

*IMSL Subprogram Library*

**EBALAC** Balance a complex general matrix and isolate eigenvalues whenever possible.

**EBALAF** Balance a real matrix.

*NAG Subprogram Library*

**F01ATF** Balances a real unsymmetric matrix.

**F01AVF** Balances a complex matrix.

*Scientific Desk PC Subprogram Library*

**D4C1A** Balances a real matrix and isolates eigenvalues whenever possible.

## D4c1b1 : Reduce matrix to tridiagonal form

*CMLIB Library (EISPACK Sublibrary)*

**BANDR** Reduces real symmetric band matrix to symmetric tridiagonal matrix and, optionally, accumulates orthogonal similarity transformations.

**HTRID3** Reduces complex Hermitian (packed) matrix to real symmetric tridiagonal matrix by unitary similarity transformations.

**HTRIDI** Reduces complex Hermitian matrix to real symmetric tridiagonal matrix using unitary similarity transformations.

**TRED1** Reduce real symmetric matrix to symmetric tridiagonal matrix using orthogonal similarity transformations.

**TRED2** Reduce real symmetric matrix to symmetric tridiagonal matrix using and accumulating orthogonal transformations.

**TRED3** Reduce real symmetric matrix stored in packed form to symmetric tridiagonal matrix using orthogonal transformations.

*IMSL Subprogram Library*

**EHOUSH** Reduction of a complex Hermitian matrix to real symmetric tridiagonal form.

**EHOUSS** Reduction of a symmetric matrix to symmetric tridiagonal form using a Householder reduction.

*MAGEV Cyber 205 Subprogram Library*

**TRED2** To reduce a symmetric matrix to a symmetric tridiagonal matrix. This is generally the first step in obtaining the complete eigensystem of the matrix.

*NAG Subprogram Library*

**F01AGF** Gives the Householder reduction of a real symmetric matrix to tridiagonal form for use with F02BEF, F02AVF and F02BFF.

**F01AJF** Gives the Householder reduction of a real symmetric matrix A to tridiagonal form for use in F02AMF.

**F01AYF** Gives the Householder reduction of a real symmetric matrix A to tridiagonal form for use in F02BEF, F02AVF and F02BFF. The routine is similar to F01AGF but is more economical in storage.

**F01BCF** Gives the Householder reduction of a complex Hermitian matrix to tridiagonal form for use in F02AVF or F02AYF.

**F01BWF** Reduces a symmetric band matrix to tridiagonal form. This routine will normally be used in conjunction with F02AVF to find all the eigenvalues of A. For selected eigenvalues there is a choice between F02BMF and the combination of F01BWF and F02BFF.

*Scientific Desk PC Subprogram Library*

**D4C1D** Reduces a real general symmetric matrix, stored as a one-dimensional array, to a symmetric tridiagonal matrix using orthogonal transformations.

## D4c1b2 : Reduce matrix to Hessenberg form

*CMLIB Library (EISPACK Sublibrary)*

**COMHES** Reduces complex general matrix to complex upper Hessenberg form using stabilized elementary similarity transformations.

**CORTH** Reduces complex general matrix to complex upper Hessenberg using unitary similarity transformations.

**ELMHES** Reduces real general matrix to upper Hessenberg form using stabilized elementary similarity transformations.

**ORTHES** Reduces real general matrix to upper Hessenberg form using orthogonal similarity transformations.

*IMSL Subprogram Library*

**EHESSC** Reduction of a general complex matrix to complex upper Hessenberg form.

**EHESSF** Reduction of a nonsymmetric matrix to upper Hessenberg form by orthogonal transformations.

*NAG Subprogram Library*

**F01AKF** Reduces a real unsymmetric matrix to upper Hessenberg form.

**F01AMF** Reduces a complex unsymmetric matrix to complex upper Hessenberg form.

*Scientific Desk PC Subprogram Library*

**D4C1B** Given a real general matrix, this subroutine reduces a submatrix situated in rows and columns LOW through IGH to upper Hessenberg form by stabilized elementary similarity transformations.

**D4C1C** Reduces a real general matrix to upper Hessenberg form using orthogonal transformations.

**D4C23** Accumulates the stabilized elementary similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by D4C1B.

**D4C24** Accumulates the orthogonal similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by D4C1C.

## D4c1b3 : Reduce matrix to other compact forms

*Collected Algorithms of the ACM*

**A589** SICEDR: a Fortran subroutine for improving the accuracy of computed real matrix eigenvalues and improving or computing the associated eigenvector. (See J.J. Dongarra, ACM TOMS 8 (1982) pp. 371-375.)

**A590** DSUBSP and EXCHQZ: Fortran subroutines for computing deflating subspaces with specified spectrum. (See P. Van Dooren, ACM TOMS 8 (1982) pp. 376-382.)

*CMLIB Library (EISPACK Sublibrary)*

**QZHES** The first step of the QZ algorithm for solving generalized matrix eigenproblems. Accepts a pair of real general matrices and reduces one of them to upper Hessenberg form and the other to upper triangular form using orthogonal transformations. Usually followed by QZIT, QZVAL, QZVEC.

**QZIT** The second step of the QZ algorithm for generalized eigenproblems. Accepts an upper Hessenberg and an upper triangular matrix and reduces the former to quasi-triangular form while preserving the form of the latter. Usually preceded by QZHES and followed by QZVAL and QZVEC.

## D4c1c : Standardize matrix eigenvalue problem

*Collected Algorithms of the ACM*

**A646** PDFIND: a Fortran subprogram for finding a positive definite linear combination of two real symmetric matrices. The algorithm is independent of the data structure used to store the matrices. The combination may be used to solve the generalized eigenproblem $Ax = \lambda Bx$ in case A and B are large and sparse, but neither is positive definite. (See C.R. Crawford, ACM TOMS 12 (1986) pp. 278-282.)

*CMLIB Library (EISPACK Sublibrary)*

**FIGI** Transforms certain real non-symmetric tridiagonal matrix to symmetric tridiagonal matrix.

**FIGI2** Transforms certain real non-symmetric tridiagonal matrix to symmetric tridiagonal matrix.

**REDUC** Reduces generalized symmetric eigenproblem $Ax=\lambda Bx$, to standard symmetric eigenproblem, using Cholesky factorization.

**REDUC2** Reduces certain generalized symmetric eigenproblems to standard symmetric eigenproblem, using Cholesky factorization.

*NAG Subprogram Library*

**F01AEF** Reduces the generalised eigenproblem $Ax = \lambda Bx$ to the standard symmetric eigenproblem $Pz = \lambda z$, where A is a real symmetric matrix and B is a real symmetric positive definite matrix.

**F01BDF** Reduces the eigenproblems $ABx = \lambda x$, $x^T BA = \lambda x^T$, $BAy = \lambda\, y$ and $y^T AB = \lambda y^T$ to the standard symmetric eigenproblem $Qz = \lambda z$, where A is a real symmetric matrix and B is a real symmetric positive definite matrix.

**F01BVF** Transforms the generalised symmetric eigenproblem $Ax = \lambda Bx$ to the equivalent standard eigenproblem $Cy = \lambda y$ where A, B and C are symmetric band matrices and B is positive definite. B must have been decomposed by F01BUF.

## D4c2a : Compute eigenvalues of tridiagonal matrix

*CMLIB Library (EISPACK Sublibrary)*

**BISECT** Compute eigenvalues of symmetric tridiagonal matrix in given interval using Sturm sequencing.

**IMTQL1** Computes eigenvalues of symmetric tridiagonal matrix using implicit QL method.

**IMTQL2** Computes eigenvalues and eigenvectors of symmetric tridiagonal matrix using implicit QL method.

**IMTQLV** Computes eigenvalues of symmetric tridiagonal matrix using implicit QL method. Eigenvectors may be computed later.

**RATQR** Computes largest or smallest eigenvalues of symmetric tridiagonal matrix using rational QR method with Newton correction.

**TQL1** Compute eigenvalues of symmetric tridiagonal matrix by QL method.

**TQL2** Compute eigenvalues and eigenvectors of symmetric tridiagonal matrix.

**TQLRAT** Computes eigenvalues of symmetric tridiagonal matrix using a rational variant of the QL method.

**TRIDIB** Computes eigenvalues of symmetric tridiagonal matrix in given interval using Sturm sequencing.

**TSTURM** Computes eigenvalues of symmetric tridiagonal matrix in given interval and eigenvectors by Sturm sequencing.

*NAG Subprogram Library*

**F02AMF** Calculates all the eigenvalues and eigenvectors of a real symmetric tridiagonal matrix or of a full real symmetric matrix that has been reduced to tridiagonal form using F01AJF.

**F02AYF** Calculates all the eigenvalues and eigenvectors of a complex Hermitian matrix which has been reduced to real symmetric tridiagonal form using F01BCF.

## D4c2b : Compute eigenvalues of Hessenberg matrix

*Collected Algorithms of the ACM*

**A506** HQR3: a Fortran subprogram which reduces an upper Hessenberg matrix to quasi-triangular form by unitary similarity transformations to yield the eigenvalues of the matrix in order of magnitude. HQR3 can be used with EISPACK routines ORTHES and ORTRAN to compute the eigenvalues of a real general matrix. (See G.W. Stewart, ACM TOMS 2 (1976) pp. 275-280.)

**A517** CONDIT and QR2NOZ: a Fortran subroutines for computing the condition numbers of matrix eigenvalues without computing eigenvectors. (See S.P. Chan, R. Feldman, and B.N. Parlett, ACM TOMS 3 (1977) pp. 186-203.)

*CMLIB Library (EISPACK Sublibrary)*

**CINVIT** Computes eigenvectors of a complex upper Hessenberg matrix associated with specified eigenvalues using inverse iteration.

**COMLR** Computes eigenvalues of a complex upper Hessenberg matrix using the modified LR method.

**COMLR2** Computes eigenvalues and eigenvectors of complex upper Hessenberg matrix using modified LR method.

**COMQR** Computes eigenvalues of complex upper Hessenberg matrix using the QR method.

**COMQR2** Computes eigenvalues and eigenvectors of complex upper Hessenberg matrix.

**HQR** Computes eigenvalues of a real upper Hessenberg matrix using the QR method.

**HQR2** Computes eigenvalues and eigenvectors of real upper Hessenberg matrix using QR method.

**INVIT** Computes eigenvectors of upper Hessenberg (real) matrix associated with specified eigenvalues by inverse iteration.

*IMSL MATH/LIBRARY Subprogram Library*

**EVCCH** Compute all of the eigenvalues and eigenvectors of a complex upper Hessenberg matrix.

**EVCRH** Compute all of the eigenvalues and eigenvectors of a real upper Hessenberg matrix.

**EVLCH** Compute all of the eigenvalues of a complex upper Hessenberg matrix.

**EVLRH** Compute all of the eigenvalues of a real upper Hessenberg matrix.

*NAG Subprogram Library*

**F02ANF** Calculates all the eigenvalues of a complex upper Hessenberg matrix.

**F02APF** Calculates all the eigenvalues of a real upper Hessenberg matrix.

**F02AQF** Calculates all the eigenvalues and eigenvectors of a real upper Hessenberg matrix or a real unsymmetric matrix which has been reduced to upper Hessenberg form by F01AKF and F01APF and which may have been balanced using F01ATF.

**F02ARF** Calculates all the eigenvalues and eigenvectors of either a complex upper Hessenberg matrix or a complex full matrix which has been reduced to upper Hessenberg form by F01AMF and may have been balanced using F01AVF.

*Scientific Desk PC Subprogram Library*

**D4C2B** Finds the eigenvalues of a real upper Hessenberg matrix by the QR method.

**D4C2C** Finds the eigenvalues and eigenvectors of a real upper Hessenberg matrix by the QR method. The eigenvectors of a real general matrix can also be found if D4C1B and D4C23 have been used to reduce this general matrix to upper Hessenberg form and to accumulate the similarity transformations.

## D4c2c : Compute eigenvalues of matrices in other compact forms

*Collected Algorithms of the ACM*

**A589** SICEDR: a Fortran subroutine for improving the accuracy of computed real matrix eigenvalues and improving or computing the associated eigenvector. (See J.J. Dongarra, ACM TOMS 8 (1982) pp. 371-375.)

*CMLIB Library (EISPACK Sublibrary)*

**QZVAL** The third step of the QZ algorithm for generalized eigenproblems. Accepts a pair of real matrices, one in quasi-triangular form and the other in upper triangular form and computes the eigenvalues of the associated eigenproblem. Usually preceded by QZHES, QZIT, and followed by QZVEC.

## D4c3 : Form matrix eigenvectors from eigenvalues

*CMLIB Library (EISPACK Sublibrary)*

**BANDV** Forms eigenvectors of real symmetric band matrix associated with a set of ordered approximate eigenvalues by inverse iteration.

**QZVEC** The optional fourth step of the QZ algorithm for generalized eigenproblems. Accepts a matrix in quasi-triangular form and another in upper triangular form and computes the eigenvectors of the triangular problem and transforms them back to the original coordinates. Usually preceded by QZHES, QZIT, QZVAL.

**TINVIT** Eigenvectors of symmetric tridiagonal matrix corresponding to some specified eigenvalues, using inverse iteration.

*NAG Subprogram Library*

**F02BKF** Calculates selected eigenvectors of a real upper Hessenberg matrix.

**F02BLF** Calculates selected eigenvectors of a complex upper Hessenberg matrix.

## D4c4 : Back transform matrix eigenvectors

*CMLIB Library (EISPACK Sublibrary)*

**BAKVEC** Forms eigenvectors of certain real non-symmetric tridiagonal matrix from symmetric tridiagonal matrix output from FIGI.

**BALBAK** Forms eigenvectors of real general matrix from eigenvectors of matrix output from BALANC.

**CBABK2** Forms eigenvectors of complex general matrix from eigenvectors of matrix output from CBAL.

**COMBAK** Forms eigenvectors of complex general matrix from eigenvectors of upper Hessenberg matrix output from COMHES.

**CORTB** Forms eigenvectors of complex general matrix from eigenvectors of upper Hessenberg matrix output from CORTH.

**ELMBAK** Forms eigenvectors of real general matrix from eigenvectors of upper Hessenberg matrix output from ELMHES.

**ELTRAN** Accumulates the stabilized elementary similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by ELMHES.

**HTRIB3** Computes eigenvectors of complex Hermitian matrix from eigenvectors of real symmetric tridiagonal matrix output from HTRID3.

**HTRIBK** Forms eigenvectors of complex Hermitian matrix from eigenvectors of real symmetric tridiagonal matrix output from HTRIDI.

**ORTBAK** Forms eigenvectors of general real matrix from eigenvectors of upper Hessenberg matrix output from ORTHES.

**ORTRAN** Accumulates orthogonal similarity transformations in reduction of real general matrix by ORTHES.

**REBAK** Forms eigenvectors of generalized symmetric eigensystem from eigenvectors of derived matrix output from REDUC or REDUC2.

**REBAKB** Forms eigenvectors of generalized symmetric eigensystem from eigenvectors of derived matrix output from REDUC2.

**TRBAK1** Forms the eigenvectors of real symmetric matrix from eigenvectors of symmetric tridiagonal matrix formed by TRED1.

**TRBAK3** Forms eigenvectors of real symmetric matrix from the eigenvectors of symmetric tridiagonal matrix formed by TRED3.

*NAG Subprogram Library*

**F01AFF** Derives eigenvectors x of the eigenproblems $Ax = \lambda Bx$, $ABx = \lambda x$ and $yBA = \lambda y$, where y is the transpose of x, from the corresponding eigenvectors $z = Mx$, where M is the transpose of the lower triangular matrix L of the derived standard symmetric eigenproblems. Matrices A and B are real and symmetric. In addition B is positive definite.

**F01AHF** Derives eigenvectors of a real symmetric matrix from eigenvectors of the tridiagonal form where the tridiagonal matrix was produced by F01AGF.

**F01ALF** Transforms eigenvectors of a Hessenberg matrix to those of a real unsymmetric matrix from which the Hessenberg matrix has previously been derived.

**F01ANF** Transforms eigenvectors of a complex upper Hessenberg matrix to those of a complex unsymmetric matrix from which the Hessenberg matrix has previously been derived.

**F01APF** Forms the matrix of accumulated transformations from information left by F01AKF.

**F01AUF** Transforms eigenvectors of a balanced matrix to those of the original real unsymmetric matrix.

**F01AWF** Transforms eigenvectors of a balanced matrix to those of the original complex matrix.

**F01AZF** Derives eigenvectors of a real symmetric matrix from eigenvectors of the tridiagonal form produced by F01AYF.

**F01BEF** Derives eigenvectors y of the problems $y^TAB = \lambda y^T$ and $BAy = \lambda y$ from the corresponding eigenvectors of the derived standard symmetric eigenproblem.

*Scientific Desk PC Subprogram Library*

**D4C4** Forms the eigenvectors of a real symmetric matrix by back transforming those of the corresponding symmetric tridiagonal matrix determined by D4C1D.

**D4C4B** Forms the eigenvectors of a real general matrix by back transforming those of the corresponding balanced matrix determined by D4C1A.

## D4c5 : Determine Jordan normal form of a matrix

*Collected Algorithms of the ACM*

**A560** JNF: a Fortran subroutine for computing the Jordan normal form of a complex square matrix. (See B. Kagstrom and A. Ruhe, ACM TOMS 6 (1980) pp. 437-443.)

## D5 : *QR* decomposition, Gram-Schmidt orthogonalization

*CMLIB Library (LINPAKC Sublibrary)*

**CQRDC** Computes QR decomposition of general complex matrix.

*CMLIB Library (LINPAKS Sublibrary)*

**SQRDC** Computes QR decomposition of real general matrix.

*CMLIB Library (SQRLSS Sublibrary)*

**SQRANK** For solving linear systems in least squares sense. Computes the QR decomposition of matrix using LINPACK subroutines.

*IMSL MATH/LIBRARY Subprogram Library*

**LQERR** Accumulate the orthogonal matrix Q from its factored form given the QR factorization of a rectangular matrix A.

**LQRRR** Compute the QR decomposition using Householder transformations.

*NAG Subprogram Library*

**F01AXF** Reduces an m-by-n real matrix, m≥n, to upper triangular form for use in F04AMF and F04ANF. The routine uses Householder transformations with column pivoting.

**F01QAF** Factorises a real matrix A as QU where Q is an m-by-m orthogonal matrix, and U is an m-by-n matrix which is zero except for the elements on and above the leading diagonal.

**F01QBF** Factorises a real m-by-n (m≤n) matrix A as UQ where Q is an n-by-n orthogonal matrix, and U is an m-by-n matrix which is zero except for the elements on and above the leading diagonal in the first m columns.

**F05AAF** Applies the Schmidt orthogonalisation process to n vectors in m dimensional space, n≤m.

*NASHLIB Subprogram Library*

**A3** Givens reduction of a real rectangular matrix. Illustrates use of algorithm A3GR. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 42-43.)

## D6 : Singular value decomposition (SVD)

*Collected Algorithms of the ACM*

**A581** HYBSVD, MGNSVD, and GRSVD: Fortran subprograms for computing the singular value decomposition of a general rectangular matrix. The algorithm is a modification of the Golub-Reinsch procedure that is more efficient when the matrix has more rows than columns. (see T. F. Chan, ACM TOMS 8 (1982) pp. 84-88.)

*CMLIB Library (EISPACK Sublibrary)*

**MINFIT** Compute singular value decomposition of rectangular real matrix and solve related linear least squares problem.

**SVD** Computes singular value decomposition of arbitrary real rectangular matrix.

*CMLIB Library (LINPAKC Sublibrary)*

**CSVDC** Computes Singular Value Decomposition of general complex matrix.

*CMLIB Library (LINPAKS Sublibrary)*

**SSVDC** Computes the singular value decomposition of a real n-by-p matrix X, dimensioned X(LDX,P). Has options to allow computation of only the singular values, or singular values and associated decomposition matrices.

*IMSL Subprogram Library*

**LSVDB** Singular value decomposition of a bidiagonal matrix.

**LSVDF** Singular value decomposition of a real matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**LSVCR** Compute the singular value decomposition of a complex matrix.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**LSVRR** Compute the singular value decomposition of a real matrix.

*JCAM Software Library*

**PSVD** Computes in an efficient and reliable way a basis for the left and/or right singular subspace of a matrix corresponding to its smallest singular values. The dimension of the desired subspace may be given or may depend on a given upper bound for those smallest singular values. From:

"An Efficient and Reliable Algorithm for Computing the Singular Subspace of a Matrix Associated with its Smallest Singular Values", by S. Van Huffel, J. Vandewalle, and A. Haegemans, J. Comp. Appl. Math. 19, (1987) 313-330.

*NAG Subprogram Library*

**F02SZF** Provides a singular value decomposition of a real n-by-n upper bidiagonal matrix A into $QDP^T$, where Q and P are n-by-n orthogonal matrices, and $D = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ with $\sigma_1 \geq \sigma_2 \geq \ldots, \sigma_n \geq 0$, these being the singular values of A.

**F02WAF** Finds the singular values and the right-hand singular vectors (principal components) of a real rectangular m-by-n matrix A, where m≥n.

**F02WBF** Finds the singular values and right-hand singular vectors (principal components) of a real rectangular m-by-n matrix A, where m≤n.

**F02WCF** Computes the singular values and left- and right-hand singular vectors of a real rectangular m-by-n matrix A, $A = QDP^T$, where $Q^TQ = P^TP = I_k$, k = min(m,n) and $D = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k)$ with $\sigma_1 \geq \sigma_2 \geq \ldots, \geq \sigma_k \geq 0$.

**F02WDF** Returns the Householder QU factorisation of a real rectangular m by n (m≥n) matrix A. Further, on request or if A is not of full rank, part or all of the singular value decomposition of A is returned.

*NASHLIB Subprogram Library*

**A1A2** Compute singular value decomposition and use it to solve linear least squares problem. Illustrates use of algorithms A1SVD and A2LSVD. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 30-34.)

**A4A4A** Givens reduction, singular value decomposition and application to solve a linear least squares problem. Illustrates use of algorithms A4A and A4LSGS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 47-49.)

*NMS Subprogram Library*

**SSVDC** Computes the singular value decomposition of a real n-by-p matrix X, dimensioned X(LDX,P). Has options to allow computation of only the singular values, or singular values and associated decomposition matrices.

*Scientific Desk PC Subprogram Library*

**D6SVD** Reduces a real n-by-p matrix X by orthogonal transformations U and V to diagonal form. The diagonal elements S(I) are the Singular Values of X. The columns of U are the corresponding left Singular Vectors, and the columns of V are the right Singular Vectors.

*SCRUNCH Subprogram Library*

**SVD** Calculates the singular value decomposition (SVD) of a given rectangular matrix. In BASIC.

## D7b : Update Cholesky decomposition

*CMLIB Library (LINPAKC Sublibrary)*

**CCHDD** Downdates Cholesky factorization of positive definite complex matrix.

**CCHEX** Updates Cholesky factorization of positive definite complex matrix.

**CCHUD** Updates Cholesky factorization of positive definite matrix.

*CMLIB Library (LINPAKS Sublibrary)*

**SCHDD** Downdates Cholesky factorization of real positive definite matrix.

**SCHEX** Updates Cholesky factorization of real positive definite matrix.

**SCHUD** Updates Cholesky factorization of real positive definite matrix.

*IMSL MATH/LIBRARY Subprogram Library*

**LDNCH** Downdate the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is removed.

**LUPCH** Update the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is added.

| **D7c :** | **Update $QR$ decomposition** |
|---|---|

*Collected Algorithms of the ACM*

**A580** QRUP: a set of Fortran subroutines which compute a QR factorization of an m-by-n matrix A based on the Gramm-Schmidt orthogonalization process. They provide for an update of the factorization when rows or columns are added or deleted from A, or when A is modified by the addition of a rank one matrix. (See A. Buckley, ACM TOMS 7 (1981) pp. 548-549.)

*IMSL MATH/LIBRARY Subprogram Library*

**LUPQR** Compute an updated QR factorization after the rank-one matrix $\alpha xy^T$ is added.

| **D8 :** | **Other matrix equations (e.g., $AX + XB = C$)** |
|---|---|

*Collected Algorithms of the ACM*

**A598** SQUINT: a Fortran subprogram for computing solvents of the matrix equation $AX^2 + BX + C = 0$. (See G.W. Davis, ACM TOMS 9 (1983) pp. 246-254.)

# D9: Overdetermined and underdetermined system of equations, singular systems, pseudo-inverses

A system of $m$ linear equations in $n$ unknowns may be represented by a matrix equation of the form

$$Ax = b$$

where $A$ is an $m \times n$ matrix, $x$ is a vector of length $n$, and $b$ is a vector of length $m$. The system is called *overdetermined* if $m > n$ (more equations than unknowns), *underdetermined* if $m < n$ (fewer equations than unknowns), and *square* if $m = n$.

A system is termed *consistent* if $y^T b = 0$ for all $y$ such that $y^T Ax = 0$. (This is equivalent to the usual meaning of the word consistent.) Consistent systems have solutions, while inconsistent ones do not. The solution to a consistent system is unique if and only if $A$ is a square nonsingular matrix; software for this problem is classified in class D2. There are an infinite number of solutions to all other consistent systems. In practical problems overdetermined systems are likely to have no solutions, while underdetermined systems are likely to possess an infinite number of solutions.

One can characterize the set of all solutions a consistent system as follows. Suppose that $x_0$ is a particular solution to $Ax = b$ and $N$ is an $n \times k$ matrix whose columns form a basis for the null space of $A$. The *null space* of $A$ is the set of all vectors $z$ such that $Az = 0$; we let $k$ denote its dimension. Any solution to $Ax = b$ may then be written in the form

$$x_0 + Ny$$

for some vector $y$ of length $k$. All routines for this problem will determine a particular solution $x_0$; some will also determine the matrix $N$. Some extra condition is usually required in order to specify which particular solution $x_0$ is desired. This is usually some function which one wants to minimize over all solutions $x$. One example is $||x||$, in which case one obtains a *minimum norm solution.*

Although there are no solutions to an inconsistent system, it is often useful to find the best approximation to the solution in some sense, that is, a vector $x$ which minimizes $||Ax - b||$, the distance between the vectors $Ax$ and $b$, in some norm. The usual case is the $L_2$ (Euclidean) norm, in which case $x$ is a solution in the least squares sense. Problems involving overdetermined systems of this type are often obtained from linear regression models. Software for approximate solution in the $L_1$ and $L_\infty$ (Chebyshev or minimax) norms are also available.

In some cases it is desirable to place additional constraints on the solution. These normally take the form of either linear equality or inequality constraints. The imposition of linear equality constraints in overdetermined systems is equivalent to the requirement that some equations be satisfied exactly.

Additional computational difficulty occurs when the problem is *rank deficient.* In the case of overdetermined systems this means that the columns of $A$ are linearly dependent, while in the case of underdetermined systems it means that the rows of $A$ are linearly dependent. Rank deficient overdetermined systems occur, for example, in regression models with redundant (or nearly redundant) variables. Some software only applies to the full rank case, and hence one should be careful in selecting software when one suspects rank deficiency.

## References

[GVL83] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 1983.

[LH74] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems.* Prentice-Hall, Englewood Cliffs, NJ, 1974.

**D9a1 : Least squares ($L_2$) solution of singular, overdetermined or underdetermined systems of linear equations without constraints**

*Collected Algorithms of the ACM*

**A576** MODGE and REFINE: Fortran subroutines for solving an n-by-n system of (possibly singular) linear algebraic equations. The algorithm consists of Gaussian elimination combined with a new pivoting strategy which is particularly well suited to problems where residuals can be made small by solving for fewer than n of the unknowns. (See I. Barrodale and G.F. Stuart, ACM TOMS 7 (1981) pp. 391-397.)

**A583** LSQR: Fortran subprogram for solving overdetermined or underdetermined sparse systems of linear equations, sparse least squares problems, and damped sparse least squares problems. (See C.C. Paige and M.A. Saunders, ACM TOMS 8 (1982) pp. 195-209.)

**A641** EXSOLG: a Fortran subprogram for the exact solution of general (m-by-n) systems of linear equations with integer coefficients. If the system is singular a least square solution is computed. (See J. Springer, ACM TOMS 12 (1986) p. 149.)

*CMLIB Library (FC Sublibrary)*

**BNDACC** Introduce new blocks of data for banded least squares problems.

**BNDSOL** Solves least squares problem AX=B for banded matrices.

**HFTI** Solves linear least squares problem AX=B.

*CMLIB Library (LINPAKC Sublibrary)*

**CQRSL** Applies the output of CQRDC to compute coordinate transformations, projections, and least squares solutions (general complex matrix).

*CMLIB Library (LINPAKS Sublibrary)*

**SQRSL** Applies the output of SQRDC to compute coordinate transformations, projections, and least squares solutions (general real matrix).

*CMLIB Library (SGLSS Sublibrary)*

**LLSIA** Computes least squares solution to AX=B with A an m-by-n matrix with m≥n. Flexible version of SGLSS.

• **SGLSS** Solves linear least squares problems. Emphasis is put on detecting possible rank deficiency. Performs QR factorization using Householder transformations. Easy-to-use driver for LLSIA and ULSIA.

**ULSIA** Finds the minimal length solution of the underdetermined system of equations AX=B where A is an m-by-n matrix with m≤n. Flexible version of SGLSS.

*CMLIB Library (SQRLSS Sublibrary)*

**SQRLSS** For solving linear systems in least squares sense. Finds solution and residual after matrix factored by SQRANK.

*CMLIB Library (SUDSODS Sublibrary)*

**SODS** Solves an overdetermined system of linear equations. For full rank matrices the unique least squares solution is provided. The least squares solution of minimal length can be obtained in the rank deficient case.

**SUDS** Solves underdetermined systems of linear equations. For full rank matrices the minimum norm solution is returned, as well as an orthonormal basis for the null space of the matrix. If the system of equations is inconsistent only the least squares solution of minimal length is computed.

*IMSL Subprogram Library*

**OFIMA3** Least squares solution to the matrix equation AT = B.

*IMSL MATH/LIBRARY Subprogram Library*

**LQRSL** Compute the coordinate transformation, projection, and solution for the least squares problem.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**LSBRR** Solve a linear least squares problem with iterative refinement.

**LSQRR** Solve a linear least squares problem without iterative refinement.

*JCAM Software Library*

**DTLS** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution. From: "The Extended Classical Total Least Squares Algorithm", by S. Van Huffel, J. Comp. Appl. Math. 25, (1989) 111-119.

**PTLS** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution, and is more efficient than the author's routine DTLS. From: "The Partial Total Least Squares Algorithm", by S. Van Huffel and J. Vandewalle, J. Comp. Appl. Math. 21, (1988) 333-341.

*NAG Subprogram Library*

**F04AMF** Calculates the accurate least squares solution of a set of m linear equations in n unknowns, m≥n and rank = n with multiple right hand sides, AX=B.

• **easy-to-use version**

**F04ANF** Calculates the approximate least squares solution of a set of m linear equations in n unknowns, m≥n and rank = n with a single right hand side, Ax=b, where A has been decomposed into triangular matrices using F01AXF.

**F04JAF** Finds the minimal solution of a linear least squares problem, Ax=b, where A is a real m by n (m≥n) matrix and b is an m element vector.

**F04JDF** Finds the minimal solution of a linear least squares problem, Ax = b, where A is a real m by n (m≤n) matrix and b is an m element vector.

**F04JGF** Finds the solution of a linear least squares problem, Ax=b, where A is a real m by n (m≥n) matrix and b is an m element vector. If the matrix of observations is not of full rank, then the minimal least squares solution is returned.

**F04QAF** Solves sparse unsymmetric equations, sparse linear least squares problems and sparse damped linear least squares problems using a Lanczos algorithm.

**F04YAF** Returns elements of the estimated variance-covariance matrix of the sample regression coefficients for the solution of a linear least squares problem.

*NASHLIB Subprogram Library*

**A1A2** Compute singular value decomposition and use it to solve linear least squares problem. Illustrates use of algorithms A1SVD and A2LSVD. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 30-34.)

**A4A4A** Givens reduction, singular value decomposition and application to solve a linear least squares problem. Illustrates use of algorithms A4A and A4LSGS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 47-49.)

*NMS Subprogram Library*

**SQRLS** Solves m-by-n system Ax=b in least squares sense.

*PORT Subprogram Library*

**CLST2** Finds the least squares solution of a complex linear algebraic system of equations AX=B. B may be a matrix. Uses real arithmetic.

**LSTSQ** Finds the least squares solution of a system of linear equations, AX=B. B may be a matrix.

*Scientific Desk PC Subprogram Library*

**D9GLS** Solves overdetermined linear least squares problems by QR factorizations using Householder transformations. It computes the least squares solution(s) to the problem AX=B where A is an m-by-n matrix, m≥n and B is the m-by-nb matrix of right hand sides. User input bounds on the uncertainty in the elements of A are used to detect numerical rank deficiency.

**D9GLU** Solves underdetermined linear system of equations by LQ factorization using Householder transformations. It computes the minimal length solution(s) to the problem AX=B where A is an m-by-n matrix, m≤n and B is the m-by-nb matrix of right hand sides. User input bounds on the uncertainty in the elements of A are used to detect numerical rank deficiency.

*SCRUNCH Subprogram Library*

**HECOMP** Performs Householder reduction of a rectangular matrix to upper triangular form. Usually used in conjunction with HOLVE to find the least squares solution of an overdetermined linear system. In BASIC.

**D9a2 :** **Chebyshev ($L_\infty$) solution of singular, overdetermined or underdetermined systems of linear equations without constraints**

*Collected Algorithms of the ACM*

**A495** CHEB: a subprogram for the solution of overdetermined systems of linear algebraic equations in the Chebyshev norm using a variant of the simplex method. (See I. Barrodale and C. Phillips, ACM TOMS 1 (1975) pp. 264-270.)

*NAG Subprogram Library*

**E02GCF** Calculates an $l_\infty$ solution to an over-determined system of linear equations.

**D9a3 :** **Least absolute value ($L_1$ ) solution of singular, overdetermined or underdetermined systems of linear equations without constraints**

*Collected Algorithms of the ACM*

**A551** L1: a Fortran subroutine for solving an overdetermined system of linear equations in the $L_1$ norm by using a dual simplex algorithm to the linear programming formulation of the given problem. (See N.N. Abdelmalek, ACM TOMS 6 (1980) pp. 228-230.)

**A552** CL1: a Fortran subroutine for computing an $L_1$ solution to a k-by-n system of linear algebraic equations subject to both linear equality and inequality constraints. The problem is solved using a modification of the simplex method. (See I. Barrodale and F.D.K. Roberts, ACM TOMS 6 (1980) pp. 231-235.)

*NAG Subprogram Library*

**E02GAF** Calculates an l1 solution to an over-determined system of linear equations.

**D9a4 :** **Other solutions of singular, overdetermined or underdetermined systems of linear equations without constraints**

*Collected Algorithms of the ACM*

**A576** MODGE and REFINE: Fortran subroutines for solving an n-by-n system of (possibly singular) linear algebraic equations. The algorithm consists of Gaussian elimination combined with a new pivoting strategy which is particularly well suited to problems where residuals can be made small by solving for fewer than n of the unknowns. (See I. Barrodale and G.F. Stuart, ACM TOMS 7 (1981) pp. 391-397.)

*CMLIB Library (SUDSODS Sublibrary)*

**SUDS** Solves underdetermined systems of linear equations. For full rank matrices the minimum norm solution is returned, as well as an orthonormal basis for the null space of the matrix. If the system of equations is inconsistent only the least squares solution of minimal length is computed.

*JCAM Software Library*

**DTLS** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution. From: "The Extended Classical Total Least Squares Algorithm", by S. Van Huffel, J. Comp. Appl. Math. 25, (1989) 111-119.

**PTLS** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution, and is more efficient than the author's routine DTLS. From: "The Partial Total Least Squares Algorithm", by S. Van Huffel and J. Vandewalle, J. Comp. Appl. Math. 21, (1988) 333-341.

*Scientific Desk PC Subprogram Library*

**D9GLU** Solves underdetermined linear system of equations by LQ factorization using Householder transformations. It computes the minimal length solution(s) to the problem AX=B where A is an m-by-n matrix, m≤n and B is the m-by-nb matrix of right hand sides. User input bounds on the uncertainty in the elements of A are used to detect numerical rank deficiency.

## D9b1 : Least squares ($L_2$) solution of singular, overdetermined or underdetermined systems of linear equations with constraints

*Collected Algorithms of the ACM*

**A544** L2A and L2B: Fortran subroutines for solving weighted least squares problems by modified Gram-Schmidt with iterative refinement. The types of problems solved include overdetermined and underdetermined systems of linear equations, and problems where the solution is subject to linear equality constraints. The covariance matrix of the solution vector is computed. (See R.H. Wampler, ACM TOMS 5 (1979) pp. 494-499.)

**A587** LSEI and WNNLS: Fortran subprograms for solving least squares problems with linear equality and/or inequality constraints. (See R.J. Hanson and K.H. Haskell, ACM TOMS 8 (1982) pp. 323-333.)

*CMLIB Library (BOCLS Sublibrary)*

**DBOCLS** Solves the general linearly constrained linear least squares problems.

**DBOLS** Solves linear least squares problems with simple bounds on the variables.

*CMLIB Library (FC Sublibrary)*

**LPDP** Solves least projected distance problem.

**LSEI** Solves linearly constrained least squares problem with equality and inequality constraints. Covariance matrix optionally computed.

**WNNLS** Solves linearly constrained non-negative least squares problem.

*IMSL Subprogram Library*

**LLBQF** Solution of linear least squares problems, high accuracy solution.

**LLSQF** Solution of linear least squares problems.

*NAG Subprogram Library*

**E04NCF** Solves linearly constrained linear least-squares problems and convex quadratic programming problems. E04NCF may also be used to solve linear programming problems and to find a feasible point with respect to a set of linear inequality constraints. E04NCF treats all matrices as dense and hence is not intended for large sparse problems.

*Scientific Desk PC Subprogram Library*

**K1A2CD** Solves the least squares problem consisting of the linear constraints CX=Y and the least squares equations EX=F. Vectors X and Y are both unknowns, both of which may have user-specified bounds on each component. The user can specify equality and inequality constraints as well as simple bounds on the solution components.

**K1A2D** Solves EX=F in the least squares sense with bounds on selected X values.

**D9b2 : Chebyshev ($L_\infty$) solution of singular, overdetermined or underdetermined systems of linear equations with constraints**

*Collected Algorithms of the ACM*

**A635** A set of Fortran subroutines for computing the Chebyshev solution of systems of complex linear equations with constraints on the unknowns. Both general linear inequality constraints and simple bound constraints are allowed. (See R.L. Streit, ACM TOMS 11 (1985) pp. 242-249.)

**D9b3 : Least absolute value ($L_1$ ) solution of singular, overdetermined or underdetermined systems of linear equations without constraints**

*Collected Algorithms of the ACM*

**A552** CL1: a Fortran subroutine for computing an $L_1$ solution to a k-by-n system of linear algebraic equations subject to both linear equality and inequality constraints. The problem is solved using a modification of the simplex method. (See I. Barrodale and F.D.K. Roberts, ACM TOMS 6 (1980) pp. 231-235.)

**A563** CL1: a set of subroutines for solving overdetermined systems of linear equations in the $L_1$ sense, with or without linear constraints. (See R.H. Bartels and A.R. Conn, ACM TOMS 6 (1980) pp. 609-614.)

*NAG Subprogram Library*

**E02GBF** Calculates an $l_1$ solution to an over-determined system of linear equations, possibly subject to linear inequality constraints.

**D9c : Generalized matrix inverses**

*Collected Algorithms of the ACM*

**A645** Fortran subroutines for testing programs that compute the generalized inverse of a matrix. (See J.C. Nash and R.L.C. Wang, ACM TOMS 12 (1986) pp. 274-277.)

*IMSL Subprogram Library*

**LGINF** Generalized inverse of a real matrix.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**LSGRR** Compute the generalized inverse of a real matrix.

*NAG Subprogram Library*

**F01BLF** Calculates the rank and pseudo-inverse of an m-by-n real matrix A, $m \geq n$, $\text{rank}(A) \leq n$, using Householder's factorisation.

# E: Interpolation

By interpolation we mean the determination of a function which "passes through" given data values. The data may be one-dimensional (curve fitting), or higher dimensional (surface fitting). More generally, interpolation can also refer to requirements that derivative data match (this is sometimes referred to as osculatory interpolation). The purpose of interpolation is to find a functional form to replace tabular data. This may be design data, for example, in which case one needs a visually pleasing curve which represents the profile of a given object. Interpolation should *not* be used if the data have noise. In that case approximation techniques such as least squares (see classes K1 or L8) are more appropriate.

# E1: Univariate data interpolation (curve fitting)

The problem is to find a function $g(x)$ such that

$$g(x_i) = y_i \quad i = 1, \ldots, n$$

where the points $(x_i, y_i)$ are given.

Polynomials have traditionally been used as interpolating functions. Unfortunately, serious computational problems can arise in determining the coefficients of interpolation polynomials. Furthermore polynomial interpolants often wiggle unphysically between data points. Alternative formulations can sometimes help but the more recent trend is to use a piecewise polynomial interpolant, most often a cubic spline.

A *piecewise polynomial* is a function which is composed of different polynomials joined together at points (called knots). A piecewise polynomial function constructed from polynomials of degree $n$ is called a *spline* if the function has $n - 1$ continuous derivatives at the knots. Thus, a cubic spline is constructed from cubic polynomials joined together so that the entire function has continuous first and second derivatives. Such functions are awkward to deal with analytically because a single neat formula is not available to represent them, but they are perfect for computations where the process of going from $x$ to $g(x)$ can be thought of as a "black box".

When users ask for a spline interpolation to data they normally must input two extra conditions in addition to the data points. That is, the interpolant has two extra degrees of freedom. Often these are used to specify the derivatives of the interpolant at the endpoints, but there are many other choices. Some programs don't have these options and this gives them the appearance of being easier to use. Rather, the extra degrees of freedom have been fixed internally and are not always consistent with the physical model. With just a little care, however, piecewise polynomials can make excellent interpolants and produce results that usually are more acceptable than polynomials.

The cubic Hermites are another heavily used set of interpolating functions. These functions are also piecewise cubic polynomials like cubic splines, but they are only joined together with one continuous derivative overall. They are easy to compute and are more flexible than splines. For example, if the data are monotonic it is possible to obtain a Hermite cubic which not only interpolates the data but is also monotonic between the data values. Splines often fail to behave this way.

Two common methods used to represent piecewise polynomial functions are the pp and B representations. In the pp representation one simply stores a list of coefficients of the polynomial on each interval. In the B representation the coefficients of an expansion in terms of a particular piecewise polynomial basis (the so-called B-splines) are stored; this is analogous to writing a polynomial as a linear sum of Legendre polynomials. The pp representation requires more storage that the B representation, but is simpler to evaluate. On the other hand, the B-representation is simpler to compute.

Subprograms for piecewise polynomial interpolation often come in pairs. One routine sets up the interpolant from the given data and is called just once. The second evaluates the interpolant or its derivatives at any point by using the results of the first subprogram and may be called many times. Sometimes a routine is also provided to integrate the interpolant.

## References


[Cox77] M. G. Cox. A survey of numerical methods for data and function approximation. In D. Jacobs, editor, *The State of the Art in Numerical Analysis*, pages 627–668. Academic Press, New York, 1977.

[deB78] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, New York, 1978.

[Hay70] J. G. Hayes, editor. *Numerical Approximation to Functions and Data.* The Athlone Press, London, 1970.

[KMN89] David Kahaner, Cleve Moler, and Stephen Nash. *Numerical Methods and Software.* Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[Sch81] L. L. Schumaker. *Spline Functions: Basic Theory.* John Wiley & Sons, New York, 1981.


| **E1a :** | **Interpolation of univariate data by polynomial splines (piecewise polynomials)** |
|---|---|

*Collected Algorithms of the ACM*

**A507** QUINAT: an Algol procedure to determine the interpolating quintic natural spline interpolant to a given set of data points. (See J.G. Herriot and C.H. Reinsch, ACM TOMS 2 (1976) pp. 281-289.)

**A514** Algol procedures for piecewise cubic interpolation using local data. (See M.R. Ellis and D.H. McLain, ACM TOMS 3 (1977) pp. 175-179.)

**A547** DCSINT and DCSSMO: Fortran subroutines for discrete cubic spline interpolation and smoothing. (See C.S. Duris, ACM TOMS 6 (1980) pp. 92-103.)

**A574** A package of Fortran subprograms for constructing a shape-preserving osculatory quadratic spline. The spline is a piecewise quadratic Bernstein polynomial with a continuous first derivative which interpolates given function and first derivative values, and preserves monotonicity and convexity in the data. (See D.F. Mcallister and J.A. Roulier, ACM TOMS 7 (1981) pp. 384-386.)

**A592** RANGE: Given values of an unknown function f at n distinct points on an interval, an integer k, and a finite bound on the kth derivative of f, this routine determines the range of possible values of f(x) in the interval. (See P.W. Gaffney, ACM TOMS 9 (1983) pp. 98-116.)

**A600** QUINAT, QUINEQ, and QUIND: Fortran subprograms for quintic natural spline interpolation. This is a translation of ACM Algorithm 507. (See J.G. Herriot and C.H. Reinsch, ACM TOMS 9 (1983) pp. 258-259.)

*CMLIB Library (BSPLINE Sublibrary)*

**BINT4** Computes B-spline which interpolates given X,Y data with various end conditions. The B representation is used.

**BINTK** Produces coefficients of k-th order B-spline with given knots and with values at given points.

*CMLIB Library (PCHIPS Sublibrary)*

**PCHIC** Determines a piecewise monotone, piecewise cubic Hermite interpolant to given data. User control is available over boundary conditions and/or treatment of points where monotonicity switches direction.

• **PCHIM** Determines a monotone piecewise cubic Hermite interpolant to given data. Default boundary values are provided which are compatible with monotonicity. The interpolant will have an extremum at each point where monotonicity switches direction.

**PCHSP** Determines the cubic spline interpolant to given data. User has control over boundary conditions.

• **easy-to-use version**

*IMSL Subprogram Library*

- • **ICSCCU** Cubic spline interpolation.
- **ICSICU** Interpolatory approximation by cubic splines with arbitrary second derivative end conditions.
- **ICSPLN** Cubic spline interpolation with periodic end conditions.
- **IQHSCU** Visually pleasing interpolant of one dimensional data via piecewise cubic Hermite function.

*IMSL MATH/LIBRARY Subprogram Library*

- **BSINT** Compute the spline interpolant, returning the B-spline coefficients.
- **CSAKM** Compute the Akima cubic spline interpolant.
- **CSCON** Compute a cubic spline interpolant which is consistent with the concavity of the data.
- **CSDEC** Compute the cubic spline interpolant with specified derivative endpoint conditions.
- **CSHER** Compute a Hermite cubic spline interpolant.
- **CSINT** Compute the cubic spline interpolant with the "not-a-knot" condition.
- **CSPER** Compute the cubic spline interpolant with periodic boundary conditions.
- **QDDER** Evaluate the derivative of a function defined on a set of points using quadratic interpolation.
- **QDVAL** Evaluate a function defined on a set of points using quadratic interpolation.

*JCAM Software Library*

- **SPISC1** Computes a shape preserving polynomial spline of arbitrary degree to a given set of data.

*NAG Subprogram Library*

- **E01BAF** Determines a cubic-spline interpolant to a given set of data.
- **E02BAF** Computes a weighted least-squares approximation to an arbitrary set of data points by a cubic spline with knots prescribed by the user. Cubic spline interpolation can also be carried out.

*NMS Subprogram Library*

- **PCHEZ** Finds either spline or visually-pleasing piecewise cubic interpolant to input N-arrays X, F. Evaluate resulting function with PCHEV; integrate it with PCHQA.

*PORT Subprogram Library*

- **CSPFI** Fits a cubic spline function to n input data pairs (x,y) with various endpoint conditions.
- • **CSPIN** Interpolates at requested points in given input data using a spline approximation, not a least squares fit.

*Scientific Desk PC Subprogram Library*

- **E1ACS** Piecewise cubic Hermite spline. This routine sets derivatives needed to determine the Hermite representation of the cubic spline interpolant to given data, with specified boundary conditions. The resulting piecewise cubic may be evaluated by E3VAL.
- **E1AM** Piecewise cubic Hermite interpolation to monotonic data. This routine sets derivatives needed to determine a monotone piecewise cubic Hermite interpolant to given data. Boundary values are provided which are compatible with monotonicity. The interpolant will have an extremum at each point where monotonicity switches direction.

*SCRUNCH Subprogram Library*

- **SPLINE** Calculates the coefficients of a spline interpolant to given data. Usually used in conjunction with SEVAL to subsequently evaluate the spline at arbitrary points. In BASIC.

| | |
|---|---|
| **E1b :** | **Interpolation of univariate data by polynomials** |

*NAG Subprogram Library*

**E01AAF** Interpolates at a given point X from a table of function values y(i) evaluated at equidistant or non-equidistant points x(i) (i = 1,2, ..., N+1), using Aitken's technique of successive linear interpolations.

**E01ABF** Interpolates at a given point X from a table of function values evaluated at equidistant points, by Everett's formula.

**E01AEF** Constructs the Chebyshev series representation of a polynomial interpolant to a set of data which may contain derivative values.

**E02AFF** Computes the coefficients of a polynomial, in its Chebyshev series form, which interpolates (passes exactly through) data at a special set of points. Least-squares polynomial approximations can also be obtained.

| | |
|---|---|
| **E1c :** | **Interpolation of univariate data by nonpolynomial functions (e.g., rational, trigonometric)** |

*Collected Algorithms of the ACM*

**A585** EXTRAP: A Fortran subroutine for sequence extrapolation (based upon the E-algorithm) and generalized interpolation by a linear combination of functions forming a Chebyshev system (based upon the Muhlbach-Neville-Aitken algorithm). Includes the Epsilon Algorithm of Wynn as a special case. (See C. Brezinski, ACM TOMS 8 (1982) pp. 290-301.)

*DATAPLOT Interactive System*

**EXACT RATIONAL FIT** Exactly fits a rational function (ratio of polynomials) to data and optionally applies that fit to a more general data set. Output includes exact fit coefficients of the specified rational function and, for the more general data set, standard deviations of the coefficients, predicted values, residuals, residual standard deviation, degrees of freedom, sum of absolute deviations, and lack-of-fit analysis if there is replication.

*NAG Subprogram Library*

**E01RAF** Produces, from a set of function values and corresponding abscissae, the coefficients of an interpolating rational function expressed in continued fraction form.

## E2: Multivariate data interpolation (surface fitting)

The typical multivariate interpolation problem is to find a function $g(x, y)$ such that

$$g(x_i, y_i) = z_i \qquad i = 1, \ldots, n$$

where the points $(x_i, y_i, z_i)$ are given. This is a two-dimensional interpolation problem. It corresponds to the determination of a surface passing through a given set of points in three-space. Higher-dimensional problems also occur, although it is difficult to visualize the interpolant.

The difficulty of multidimensional interpolation depends upon the regularity of the data. If data are provided at all points on a rectangular grid, then the problem is relatively easy, since essentially one-dimensional methods may be applied. Most of the software for this case is based upon tensor products of one-dimensional piecewise polynomials. A number of libraries now provide software for two-dimensional gridded data interpolation, and several also handle the three-dimensional case. The more difficult case is when the data are not on a regular grid; this is the so-called scattered data interpolation problem. One of the difficulties here is that data may be dense in some areas and very sparse (or non-existent) in others; no numerical method can be expected to reproduce the underlying function in the absence of data. One should avoid using software for this problem without carefully examining the result. Much less software is available for this case, and what is available is restricted to two dimensions.

Numerical methods for scattered data interpolation can be classified as either global or local, although several hybrids have also been proposed. In a global method one writes the interpolant as a linear combination of functions each of which is non-zero at a single interpolation point and zero at all others. Such interpolants are easy to construct, expensive to evaluate, and sometimes lead to visually displeasing surfaces. Local methods often begin by triangularizing the data (determining a triangular grid with vertices corresponding to interpolation points). A separate polynomial function is then determined on each triangle which interpolates the local data. Although the resulting interpolant is easy to evaluate, determining it is more complex, especially if continuity restrictions are imposed.

## References

[Bar77] R. E. Barnhill. Representation and approximation of surfaces. In J. R. Rice, editor, *Mathematical Software III*, pages 69–120. Academic Press, New York, 1977.

[Fra82] R. Franke. Scattered data interpolation: A comparison of methods. *Mathematics of Computation*, 38:181–200, 1982.

[Law77] C. L. Lawson. Software for c1 surface interpolation. In J. R. Rice, editor, *Mathematical Software III*, pages 161–194. Academic Press, New York, 1977.

[Pre75] P. M. Prenter. *Splines and Variational Methods*, chapter 6. John Wiley & Sons, New York, 1975.

## E2a : Interpolation of gridded multivariate data

*CMLIB Library (TENSBS Sublibrary)*

**B2INK** Computes parameters of a piecewise-polynomial that interpolates a given set of two-dimensional gridded data. (Use B2VAL to evaluate function.)

**B3INK** Computes parameters of a piecewise-polynomial that interpolates a given set of three-dimensional gridded data. (Use B3VAL to evaluate function.)

*IMSL Subprogram Library*

**IBCCCU** Bicubic spline two-dimensional coefficient calculator.

**IBCIEU** Bicubic spline two-dimensional interpolator.

*IMSL MATH/LIBRARY Subprogram Library*

**BS2IN** Compute a two-dimensional tensor-product spline interpolant, returning the tensor-product B-spline coefficients.

**BS3IN** Compute a three-dimensional tensor-product spline interpolant, returning the tensor-product B-spline coefficients.

**QD2DR** Evaluate the derivative of a function defined on a rectangular grid using quadratic interpolation.

**QD2VL** Evaluate a function defined on a rectangular grid using quadratic interpolation.

**QD3DR** Evaluate the derivative of a function defined on a rectangular three-dimensional grid using quadratic interpolation.

**QD3VL** Evaluate a function defined on a rectangular three-dimensional grid using quadratic interpolation.

*NAG Subprogram Library*

**E01ACF** Interpolates at a given point (A,B) from a table of function values defined on a rectangular grid in the X-Y plane, by fitting bi-cubic spline functions.

## E2b : Interpolation of scattered, non-gridded multivariate data

*Collected Algorithms of the ACM*

**A526** IDBVIP and IDSFFT: Fortran subprograms for bivariate interpolation and smooth surface fitting for irregularly distributed data points. (See H. Akima, ACM TOMS 4 (1978) pp. 160-164.)

**A623** A set of Fortran programs for constructing an interpolant with one continuous derivative from data values associated with arbitrarily distributed nodes on the surface of a sphere. (See R.J. Renka, ACM TOMS 10 (1984) pp. 417-436 and 437-439.)

**A624** A set of Fortran programs for triangulation and interpolation at arbitrarily distributed points in the plane. (See R.J. Renka, ACM TOMS 10 (1984) pp. 440-442.)

**A660** QSHEP2D: Fortran routines implementing the quadratic Shepard method for bivariate interpolation of scattered data. (See R. J. Renka, ACM TOMS 14 (1988) pp. 149-150.)

**A661** QSHEP3D: Fortran routines implementing the quadratic Shepard method for trivariate interpolation of scattered data. (See R. J. Renka, ACM TOMS 14 (1988) pp. 151-152.)

*CMLIB Library (LOTPS Sublibrary)*

**LOTPS** Passes smooth function through two-dimensional scattered data and returns an array of interpolated values on user specified grid. Based on local thin-plate splines.

*IMSL Subprogram Library*

**IQHSCV** Smooth surface fitting with irregularly distributed data points.

*IMSL MATH/LIBRARY Subprogram Library*

**SURF** Compute a smooth bivariate interpolant to scattered data which is locally a quintic polynomial in two variables.

## E3a1 : Evaluation of fitted functions

*CMLIB Library (BSPLINE Sublibrary)*

**BSPEV** Calculates the value of a spline and its derivatives at X from its B representation.

**BVALU** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its B representation.

**PPVAL** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its piecewise polynomial representation.

*CMLIB Library (PCHIPS Sublibrary)*

**CHFDV** Evaluates a cubic polynomial and its first derivative at an array of points. The polynomial must be given in Hermite form.

**CHFEV** Evaluates a cubic polynomial given in Hermite form at an array of points.

**PCHFD** Evaluates a piecewise cubic Hermite function and its first derivative at an array of points.

**PCHFE** Evaluates a piecewise cubic Hermite function at an array of points.

*CMLIB Library (TENSBS Sublibrary)*

**B2VAL** Evaluates the two-dimensional interpolating function computed by B2INK or one of its partial derivatives.

**B3VAL** Evaluates the three-dimensional interpolating function computed by B3INK or one of its partial derivatives.

*IMSL Subprogram Library*

**IBCEVL** Evaluation of a bicubic spline.

**ICSEVU** Evaluation of a cubic spline.

*IMSL MATH/LIBRARY Subprogram Library*

**BS2VL** Evaluate a two-dimensional tensor-product spline, given its tensor-product B-spline representation.

**BS3VL** Evaluate a three-dimensional tensor-product spline, given its tensor-product B-spline representation.

**BSVAL** Evaluate a spline, given its B-spline representation.

**CSVAL** Evaluate a cubic spline.

**PPVAL** Evaluate a piecewise polynomial.

*NAG Subprogram Library*

**E01RBF** Evaluates continued fractions of the form produced by NAG FORTRAN Library routine E01RAF.

**E02AEF** Evaluates a polynomial from its Chebyshev series representation.

**E02AKF** Evaluates a polynomial from its Chebyshev series representation, allowing an arbitrary index increment for accessing the array of coefficients.

**E02BBF** Evaluates a cubic spline from its B-spline representation.

**E02BCF** Evaluates a cubic spline and its first three derivatives from its B-spline representation.

**E02CBF** Evaluates a bivariate polynomial from the rectangular array of coefficients in its double Chebyshev series representation.

**E02DBF** Calculates values of a bicubic spline from its B-spline representation.

*NMS Subprogram Library*

**PCHEV** Evaluates piecewise cubic polynomial and its derivative at NVAL points in array XVAL given N-arrays X,F,D; results are put in arrays FVAL and DVAL. Usually used following a call to PCHEZ, but can be used independently.

*PORT Subprogram Library*

**CSPFE** Evaluates a cubic spline function which has already been fit to n input data pairs (x,y) by CSPFI.

**SPLN1** Evaluates a function and derivatives described previously by an expansion in terms of B-splines.

**SPLN2** Evaluates a function described by a previously determined expansion in B-splines. More flexible than SPLN1.

**SPLND** Evaluates at a given set of points a function described by a previously determined expansion in terms of B-splines.

**SPLNE** Evaluates, at a set of points, a function described by a previously determined expansion in terms of B-splines.

*Scientific Desk PC Subprogram Library*

**E1AHE** Evaluates a piecewise cubic Hermite function and its first derivative at an array of points. It may be used by itself for Hermite interpolation, or as an evaluator for E1AM or E1AIC.

**E1AQ** Controls the evaluation of an osculatory quadratic spline which is consistent with the shape of the data (shape preserving).

**E1VDV** Evaluates a cubic polynomial given in Hermite form and its first derivative at an array of points where the interval is known. While designed for use by E1AHE, it may be useful directly as an evaluator for a piecewise cubic Hermite function in applications such as graphing, where the interval is known. If only function values are required, use E3V3V instead.

**E3VAL** Evaluates a piecewise cubic Hermite function at an array of points where the interval is unknown or the evaluation array spans more than one interval. This routine may be used by itself for Hermite interpolation or as an evaluator for E1AM.

**E3VEV** Evaluates a cubic polynomial given in Hermite form at an array of points where the interval is known in advance. While designed for use by E3VAL, this routine may be used by itself as an evaluator of a piecewise cubic Hermite function in applications, such as graphing, where the interval is known in advance.

## E3a2 : Evaluation of derivatives of fitted functions

*CMLIB Library (BSPLINE Sublibrary)*

**BSPEV** Calculates the value of a spline and its derivatives at X from its B representation.

**BVALU** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its B representation.

**PPVAL** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its piecewise polynomial representation.

*CMLIB Library (PCHIPS Sublibrary)*

**CHFDV** Evaluates a cubic polynomial and its first derivative at an array of points. The polynomial must be given in Hermite form.

**PCHFD** Evaluates a piecewise cubic Hermite function and its first derivative at an array of points.

*CMLIB Library (TENSBS Sublibrary)*

**B2VAL** Evaluates the two-dimensional interpolating function computed by B2INK or one of its partial derivatives.

**B3VAL** Evaluates the three-dimensional interpolating function computed by B3INK or one of its partial derivatives.

*IMSL Subprogram Library*

**DBCEVL** Bicubic spline mixed partial derivative evaluator.

**DCSEVU** Cubic spline first and second derivative evaluator.

*IMSL MATH/LIBRARY Subprogram Library*

**BS2DR** Evaluate the derivative of a two-dimensional tensor-product spline, given its tensor-product B-spline representation.

**BS3DR** Evaluate the derivative of a three-dimensional tensor-product spline, given its tensor-product B-spline representation.

**BSDER** Evaluate the derivative of a spline, given its B-spline representation.

**CSDER** Evaluate the derivative of a cubic spline.

**PPDER** Evaluate the derivative of a piecewise polynomial.

*NAG Subprogram Library*

**E02AHF** Determines the coefficients in the Chebyshev series representation of the derivative of a polynomial given in Chebyshev series form.

**E02BCF** Evaluates a cubic spline and its first three derivatives from its B-spline representation.

*NMS Subprogram Library*

**PCHEV** Evaluates piecewise cubic polynomial and its derivative at NVAL points in array XVAL given N-arrays X,F,D; results are put in arrays FVAL and DVAL. Usually used following a call to PCHEZ, but can be used independently.

*PORT Subprogram Library*

**SPLN1** Evaluates a function and derivatives described previously by an expansion in terms of B-splines.

**SPLN2** Evaluates a function described by a previously determined expansion in B-splines. More flexible than SPLN1.

**SPLND** Evaluates at a given set of points a function described by a previously determined expansion in terms of B-splines.

*Scientific Desk PC Subprogram Library*

**E1AHE** Evaluates a piecewise cubic Hermite function and its first derivative at an array of points. It may be used by itself for Hermite interpolation, or as an evaluator for E1AM or E1AIC.

**E1VDV** Evaluates a cubic polynomial given in Hermite form and its first derivative at an array of points where the interval is known. While designed for use by E1AHE, it may be useful directly as an evaluator for a piecewise cubic Hermite function in applications such as graphing, where the interval is known. If only function values are required, use E3V3V instead.

## E3a3 : Quadrature involving fitted functions

*CMLIB Library (BSPLINE Sublibrary)*

**BFQAD** Integrates function times derivative of B-spline from X1 to X2. The B-spline is in B representation.

**BSQAD** Computes the integral of a B-spline from X1 to X2. The B-spline must be in B representation.

**PFQAD** Computes integral on (X1,X2) of product of function and the ID-th derivative of B-spline which is in piecewise polynomial representation.

**PPQAD** Computes the integral of a B-spline from X1 to X2. The B-spline must be in piecewise polynomial representation.

*CMLIB Library (PCHIPS Sublibrary)*

**PCHIA** Evaluates the definite integral of a piecewise cubic Hermite function over an arbitrary interval.

**PCHID** Evaluates the definite integral of a piecewise cubic Hermite function over an interval whose endpoints are data points.

*IMSL Subprogram Library*

**DCSQDU** Cubic spline quadrature.

*IMSL MATH/LIBRARY Subprogram Library*

**BS2IG** Evaluate the integral of a tensor-product spline on a rectangular domain, given its tensor-product B-spline representation.

**BS3IG** Evaluate the integral of a tensor-product spline in three dimensions over a three-dimensional rectangle, given its tensor-product B-spline representation.

**BSITG** Evaluate the integral of a spline, given its B-spline representation.

**CSITG** Evaluate the integral of a cubic spline.

**PPITG** Evaluate the integral of a piecewise polynomial.

*NAG Subprogram Library*

**E02AJF** Determines the coefficients in the Chebyshev series representation of the indefinite integral of a polynomial given in Chebyshev series form.

**E02BDF** Computes the definite integral of a cubic spline from its B-spline representation.

*NMS Subprogram Library*

**PCHQA** Integrates piecewise cubic from A to B given N-arrays X,F,D. Usually used in conjunction with PCHEZ to form cubic, but can be used independently, especially if the abscissas are equally spaced.

*PORT Subprogram Library*

**BSPLI** Obtains the integrals of basis splines, from the left-most mesh point to a specified set of points.

**SPLNI** Integrates a function described previously by an expansion in terms of B-splines. Several integrations can be performed in one call.

*Scientific Desk PC Subprogram Library*

**E3HIN** Evaluates the definite integral of a piecewise cubic Hermite function over an arbitrary interval.

**E3INT** Evaluates the definite integral of a piecewise cubic Hermite function over an interval whose endpoints are data points.

## E3b : Grid or knot generation

*IMSL MATH/LIBRARY Subprogram Library*

**BSNAK** Compute the not-a-knot spline knot sequence.

**BSOPK** Compute the optimal spline knot sequence.

*PORT Subprogram Library*

**ILUMB** Given a basic mesh, this subdivides each interval into the same number of uniformly spaced points for B-spline use.

**ILUMD** Given a basic mesh, this subdivides each interval into the same number or uniformly spaced points for B-spline use.

**IMNPB** Creates a B-spline mesh from an array of fitting points, using at least n fitting points in each mesh interval.

**IPUMB** Given a basic mesh, this subdivides each interval. Number of points per interval can vary, but uniform in each subdivision.

**IPUMD** Given a basic mesh, this subdivides each interval with a variable number of points. Points are uniform in each interval.

**IUMB** Given interval endpoints, this generates a uniform mesh for B-spline use.

| | |
|---|---|
| IUMD | Given interval endpoints, this generates a uniform mesh. |
| LUMB | Given a basic mesh, this subdivides each interval uniformly for B-spline use. Multiplicities are allowed. |
| LUMD | Given a basic mesh, this subdivides each interval into the same number of uniformly spaced points. |
| MNPB | Creates a B-spline mesh from an array of fitting points, using at least n fitting points in each interval. |
| PUMB | Given a basic mesh, this subdivides each interval into a uniform but variable number of points. Multiplicities can occur. |
| PUMD | Given a basic mesh, this subdivides each interval into a uniform but variable number of points. |
| UMB | Given interval endpoints, this generates a uniform mesh, with needed multiplicities for B-spline use. |
| UMD | Given interval endpoints, this generates a uniform mesh of distinct points. |

## E3c : Manipulation of basis functions (e.g., evaluation, change of basis)

*CMLIB Library (BSPLINE Sublibrary)*

| | |
|---|---|
| BSPDR | Constructs divided difference table from B representation of B-spline for a derivative calculation. |
| BSPPP | Converts from B representation of B-spline to piecewise polynomial representation. |
| BSPVD | Calculates value and derivatives of order less than NDERIV of all B-spline basis functions which do not vanish at X. |
| BSPVN | Calculates the value of all (possibly) nonzero B-spline basis functions at X of a given order. |
| INTRV | Computes the index into a knot or breakpoint sequence corresponding to a given point X. |

*IMSL MATH/LIBRARY Subprogram Library*

| | |
|---|---|
| BSCPP | Convert a spline in B-spline representation to piecewise polynomial representation. |

*PORT Subprogram Library*

| | |
|---|---|
| BSPL1 | Evaluates, at a given set of points in a specified mesh interval, basis splines together with selected orders of derivatives. |
| BSPLD | Evaluates at a given set of points in a specified mesh interval, basis splines and their derivatives. |
| BSPLN | Evaluates at a given set of points in a specified mesh interval, all the basis splines which are nonzero in that interval. |

## E3d : Other service routines for interpolation

*Collected Algorithms of the ACM*

| | |
|---|---|
| A546 | SOLVEBLOK: a Fortran subprogram for solving almost block diagonal linear systems. Such matrices arise naturally in piecewise polynomial interpolation or approximation and in finite element methods for two-point boundary value problems. (See C. de Boor and R. Weiss, ACM TOMS 6 (1980) pp. 88-91.) |
| A603 | COLROW and ARCECO: Fortran subroutines for solving certain almost block diagonal linear systems by modified alternate row and column elimination. Such systems arise when solving boundary-value problems for ordinary differential equations. COLROW is designed for systems |

whose blocks all have the same dimension; ARCECO is designed for systems whose blocks may have different dimensions. (See J.C. Diaz, G. Fairweather, and P. Keast, ACM TOMS 9 (1983) pp. 376-380.)

*CMLIB Library (PCHIPS Sublibrary)*

**PCHMC** Checks a cubic Hermite function for monotonicity.

*NAG Subprogram Library*

**E02ZAF** Sorts two-dimensional data into rectangular panels.

*PORT Subprogram Library*

**EEBSF** Estimates the error in a given B-spline fit to a function by refining the mesh.

**EEBSI** Estimates the error in a given B-spline fit to a function by refining the mesh intervals selected by user.

**EESFF** Finds the maximum absolute error in a given B-spline fit to a function.

**EESFI** Finds the maximum absolute error in a given B-spline fit to a function on a set of user selected intervals.

*Scientific Desk PC Subprogram Library*

**E3CHK** Piecewise cubic Hermite monotonicity checker.

# F: Solution of Nonlinear Equations

This class covers the solution of a system of $m$ nonlinear equations for $m$ unknowns. Most routines for this problem find a single solution of the given system based upon a starting guess provided by the user. Convergence cannot usually be guaranteed unless the starting guess is reasonably good. A special case is $m = 1$, a single nonlinear equation. Globally convergent algorithms are available for this case, provided one knows an interval in which the solution lies. Algorithms for the case of a single polynomial equation are even more specialized, and a number of programs are available which reliably compute all the zeros of a polynomial of moderate order.

# F1: Single equation

Solving one nonlinear equation,

$$f(x) = 0,$$

for one root is easy if the user has a good estimate of the answer or can bound the interval where the root is to be found.

Usually, the user provides a guess at the root, a Fortran SUBROUTINE or FUNCTION to calculate $f$ for any given $x$, and a specification of the accuracy desired. Some programs stop when $|f(x)|$ is small enough, and others stop when $x$ is believed to be close to a root. Either tolerance might be absolute or relative, depending on the program. The most reliable programs require the user to provide an interval $(a, b)$ in which a root lies. By evaluating $f$ at a sequence of points in the interval, ever smaller sub-intervals are produced which contain the root. Some such programs require $f(a) > 0$ and $f(b) < 0$ or the reverse, and then can guarantee finding a root. Programs which use derivatives of $f$, or estimates of the derivatives, are typically faster than the programs which shrink the interval surrounding the root. Polynomial equations should ordinarily be solved by a special purpose program.

Some library subprograms allow the user to specify a name for the function $f$. In that case the user should be sure that this name appears

1. in an EXTERNAL statement in the main program,
2. in the subroutine CALL, and
3. as the name of a FUNCTION.

## References


[Bre73] R. P. Brent. *Algorithms for Minimization Without Derivatives.* Prentice-Hall, Englewood Cliffs, NJ, 1973.

[KMN89] David Kahaner, Cleve Moler, and Stephen Nash. *Numerical Methods and Software.* Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[SA73] L. F. Shampine and R. C. Allen Jr. *Numerical Computing: An Introduction.* Saunders, Philadelphia, 1973.


# F1a1: Polynomial

Programs for solving polynomial equations,

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n = 0,$$

are quite reliable. They find all $n$ roots of the polynomial. The difficulty is that high-degree polynomials are inherently unstable. Small errors in calculating the $a$'s can give large errors in the roots. Fortunately, many problems which require solving a polynomial equation have formulations which are preferable. The casual user should consult an expert before attempting to solve such a problem with $n$ larger than five.

## References

[JT70] M. A. Jenkins and J. F. Traub. A three-stage variable-shift iteration for polynomial zeros and its relation to the generalized Rayleigh iteration. *Numerische Mathematik*, 14(3):253–263, 1970.

[JT75] M. A. Jenkins and J. F. Traub. Principles for testing polynomial zerofinding programs. *ACM Transactions on Mathematical Software*, 1(1):26–34, 1975.

## F1a1 : Roots of polynomials with real coefficients

*Collected Algorithms of the ACM*

**A493** RPOLY: a subroutine for finding all the zeros of a real polynomial using an algorithm of Jenkins and Traub. (See M.A. Jenkins, ACM TOMS 1 (1975) pp. 178-189.)

*CMLIB Library (CPQR79 Sublibrary)*

**RPQR79** Computes all the zeros of a general real polynomial using eigenvalue methods, requiring n-by-n storage for Nth degree polynomial.

*CMLIB Library (CPZERO Sublibrary)*

**RPZERO** Computes all the zeros of a polynomial with real coefficients. Error bounds are also computed. Uses Newton's method for systems.

*IMSL Subprogram Library*

**ZPOLR** Finds zeros of a polynomial with real coefficients using Laguerre's method.

**ZQADR** Finds zeros of a quadratic with real coefficients.

**ZRPOLY** Finds zeros of a polynomial with real coefficients using the Jenkins-Traub method.

*IMSL MATH/LIBRARY Subprogram Library*

**ZPLRC** Find the zeros of a polynomial with real coefficients using Laguerre's method.

**ZPORC** Find the zeros of a polynomial with real coefficients using the Jenkins-Traub three-stage algorithm.

*NAG Subprogram Library*

**C02AEF** Finds all the roots of a real polynomial equation, using the method of Grant and Hitchins.

*PORT Subprogram Library*

**RPOLY** Finds zeros of a polynomial with real coefficients. Output zeros are in a pair of arrays, for real and imaginary part

*Scientific Desk PC Subprogram Library*

**F1A1A** Finds the zeros of a polynomial with real coefficients.

## F1a2 : Roots of polynomials with complex coefficients

*CMLIB Library (CPQR79 Sublibrary)*

**CPQR79** Computes all the zeros of a general complex polynomial using eigenvalue methods, requiring n-by-n storage for Nth degree polynomial.

*CMLIB Library (CPZERO Sublibrary)*

**CPZERO** Computes all the zeros of a polynomial with complex coefficients. Error bounds are also obtained. Uses Newton's Method for systems.

*IMSL Subprogram Library*

**ZCPOLY** Finds zeros of a polynomial with complex coefficients using the Jenkins-Traub method.

**ZQADC** Finds zeros of a quadratic with complex coefficients.

*IMSL MATH/LIBRARY Subprogram Library*

**ZPOCC** Finds the zeros of a polynomial with complex coefficients.

*NAG Subprogram Library*

**C02ADF** Finds all the roots of a complex polynomial equation, using the method of Grant and Hitchins.

*PORT Subprogram Library*

**CPOLY** Finds the zeros of a polynomial with complex coefficients. Uses an inconvenient representation of two real arrays to represent complex numbers.

*Scientific Desk PC Subprogram Library*

**F1A1B** Finds the zeros of a polynomial with complex coefficients.

## F1b : Solution of single general nonlinear equations

*Collected Algorithms of the ACM*

**A631** ZERO1 and ZERO2: Fortran subroutines for finding a bracketed zero by Larkin's method of rational interpolation. ZERO2 requires derivative; ZERO1 does not. (See V. Nortin, ACM TOMS 11 (1985) pp. 120-134.)

*CMLIB Library (ZEROIN Sublibrary)*

**ZEROIN** Finds a zero of a user defined function on a given interval in which the function changes sign.

*IMSL Subprogram Library*

**ZANLYT** Finds zeros of an analytic complex function using the Muller method with deflation.

**ZBRENT** Zero of a function which changes sign in a given interval (Brent algorithm).

**ZFALSE** Finds a zero of a function given an interval containing the zero.

**ZREAL1** Finds real zeros of a real function. (To be used when initial guesses are poor.)

**ZREAL2** Finds real zeros of a real function. (To be used when initial guesses are good.)

*IMSL MATH/LIBRARY Subprogram Library*

**ZANLY** Finds a zero of a univariate analytic function.

**ZBREN** Finds a zero of a function which changes sign in a given interval.

**ZREAL** Finds a real zero of a real function.

*NAG Subprogram Library*

**C05ADF** Locates a zero of a continuous function in a given interval (a,b) by a combination of the methods of linear interpolation, extrapolation and bisection.

**C05AGF** Locates a simple zero of a continuous function from a given starting value, using a binary search to locate an interval containing a zero of the function, then a combination of the methods of linear interpolation, extrapolation and bisection to locate the zero precisely.

**C05AJF** Attempts to locate a zero of a continuous function by a continuation method using a secant iteration.

**C05AVF** Attempts to locate an interval containing a simple zero of a continuous function f(x) using a binary search. It uses 'reverse communication' for evaluating the function.

**C05AXF** Attempts to locate a zero of a continuous function using a continuation method based on a secant iteration.

**C05AZF** Locates a simple zero of a continuous function on a given interval by a combination of the methods of linear interpolation, linear extrapolation and bisection. It uses reverse communication for evaluating the function.

*NASHLIB Subprogram Library*

**A18** Finds roots using false position and bisection. Illustrates use of algorithm A18RF. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 136-137.)

*NMS Subprogram Library*

**FZERO** Finds zero of REAL FUNCTION F(X) on interval [B,C] to relative and absolute accuracy RE, AE. Requires EXTERNAL statement for user selected function name. R is input estimate of location of zero, if this is known. Best output estimate of zero returned in B.

*PORT Subprogram Library*

**ZERO** Finds a single real root of a function within an interval specified by the user.

*Scientific Desk PC Subprogram Library*

**F1A21** Searches for a zero of a user-defined function between given input values until the width of the interval has collapsed to within a tolerance set by the user.

**F1A22** Searches for a zero of a user-defined function between given input values until the width of the interval has collapsed to within a tolerance set by the user. User can specify if the routine is allowed to search outside the input interval in the case that there is no sign change in that interval.

*SCRUNCH Subprogram Library*

**ZEROIN** Finds a root of a nonlinear user-supplied function given an initial interval in which the function changes sign. In BASIC.

## F2: System of equations

Solving a system of nonlinear equations,

$$F(x) = 0,$$

where $x$ is a vector and $F$ is a vector-valued function, both of length $n$, is not easy unless the user can provide a good estimate of the solution. Usually the user provides this estimate, a Fortran SUBROUTINE subprogram to calculate $F$ for any given $x$, and a specification of the accuracy required. The software then constructs a sequence of $x$'s which may converge to a solution. Some programs stop when $\| F(x) \|$ is small enough, and others stop when $x$ is believed to be close to a root. Some programs interpret $\| F(x) \|$ as the largest component of $F$, others as the square root of the sum of squares of the components. Tolerances might be absolute or relative, depending on the program. Programs vary as to when they quit if the process does not seem to be converging.

Most algorithms to solve systems of nonlinear equations utilize Jacobian matrix, the matrix of first partial derivatives of the components of $F$ with respect to the components of $x$. Some programs allow the user to provide this matrix, while others compute approximations to it.

In Fortran subprograms that allow the user to specify a name for the SUBROUTINE subprogram which evaluates $F$ the user should be sure that this name appears

1. in an EXTERNAL statement in the main program,
2. in the subroutine CALL, and
3. as the name of a SUBROUTINE.

## References

[AG80] E. Allgower and K. Georg. Simplicial and continuation methods for approximating fixed points and solutions to systems of equations. *SIAM Review*, 22(1):28–85, 1980.

[DS83] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[MC79] J. J. More and M. Y. Cosnard. Numerical solution of nonlinear equations. *ACM Transactions on Mathematical Software*, 5(1):64–85, 1979.

[OR70] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, New York, 1970.

[Rhe74] W. C. Rheinboldt. *Methods for Solving Systems of Nonlinear Equations.* SIAM, Philadelphia, 1974.

[Sar81] R. W. Sargent. A review of methods for solving nonlinear algebraic equations. In R. S. H. Mah and W. D. Seider, editors, *Foundations of Computer Aided Chemical Process Design*, volume 1. Engineering Foundation, New York, 1981.

**F2 : Solution of a system of nonlinear equations**

*Collected Algorithms of the ACM*

**A502** DERPAR: a Fortran subprogram for the evaluation of the dependence of the solution of a nonlinear system on a parameter. The modified method of Davidenko, which applies the implicit function theorem, is used in combination with Newton's method and Adam's integration formulas. (See M. Kubicek, ACM TOMS 2 (1976) pp. 98-107.)

**A554** BRENTM: a Fortran subroutine for solving a system of nonlinear equations in n variables by using a modification of Brent's method. (See J.J. More and M.Y. Cosnard, ACM TOMS 6 (1980) pp. 240-251.)

**A555** FIXPT: a set of Fortran subroutines for computing fixed points or zeros of an n-dimensional vector function f(x). For the fixed point problem f is assumed to be a $C^2$ map of some ball into itself. For the zero finding problem, f is assumed to be a $C^2$ map such that for some $r>0$, $xf(x)\geq 0$ whenever $||x||=r$. (See L.T. Watson and D. Fenner, ACM TOMS 6 (1980) pp. 252-259.)

**A596** PITCON: a Fortran subprogram for continuation. Computes a set of solutions to nonlinear system of equations containing a parameter. On such a solution curve the exact location of target points where a given variable has a specified value can be located. Limit points are also identified. Based upon a local parameterization which uses curvature estimates to control the choice of parameter value. (See W.C. Rheinboldt and J.V. Burkardt, ACM TOMS 9 (1983) pp. 236-241.)

**A617** DAFNE: a Fortran subprogram for solving nonlinear systems based on the numerical solution of a Cauchy problem for a system of ordinary differential equations inspired by classical mechanics. (See F. Aluffi-Pentini, V. Parisi, and F. Zirilli, ACM TOMS 10 (1984) pp. 317-324)

**A652** HOMPACK: Fortran routines for globally convergent homotopy algorithms, for finding zeros or fixed points of nonlinear systems of equations. (See L. T. Watson, S. C. Billups, and A. P. Morgan, ACM TOMS 13 (1987) pp. 281-310.)

*CMLIB Library (SNLS1E Sublibrary)*

**SNSQ** Finds a zero of a system of N nonlinear equations in N variables by a modification of the Powell hybrid method. Flexible usage.

• **SNSQE** Finds a zero of a system of N nonlinear equations in N variables by a modification of Powell's hybrid method. An easy to use driver for SNSQ.

**SOS** Finds a zero of a system of N nonlinear equations in N unknowns using Brown's method.

*IMSL Subprogram Library*

**ZSCNT** Solve a system of nonlinear equations.

**ZSPOW** Solve a system of nonlinear equations (uses function values only).

*IMSL MATH/LIBRARY Subprogram Library*

**NEQNF** Solve a system of nonlinear equations using the Levenberg-Marquardt algorithm and a finite-difference Jacobian.

**NEQNJ** Solve a system of nonlinear equations using the Levenberg-Marquardt algorithm with a user-supplied Jacobian.

*NAG Subprogram Library*

• **C05NBF** Is an easy-to-use routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method.

**C05NCF** Is a comprehensive routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method.

• **C05PBF** Is an easy-to-use routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method. The user must provide the Jacobian.

**C05PCF** Is a comprehensive routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method. The user must provide the Jacobian.

*NMS Subprogram Library*

**SNSQE** Solves N nonlinear equations given in SUBROUTINE FCN(N,X,FVEC, IFLAG) to accuracy TOL. If IOPT=2 then Jacobian subroutine JAC need not be supplied. Needs EXTERNAL statement for user selected subroutine name. NMPINT=0 for simple use. Returns solution in X(N), residuals in FVEC(N).

*PORT Subprogram Library*

• **ZONE** Finds a solution of a system of non-linear equations.

**ZONEJ** Finds a solution of a system of non-linear equations. User must provide a SUBROUTINE to compute the Jacobian matrix

• **easy-to-use version**

*Scientific Desk PC Subprogram Library*

F2B Solves a system of N simultaneous nonlinear equations. Uses an iterative method which is a variation of Newton's method using Gaussian elimination in a manner similar to the Gauss-Seidel process.

• F2BE Solves a system of N simultaneous nonlinear equations. (Easy to use.) Uses an iterative method which is a variation of Newton's method using Gaussian elimination in a manner similar to the Gauss-Seidel process.

F2BP Solves a system of N simultaneous nonlinear equations. Uses a modification of the Powell hybrid method. Jacobian can be provided by the user or approximated by the program.

• F2BPE Solves a system of N simultaneous nonlinear equations. (Easy to use.) Uses a modification of the Powell hybrid method. Jacobian can be provided by the user or approximated by the program.

**F3 : Service routines for solution of nonlinear equations (e.g., check user-supplied derivatives)**

*Collected Algorithms of the ACM*

A566 Fortran subroutines for testing unconstrained optimization software in three areas: zeros of systems of N nonlinear functions in n variables, least squares minimization of M nonlinear functions in N variables, and unconstrained minimization of an objective function with N variables. (See J.J. More, B.S. Garbow, and K.E. Hillstrom, ACM TOMS 7 (1981) pp. 136-140.)

A618 DSM and FDJS: Fortran subroutines for estimating sparse Jacobian matrices. (See T.J. Coleman, B.S. Garbow, and J.J. More, ACM TOMS 10 (1984) pp. 346-347.)

*IMSL Subprogram Library*

ZSRCH Generate starting points in an N dimensional space for algorithms which optimize functions of several variables or algorithms which solve simultaneous nonlinear equations.

*NAG Subprogram Library*

C05ZAF Checks the user-provided gradients of M non-linear functions in N variables for consistency with the functions themselves. The routine must be called twice.

E04HCF Checks that a user-supplied routine for evaluating an objective function and its first derivative values are consistent with the function values calculated.

E04HDF Checks that a user-supplied routine for calculating second derivatives of an objective function is consistent with a user-supplied routine for calculating the corresponding first derivatives.

*Scientific Desk PC Subprogram Library*

F1A1E Evaluates a complex polynomial and its derivatives, with optional error bounds.

# G: Optimization

The general problem addressed in this class is that of minimizing or maximizing a function $f(x)$ of one or more variables $x^T = (x_1, x_2, \ldots, x_n)$. In addition, the values of $x$ which are allowed might be required to satisfy an expression such as $c(x) = 0$, or $d(x) > 0$, or that the $x_i$ can assume only integer values. Such expressions are referred to as constraints, and a number of constraints are referred to collectively as the constraint set $C$ (e.g. $C = \{c_i(x) = 0, x_j > 0, \text{ for } i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n\}$).

Solution methods depend upon the functional form of $f$ and the constraint set $C$. For example, if $f$ and each constraint of $C$ are linear, then *linear programming* (LP) techniques can be applied. Problems with tens of thousands of variables and thousands of constraints can be solved using these techniques. If, in addition, the constraint set can be formulated as a *network* (arcs representing variables and nodes representing constraints) then network algorithms can be used to solve problems having millions of arcs and thousands of nodes.

Once the problem becomes nonlinear, that is, either the objective function $f$ or one or more of the constraints is nonlinear, then computational effort increases and choosing an algorithm becomes more difficult. Different algorithms are designed for solving various classes of *nonlinear programming* problems, such as unconstrained optimization problems, problems with inequality constraints, problems with equality constraints, and problems with both types of constraints. Within each of these categories, different algorithms make specific assumptions about the problem structure. For example, in unconstrained optimization, some procedures assume that the objective function is differentiable and use gradient values, $\partial f/\partial x_i$, whereas other algorithms do not make this assumption and rely primarily on function evaluations. (Algorithms requiring derivatives often provide the option of finite difference approximations to derivatives.) For problems with equality constraints, some algorithms can only handle linear constraints, while others can handle nonlinear constraints as well.

In general, one should use an algorithm which exploits the structure of the problem. Thus, a network code is preferred over a general linear programming code when possible, especially for problems having all integer data. Problems having only linear constraints should be solved by an algorithm which fully exploits this structure. Problems for which one can provide derivatives should be solved by algorithms that capitalize on this. Similarly, it is much more efficient to use an algorithm that exploits least-squares structure when it exists. Other types of problems for which specific techniques have been developed include least absolute value, quadratic objective function with linear constraints, geometric programming problems, linear-fractional objective function with linear constraints, fixed point problems, shortest path, longest path, and minimum spanning tree problems, and a variety of integer programming problems where the linear constraint set has a specific structure.

Finally, inherent structure in the underlying application can sometimes be exploited to create a more tractable mathematical model. Nevertheless, there are a variety of special-purpose algorithms that can be constructed for quickly giving good approximate solutions with known error bounds for some difficult problems.

## References

[BS79] M. S. Bazaraa and C. M. Shetty. *Nonlinear Programming.* John Wiley & Sons, New York, 1979.

[DS83] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[Gas85] S. I. Gass. *Linear Programming: Methods and Applications.* McGraw-Hill, New York, 5th edition, 1985.

[Gil79] P. E. Gill et al. The design and structure of a Fortran program library for optimization. *ACM Transactions on Mathematical Software*, 5(3):259–283, 1979.

[GMW81] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, New York, 1981.

[Hie81] K. L. Hiebert. An evaluation of mathematical software that solves nonlinear least squares problems. *ACM Transactions on Mathematical Software*, 7(1):1–16, 1981.

[HL80] F. S. Hillier and G. J. Lieberman. *Introduction to Operations Research.* Holden-Day Inc., San Francisco, 1980.

[Hu82] T. C. Hu. *Combinatorial Algorithms.* Addison-Wesley, Menlo Park, 1982.

[Lue84] D. G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, Reading, MA, 2nd edition, 1984.

[Wag75] H. M. Wagner. *Principles of Management Science.* Prentice-Hall, Englewood Cliffs, NJ, 1975.

**G1a1a : Unconstrained optimization of a smooth univariate function, user provides no derivatives**

*IMSL Subprogram Library*

**ZXLSF** One-dimensional minimization of a smooth function using safeguarded quadratic interpolation.

*IMSL MATH/LIBRARY Subprogram Library*

**UVMIF** Find the minimum point of a smooth function of a single variable using only function evaluations.

*IMSL STAT/LIBRARY Subprogram Library*

**UMINF** Minimize a function of N variables using a quasi-Newton method and a finite-difference gradient.

*NAG Subprogram Library*

**E04ABF** Searches for a minimum, in a given finite interval, of a continuous function of a single variable, using function values only. The method (based on quadratic interpolation) is intended for functions which have a continuous first derivative (although it will usually work if the derivative has occasional discontinuities).

*NASHLIB Subprogram Library*

**A17** Line search using parabolic inverse interpolation. Illustrates use of algorithm A17LS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 117-118.

*NMS Subprogram Library*

**FMIN** Finds minimum of REAL FUNCTION F(X) on interval [AX,BX] to accuracy EPS. Needs EXTERNAL statement for user selected function name. Returns FMIN as location of minimum with AX,BX bracketing this value.

*SCRUNCH Subprogram Library*

**FNM** Calculates an approximation to the point where a user defined function attains a minimum on a given interval. In BASIC.

**G1a1b : Unconstrained optimization of a smooth univariate function, user provides first derivatives**

*IMSL MATH/LIBRARY Subprogram Library*

**UVMID** Find the minimum point of a smooth function of a single variable using both function evaluations and first derivative evaluations.

*NAG Subprogram Library*

**E04BBF** Searches for a minimum, in a given finite interval, of a continuous function of a single variable, using function and first derivative values. The method (based on cubic interpolation) is intended for functions which have a continuous first derivative (although it will usually work if the derivative has occasional discontinuities).

**G1a2 : Unconstrained optimization of a general univariate function (no smoothness assumed)**

*IMSL Subprogram Library*

**ZXGSN** One-dimensional unimodal function minimization using the golden section search method.

**ZXGSP** One-dimensional unimodal function minimization using the golden section search method (data parameters specified).

*IMSL MATH/LIBRARY Subprogram Library*

**UVMGS** Find the minimum point of a nonsmooth function of a single variable.

*NASHLIB Subprogram Library*

**A16** Perform a grid or equal-interval line search. Illustrates use of algorithm A16GS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, page 125.)

**A17** Line search using parabolic inverse interpolation. Illustrates use of algorithm A17LS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 117-118.)

*PORT Subprogram Library*

**EXTRMI** Finds extremal points of an integer function defined on a mesh.

**EXTRMR** Finds extremal points of a real function defined on a mesh.

**FMIN** Finds an approximate local minimum of a univariate user defined function.

**G1b1a : Unconstrained optimization of a smooth multivariate function, user provides no derivatives**

*Collected Algorithms of the ACM*

**A630** BBVSCG: a variable storage Fortran subprogram for function minimization. (See A. Buckley and A. Lenir, ACM TOMS 11 (1985) pp. 103-119.)

*CMLIB Library (NL2SN Sublibrary)*

**SMSNO** Minimize a general unconstrained objective function using finite difference gradients and secant Hessian approximations.

*CMLIB Library (UNCMIN Sublibrary)*

- **OPTIF0** Solves the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. Easy to use version of OPTIF9. User has no control of options.

**OPTIF9** Solves the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. User has full control of options.

*IMSL Subprogram Library*

**ZXCGR** A conjugate gradient algorithm for finding the minimum of a function of n variables.

**ZXMIN** Minimum of a function of n variables using a quasi-Newton method.

*IMSL MATH/LIBRARY Subprogram Library*

**UMCGF** Minimize a function of N variables using a conjugate gradient algorithm and a finite-difference gradient.

**UMINF** Minimize a function of N variables using a quasi-Newton method and a finite-difference gradient.

**UNLSF** Solve a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.

*NAG Subprogram Library*

- **E04CGF** Is an easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a function of N independent variables using function values only. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04JBF** Is a comprehensive quasi-Newton algorithm for finding a minimum of a function of several variables, optionally subject to fixed upper and/or lower bounds on the variables. The user does not need to supply any derivatives of the function. The routine is intended for functions which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

*NMS Subprogram Library*

**UNCMIN** Finds minimum of N-variable function given initial values X0(N). Returns solution in X(N) and minimum function value in F. Needs EXTERNAL statement for user selected SUBROUTINE FCN(N,X,F) returning value of F at X(N).

## G1b1b : Unconstrained optimization of a smooth multivariate function, user provides first derivatives

*Collected Algorithms of the ACM*

**A500** MINI: a subprogram for finding an unconstrained minimum of a multivariate function f(x) to within the accuracy $||g(x))|| < \epsilon$, where g(x) is the gradient of f(x) and $\epsilon$ is the required accuracy provided by the user. Based on a quasi-Newton method suggested by Shanno and Phua. (See D.F. Shanno and K.H. Phua, ACM TOMS 1 (1975) pp. 87-94.)

**A611** SMSNO, SUMSL, and HUMSL: Fortran subprograms for solving general unconstrained minimization problems using a model/trust-region approach. The routines offer the option of providing function and gradient, or function, gradient, and Hessian values by reverse communication. (See D.M. Gay, ACM TOMS 9 (1983) pp. 503-524.)

**A630** BBVSCG: a variable storage Fortran subprogram for function minimization. (See A. Buckley and A. Lenir, ACM TOMS 11 (1985) pp. 103-119.)

*CMLIB Library (NL2SN Sublibrary)*

**SUMSL** Minimizes a general unconstrained objective function using analytic gradient and a Hessian approximation from a secant update.

*CMLIB Library (UNCMIN Sublibrary)*

**OPTIF9** Solves the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. User has full control of options.

*IMSL MATH/LIBRARY Subprogram Library*

**UMCGG** Minimize a function of N variables using a conjugate gradient algorithm and a user-supplied gradient.

**UMIDH** Minimize a function of N variables using a modified Newton method and a finite-difference Hessian.

**UMING** Minimize a function of N variables using a quasi-Newton method and a user-supplied gradient.

**UNLSJ** Solve a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

*NAG Subprogram Library*

**E04DBF** Minimizes a general function F(X) of N independent variables by the conjugate gradient method due to Fletcher and Reeves. Formulae to calculate the value of the function and its first derivatives must be supplied.

**E04DEF** Is an easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a function of N independent variables when first derivatives are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04DFF** Is an easy-to-use modified-Newton algorithm for finding an unconstrained minimum of a function of N independent variables when first derivatives are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04DGF** Minimizes an unconstrained nonlinear function of several variables using a pre-conditioned, limited memory quasi-Newton conjugate gradient method. The routine is intended for use on large scale problems.

*NASHLIB Subprogram Library*

**A21** Finds the minimum of a function of n variables using a variable metric algorithm. Illustrates use of algorithm A21VM. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 159-160.)

**A22** Finds the minimum of a function of n variables using a conjugate gradient algorithm. Illustrates use of algorithm A22CGM. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 165-167.)

**G1b1c : Unconstrained optimization of a smooth multivariate function, user provides first and second derivatives**

*Collected Algorithms of the ACM*

**A611** SMSNO, SUMSL, and HUMSL: Fortran subprograms for solving general unconstrained minimization problems using a model/trust-region approach. The routines offer the option of providing function and gradient, or function, gradient, and Hessian values by reverse communication. (See D.M. Gay, ACM TOMS 9 (1983) pp. 503-524.)

• **easy-to-use version**

*CMLIB Library (NL2SN Sublibrary)*

**HUMSL** Minimizes a general unconstrained objective function using (analytic) gradient and Hessian provided by the user.

*CMLIB Library (UNCMIN Sublibrary)*

**OPTIF9** Solves the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. User has full control of options.

*IMSL MATH/LIBRARY Subprogram Library*

**UMIAH** Minimize a function of N variables using a modified Newton method and a user-supplied Hessian.

*NAG Subprogram Library*

**E04EBF** Is an easy-to-use modified-Newton algorithm for finding an unconstrained minimum of a function of N independent variables when first and second derivatives are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

## G1b2 : Unconstrained optimization of a general multivariate function (no smoothness assumed)

*IMSL MATH/LIBRARY Subprogram Library*

**UMPOL** Minimize a function of N variables using a direct search polytope algorithm.

*NAG Subprogram Library*

**E04CCF** Minimizes a general function F(X) of N independent variables X = (X(1),X(2),...,X(N))(T) by the Simplex method. Derivatives of the function need not be supplied.

*NASHLIB Subprogram Library*

**A19A20** Finds the minimum of a function of n variables using a Nelder Mead algorithm and axial search. Illustrates use of algorithms A19NM and A20AS. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 146-149.)

## G2a1 : Linear programming with a dense matrix of constraints

*Collected Algorithms of the ACM*

**A557** PAGP: a partitioning algorithm for linear goal programming problems. (See J.L. Arthur and A. Ravindran, ACM TOMS 6 (1980) pp. 429.)

*IMSL Subprogram Library*

**ZX0LP** Solve the linear programming problem (phase one or phase two) via the revised simplex algorithm.

• **ZX3LP** Solve the linear programming problem via the revised simplex algorithm. (Easy to use version.)

• **ZX4LP** Solve the linear programming problem via the revised simplex algorithm. (Alternate easy to use version.)

*IMSL MATH/LIBRARY Subprogram Library*

**DLPRS** Solve a linear programming problem via the revised simplex algorithm.

*NAG Subprogram Library*

• **E04MBF** Is an easy-to-use routine for solving linear programming problems, or for finding a feasible point for such problems. It is not intended for large sparse problems.

**E04NAF** Is a comprehensive routine for solving quadratic programming (QP) or linear programming (LP) problems. It is not intended for large sparse problems.

**E04NCF** Solves linearly constrained linear least-squares problems and convex quadratic programming problems. E04NCF may also be used to solve linear programming problems and to find a feasible point with respect to a set of linear inequality constraints. E04NCF treats all matrices as dense and hence is not intended for large sparse problems.

*Scientific Desk PC Subprogram Library*

**G2A1** Solves the linear programming problem: minimize (Transpose of costs) × X, subject to the constraint equations AX+W=R and bounds on the components of X and W.

## G2a2 : Linear programming with a sparse matrix of constraints

*CMLIB Library (SPLP Sublibrary)*

**SPLP** Solves linear optimization problems, that is, it minimizes the linear function (Transpose of costs) × X subject to AX=W, where the entries of the vectors X and W may have simple upper or lower bounds. Uses a sparse storage mode for the matrix A and out-of-core scratch storage.

## G2b : Transportation and assignments problem

*Collected Algorithms of the ACM*

**A548** ASSCT: a Fortran subroutine for solving the square assignment problem. (See G. Carpaneto and P. Toth, ACM TOMS 6 (1980) pp. 104-111.)

**A608** HGW: A Fortran subprogram to compute an approximate solution to the extended Koopmans-Beckmann quadratic assignment problem. (See D.H. West, ACM TOMS 9 (1983) pp. 461-466.)

*NAG Subprogram Library*

**H03ABF** Solves the classical Transportation ( Hitchcock ) problem.

## G2c3 : Knapsack problems in integer programming

*Collected Algorithms of the ACM*

**A632** MKP: a Fortran subroutine for the 0-1 multiple knapsack problem. (See S. Martello and P. Toth, ACM TOMS 11 (1985) pp. 135-140.)

## G2c5 : Routing, scheduling, and location problems in integer programming

*Collected Algorithms of the ACM*

**A520** ARSME: a Fortran subroutine for solving a resource constrained network scheduling problem for the case in which the activities may be arbitrarily interrupted and restarted later with no

• **easy-to-use version**

increase in activity duration. The method used is an automatic revised simplex method. (See J. Weglarzet et al., ACM TOMS 3 (1977) pp. 295-300.)

**A558** LOCATE: a Fortran subroutine for the one-dimensional multifacility location problem with rectilinear distance by a minimum-cut approach. (See T. Cheung, ACM TOMS 6 (1980) pp. 430-431.)

## G2c6 : Pure integer programming

*NAG Subprogram Library*

**H02BAF** Solves the integer linear programming problem, with all integer coefficients, via Gomory's method. It is enhanced by including the technique known as Wilson's cuts.

## G2d1 : Shortest path network optimization

*Collected Algorithms of the ACM*

**A562** A Fortran program for calculating the shortest path length from a specific node to all other nodes in a network. (See U. Pape, ACM TOMS 6 (1980) pp. 450-455.)

## G2d2 : Minimum spanning tree network optimization

*Collected Algorithms of the ACM*

**A613** MSTPAC: a Fortran subroutine for calculating the minimum spanning tree for moderate integer weights in a connected undirected graph represented in a forward star data structure. (See R.E. Haymond, J.P. Jarvis, and D.R. Shier, ACM TOMS 10 (1984) pp. 108-111.)

## G2e1 : Quadratic programming for positive definite Hessians (i.e., convex problems)

*Collected Algorithms of the ACM*

**A559** HSQP: a subroutine for computing the stationary point of a quadratic function of n variables subject to linear constraints. (See J.T. Betts, ACM TOMS 6 (1980) pp. 432-436.)

*IMSL MATH/LIBRARY Subprogram Library*

**QPROG** Solve a quadratic programming problem subject to linear equality/ inequality constraints.

*NAG Subprogram Library*

**E04NAF** A comprehensive routine for solving quadratic programming (QP) or linear programming (LP) problems. It is not intended for large sparse problems.

**E04NCF** Solves linearly constrained linear least-squares problems and convex quadratic programming problems. E04NCF may also be used to solve linear programming problems and to find a feasible point with respect to a set of linear inequality constraints. E04NCF treats all matrices as dense and hence is not intended for large sparse problems.

**G2e2 : Quadratic programming for indefinite Hessians**

*Collected Algorithms of the ACM*

**A559** HSQP: a subroutine for computing the stationary point of a quadratic function of n variables subject to linear constraints. (See J.T. Betts, ACM TOMS 6 (1980) pp. 432-436.)

*NAG Subprogram Library*

**E04NAF** A comprehensive routine for solving quadratic programming (QP) or linear programming (LP) problems. It is not intended for large sparse problems.

**G2h1a1 : General nonlinear optimization with simple bounds of a smooth function, user provides no derivatives**

*IMSL Subprogram Library*

**ZXMWD** Global minimum (with constraints) of a function of n variables.

*NAG Subprogram Library*

• **E04JAF** An easy-to-use quasi-Newton algorithm for finding a minimum of a function, subject to fixed upper and lower bounds of the independent variables, using function values only.

**E04JBF** A comprehensive quasi-Newton algorithm for finding a minimum of a function of several variables, optionally subject to fixed upper and/or lower bounds on the variables. The user does not need to supply any derivatives of the function. The routine is intended for functions which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04UCF** Minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. The user may provide subroutines that define the objective and constraint functions and as many of their first partial derivatives as possible. Unspecified derivatives are approximated by finite differences. All matrices are treated as dense, and hence E04UCF is not intended for large sparse problems.

**G2h1a2 : General nonlinear optimization with simple bounds of a smooth function, user provides first derivatives**

*IMSL MATH/LIBRARY Subprogram Library*

**BCODH** Minimize a function of N variables subject to bounds on the variables using a modified Newton method and a finite-difference Hessian.

**BCONG** Minimize a function of N variables subject to bounds on the variables using a quasi-Newton method and a user-supplied gradient.

*NAG Subprogram Library*

• **E04KAF** An easy-to-use quasi-Newton algorithm for finding a minimum of a function subject to fixed upper and lower bounds on the independent variables when first derivatives of F are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04KBF** A comprehensive quasi-Newton algorithm for finding a minimum of a function of several variables optionally subject to a fixed upper and/or lower bounds on the variables. First derivatives are required. The routine is intended for functions which have continuous first and second

• **easy-to-use version**

derivatives (although it will usually work even if the derivatives have occasional discontinuities).

- **E04KCF** An easy-to-use modified-Newton algorithm for finding a minimum of a function, subject to fixed upper and lower bounds on the independent variables, when first derivatives of F are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04KDF** A comprehensive modified Newton algorithm for finding: – an unconstrained minimum of a function of several variables – a minimum of a function of several variables subject to fixed upper and/or lower bounds on the variables.

**E04UCF** Minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. The user may provide subroutines that define the objective and constraint functions and as many of their first partial derivatives as possible. Unspecified derivatives are approximated by finite differences. All matrices are treated as dense, and hence E04UCF is not intended for large sparse problems.

**E04VCF** A comprehensive routine to minimize an arbitrary smooth function subject to constraints, including simple bounds, linear constraints and smooth nonlinear constraints. The user must provide subroutines that define the objective, constraints, and their gradients. All matrices are treated as dense. Uses a sequential quadratic programming algorithm.

- **E04VDF** An easy-to-use routine designed to minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. (E04VDF may also be used for unconstrained, bound-constrained and linearly constrained optimization.) The user must provide subroutines that define the objective and constraint functions and their gradients. All matrices are treated as dense, and hence E04VDF is not intended for large sparse problems.

**G2h1a3 : General nonlinear optimization with simple bounds of a smooth function, user provides first and second derivatives**

*IMSL MATH/LIBRARY Subprogram Library*

**BCOAH** Minimize a function of N variables subject to bounds on the variables using a modified Newton method and a user-supplied Hessian.

*NAG Subprogram Library*

- **E04LAF** An easy-to-use modified-Newton algorithm for finding a minimum of a function, subject to fixed upper and lower bounds on the independent variables, when first and second derivatives of F are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**E04LBF** A comprehensive modified Newton algorithm for finding a minimum of a function of several variables optionally subject to fixed upper and/or lower bounds on the variables. First and second derivatives are required. The routine is intended for functions which have continuous first and second derivative (although it will usually work even if the derivatives have occasional discontinuities).

**G2h1b : General nonlinear optimization with simple bounds of a general function (no smoothness assumed)**

*IMSL MATH/LIBRARY Subprogram Library*

**BCPOL** Minimize a function of N variables subject to bounds on the variables using a direct search Complex algorithm.

**G2h2a1 : General nonlinear optimization with linear equality or inequality constraints of a smooth function, user provides no derivatives**

*NAG Subprogram Library*

**E04UCF** Minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. The user may provide subroutines that define the objective and constraint functions and as many of their first partial derivatives as possible. Unspecified derivatives are approximated by finite differences. All matrices are treated as dense, and hence E04UCF is not intended for large sparse problems.

**G2h2a2 : General nonlinear optimization with linear equality or inequality constraints of a smooth function, user provides first derivatives**

*NAG Subprogram Library*

**E04UCF** Minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. The user may provide subroutines that define the objective and constraint functions and as many of their first partial derivatives as possible. Unspecified derivatives are approximated by finite differences. All matrices are treated as dense, and hence E04UCF is not intended for large sparse problems.

**E04VCF** A comprehensive routine to minimize an arbitrary smooth function subject to constraints, including simple bounds, linear constraints and smooth nonlinear constraints. The user must provide subroutines that define the objective, constraints, and their gradients. All matrices are treated as dense. Uses a sequential quadratic programming algorithm.

• **E04VDF** An easy-to-use routine designed to minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. (E04VDF may also be used for unconstrained, bound-constrained and linearly constrained optimization.) The user must provide subroutines that define the objective and constraint functions and their gradients. All matrices are treated as dense, and hence E04VDF is not intended for large sparse problems.

**G2h3a1 : General nonlinear optimization of a smooth function with only smooth equality nonlinear constraints**

*NAG Subprogram Library*

**E04UCF** Minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. The user may provide subroutines that define the objective and constraint functions and as many of their first partial derivatives as possible. Unspecified derivatives are approximated by finite differences. All matrices are treated as dense, and hence E04UCF is not intended for large sparse problems.

**G2h3b1a : General nonlinear optimization of a smooth function with smooth equality and inequality nonlinear constraints, user provides no derivatives**

• **easy-to-use version**

*IMSL MATH/LIBRARY Subprogram Library*

**NCONF** Solve a general nonlinear programming problem using the successive quadratic programming algorithm and a finite difference gradient.

*NAG Subprogram Library*

**E04UAF** Attempts to find a minimum of a function of several variables subject to fixed bounds on the variables and to general equality and/or inequality constraints. A sequential augmented Lagrangian method is used, the minimization subproblems involved being solved by a quasi-Newton method. No derivatives are required. The routine is intended for functions and constraints which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

**G2h3b1b : General nonlinear optimization of a smooth function with smooth equality and inequality nonlinear constraints, user provides first derivatives of function and constraints**

*IMSL MATH/LIBRARY Subprogram Library*

**NCONG** Solve a general nonlinear programming problem using the successive quadratic programming algorithm and a user-supplied gradient.

*NAG Subprogram Library*

**E04VCF** A comprehensive routine to minimize an arbitrary smooth function subject to constraints, including simple bounds, linear constraints and smooth nonlinear constraints. The user must provide subroutines that define the objective, constraints, and their gradients. All matrices are treated as dense. Uses a sequential quadratic programming algorithm.

• **E04VDF** An easy-to-use routine designed to minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. ( E04VDF may also be used for unconstrained, bound-constrained and linearly constrained optimization.) The user must provide subroutines that define the objective and constraint functions and their gradients. All matrices are treated as dense, and hence E04VDF is not intended for large sparse problems.

**G2i : Global solution to nonconvex nonlinear optimization problems**

*Collected Algorithms of the ACM*

**A659** A Fortran implementation of Sobol's quasirandom sequence generator for multivariate quadrature and optimization. (See P. Bratley and B. L. Fox, ACM TOMS 14 (1988) pp. 88-100.)

**G3 : Optimal control**

*Collected Algorithms of the ACM*

**A640** SFRMG: A Fortran subprogram which takes real matrices A (n-by-n), B (n-by-m), and C (l-by-n) and forms the complex frequency response matrix CEB, where E is the inverse of (FREQ × I – A), I is the n-by-n identity matrix and FREQ is a complex scalar parameter taking values along the imaginary axis for continuous-time systems and on the unit circle for discrete-time systems. (See A.J. Laub, ACM TOMS 12 (1986) pp. 26-33.)

## G4c : Check user-supplied derivatives for optimization

*IMSL MATH/LIBRARY Subprogram Library*

**CHGRD** Check a user-supplied gradient of a function.

**CHHES** Check a user-supplied Hessian of an analytic function.

**CHJAC** Check a user-supplied Jacobian of a system of equations with M functions in N unknowns.

*NAG Subprogram Library*

**E04HCF** Checks that a user-supplied routine for evaluating an objective function and its first derivative values which are consistent with the function values calculated.

**E04HDF** Checks that a user-supplied routine for calculating second derivatives of an objective function is consistent with a user-supplied routine for calculating the corresponding first derivatives.

**E04YAF** Checks that a user-supplied routine for evaluating a vector of functions and the matrix of their first derivatives produces derivative values which are consistent with the function values calculated.

**E04YBF** Checks that a user-supplied routine for evaluating the second derivative term of the Hessian matrix of a sum of squares is consistent with a user-supplied routine for calculating the corresponding first derivatives.

**E04ZCF** Checks that user-supplied routines for evaluating an objective function, constraint functions and their first derivatives produce derivative values which are consistent with the function and constraint values calculated.

*Scientific Desk PC Subprogram Library*

**G4C** Checks the gradients of M nonlinear functions in N variables evaluated at a point X, for consistency with the functions themselves.

## G4d : Find feasible point for optimization

*IMSL Subprogram Library*

**ZSRCH** Generate starting points in an N dimensional space for algorithms which optimize functions of several variables or algorithms which solve simultaneous nonlinear equations.

*IMSL MATH/LIBRARY Subprogram Library*

**GGUES** Generate points in an N-dimensional space.

*NAG Subprogram Library*

• **E04MBF** Is an easy-to-use routine for solving linear programming problems, or for finding a feasible point for such problems. It is not intended for large sparse problems.

**E04NCF** Solves linearly constrained linear least-squares problems and convex quadratic programming problems. E04NCF may also be used to solve linear programming problems and to find a feasible point with respect to a set of linear inequality constraints. E04NCF treats all matrices as dense and hence is not intended for large sparse problems.

• **easy-to-use version**

## G4f : Other service routines for optimization

*Collected Algorithms of the ACM*

**A566** Fortran subroutines for testing unconstrained optimization software in three areas: zeros of systems of N nonlinear functions in n variables, least squares minimization of M nonlinear functions in N variables, and unconstrained minimization of an objective function with N variables. (See J.J. More, B.S. Garbow, and K.E. Hillstrom, ACM TOMS 7 (1981) pp. 136-140.)

**A636** DSSM and FDHS: Fortran subroutines for estimating sparse Hessian matrices. (See T.F. Coleman, B.S. Garbow, and J.J. More, ACM TOMS 11 (1985) pp. 363-377 and 378.)

*IMSL MATH/LIBRARY Subprogram Library*

**CDGRD** Approximate the gradient using central differences.

**FDGRD** Approximate the gradient using forward differences.

**FDHES** Approximate the Hessian using forward differences and function values.

**FDJAC** Approximate the Jacobian of M functions in N unknowns using forward differences.

**GDHES** Approximate the Hessian using forward differences and a user-supplied gradient.

*NAG Subprogram Library*

**E04DJF** Supplies optional parameters to NAG Fortran Library routine E04DGF from an external file.

**E04DKF** Supplies individual optional parameters to NAG Fortran Library routine E04DGF.

**E04HBF** Computes a sensible set of finite-difference intervals for input to a quasi-Newton minimization routine which does not require derivatives.

**E04NDF** Supplies optional parameters to NAG Fortran Library routine E04NCF from an external file.

**E04NEF** Supplies individual optional parameters to NAG Fortran Library routine E04NCF.

**E04UDF** Supplies optional parameters to NAG Fortran Library routine E04UCF from an external file.

**E04UEF** Supplies individual optional parameters to NAG Fortran Library routine E04UCF.

**E04XAF** Computes an approximation to the gradient vector and the Hessian matrix for use in conjunction with an optimization routine (such as E04UCF).

# H: Differentiation and Integration

This class contains programs for performing the fundamental operations of calculus, the computation of derivatives and the evaluation of definite integrals.

## H1: Numerical differentiation

This class contains programs which approximate derivatives of a function (given analytically or by data values) at one or more points. It does not include programs which first produce a fit if the fitted function can be subsequently differentiated. Programs which do this are classified in classes E and K.

Numerical differentiation often leads to serious rounding errors. If the function to be differentiated is available as a subprogram rather than as data values, adaptive procedures can be used to locate the best mesh to use in some interval. If the function can be extended to the complex plane then the most effective method is to represent the derivative by the Cauchy integral formula and use the trapezoidal quadrature rule to evaluate it.

### References

[LM67] J. N. Lyness and C. B. Moler. Numerical differentiation of analytic functions. *SIAM Journal on Numerical Analysis*, 4(2):202–210, 1967.

| **H1 :** | **Numerical differentiation** |
|---|---|

*DERIV Software Library*

**DIFF** A subroutine that computes the first, second or third derivative of a real function of a single real variable, where the function is defined by a Fortran FUNCTION subprogram. The method used is Neville's process of extrapolating from a sequence of interpolating polynomials with interpolating points distributed symmetrically about $x_0$ or, if this is not possible, to one side of $x_0$.

*IMSL Subprogram Library*

**DRVTE** Calculate first, second, or third derivative of a user-supplied function.

*IMSL MATH/LIBRARY Subprogram Library*

**DERIV** Compute the first, second or third derivative of a user-supplied function.

*NAG Subprogram Library*

**D04AAF** Calculates a set of derivatives (up to order 14) of a function of one real variable at a point, together with a corresponding set of error estimates, using an extension of the Neville algorithm.

*PORT Subprogram Library*

**CSPDI** Finds a numerical approximation to the first derivative at requested points in given input data by using spline interpolation.

## H2: Quadrature (numerical evaluation of definite integrals)

This class includes routines which evaluate the definite integral of a function which is either given explicitly or by means of a table of data. Programs can either be automatic or non-automatic.

Most routines for the integration of an explicit function require that the integrand be in the form of a Fortran function subprogram. In this case the exact function name must appear

1. in an EXTERNAL statement in the user's main program,
2. in the CALL statement which invokes the quadrature subroutine, and
3. as a Fortran FUNCTION itself.

The one exception to this is the very easy to use module Q1DA, which assumes that the function is called F, in which case item 1 is not needed.

Users must always be careful that their integrand function is defined wherever it might be evaluated. This depends on the specific routine but programs typically evaluate the integrand at the endpoints (in one dimension). If the integrand is singular (e.g., $1/\sqrt{x}$ at 0.0) the user must arbitrarily set its value at the singularity to prevent a divide by zero from occurring. Similarly, if the integrand has an apparent singularity (e.g., $\sin x/x$ at 0.0) the user must set the integrand value properly at this point (for this example $f(0) = 1$ is appropriate). A more correct solution is to use a program which does not perform endpoint evaluation. Sometimes the documentation omits this information. If internal singularities occur this same type of difficulty will arise. If the location of any singularities are known, it is highly recommended that the range of integration be split into pieces with the singularities only at the endpoints. This can be done automatically by some programs, for example, Q1DAX or QAGP by simply telling the integrator the locations of any problem points.

For "one shot" problems *adaptive* methods are very efficient in terms of human time. Adaption means that the integrand evaluation points are selected by the program and will vary from problem to problem. *Automatic* non-adaptive programs are also common. Some of these use a sequence of meshes, increasingly fine, and combine the results using an extrapolation algorithm. For the most part these techniques are not as efficient as the adaptive ones.

All automatic programs incur a penalty in terms of overhead, or surcharge. Non-automatic routines depend on a quadrature evaluation rule, a formula. Gauss quadrature rules are very efficient in terms of integrand evaluations but do not easily lend themselves to estimating errors. These formulas involve irrational points and weights which must be computed and tabulated in advance to be efficient. However they can save substantial computing time if the user thinks about the problem and experiments with different numbers of points. A number of routines which can generate and retrieve these formulas are listed in H2c. Programs which evaluate the formulas are listed in H2a1a2, H2a2a2, H2a3a2, H2b2a2 and H2b2b2.

Kronrod quadrature is a relatively new development. This involves the use of a pair of formulas, an $n$-point Gauss and a $2n + 1$ point Kronrod of which $n$ points are the original Gauss points. This allows for an accurate error estimate at modest cost. Many new programs use these formulas. See [Pie83] for details.

There are several programs for the integration of tabular, or gridded, data. The underlying idea is that some function is "fit" to the data and the function is then integrated. Some programs accept the user's data directly, such as CSPQU or PCHQA. Others require that the user input the form of the function, usually either a spline or a polynomial, such as PCHIA or E02AJF. In the second case most users will have to precede the call to the integrator by a call to a "fitter", either an interpolator from class E or an approximator from class K.

Programs also exist for multiple integration. For the integration over hyperrectangles, one can achieve 5–6 figure accuracy for smooth functions and 2–3 figure accuracy for non-smooth functions up to dimension 15. For integration over a bounded irregular region, one embeds this region inside a hyperrectangle and defines the integrand to be zero outside the region. In such cases a Monte Carlo type program, such as D01GBF, is recommended. One can also integrate over hyperspheres up to dimension 4 and over simplexes.

Two dimensional adaptive quadrature programs are also available. The overhead surcharge factor is even higher and there is much more scope for creativity by the problem originator. It has been common practice to use a pair of automatic one-dimensional programs to calculate iteratively a 2-D integral, see for example [KMN89].

## References

[DR84] P. J. Davis and P. Rabinowitz. *Methods of Numerical Integration.* Academic Press, New York, 2nd edition, 1984.

[KMN89] David Kahaner, Cleve Moler, and Stephen Nash. *Numerical Methods and Software.* Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[Lyn77] J. N. Lyness. Quid, quo, quadrature? In D. Jacobs, editor, *The State of the Art in Numerical Analysis*, pages 535–560. Academic Press, New York, 1977.

[Pie83] R. Piessens et al. QUADPACK. Springer-Verlag, 1983.

[Str71] A. Stroud. *Approximate Calculation of Multiple Integrals.* Prentice-Hall, Englewood Cliffs, NJ, 1971.

**H2a1a1 : Automatic 1-D finite interval quadrature (user need only specify required accuracy), integrand available via user-defined procedure**

*Collected Algorithms of the ACM*

**A614** INTHP: a Fortran subroutine for automatic numerical integration in $H_p$. The functions may have singularities at one or both endpoints of an interval. Each of finite, semi-infinite, and infinite intervals are admitted. (See K. Sikorski, F. Stenger, and J. Schwing, ACM TOMS 10 (1984) pp. 152-160.)

*CMLIB Library (Q1DA Sublibrary)*

• **Q1DA** Automatic integration of a user-defined function of one variable. Special features include randomization and singularity weakening.

**Q1DAX** Flexible subroutine for the automatic integration of a user-defined function of one variable. Special features include randomization, singularity weakening, restarting, specification of an initial mesh (optional), and output of smallest and largest integrand values.

**Q1DB** Automatic integration of a user-defined function of one variable. Integrand must be a Fortran FUNCTION but user may select name. Special features include randomization and singularity weakening. Intermediate in usage difficulty between Q1DA and Q1DAX.

*CMLIB Library (QUADPKS Sublibrary)*

• **QAG** Automatic adaptive integrator, will handle many non-smooth integrands using Gauss Kronrod formulas.

**QAGE** Automatic adaptive integrator, can handle most non-smooth functions, also provides more information than QAG.

• **QAGS** Automatic adaptive integrator, will handle most non-smooth integrands including those with endpoint singularities, uses extrapolation.

**QAGSE** Automatic adaptive integrator, can handle integrands with endpoint singularities, provides more information than QAGS.

**QNG** Automatic non-adaptive integrator for smooth functions, using Gauss Kronrod Patterson formulas.

*IMSL Subprogram Library*

**DCADRE** Numerical integration of a function using cautious adaptive Romberg extrapolation.

*IMSL MATH/LIBRARY Subprogram Library*

**QDAG** Integrate a function using a globally adaptive scheme based on Gauss-Kronrod rules.

**QDAGS** Integrate a function (which may have endpoint singularities).

**QDNG** Integrate a smooth function using a nonadaptive rule.

*IMSL STAT/LIBRARY Subprogram Library*

**QDAGS** Integrate a function (which may have endpoint singularities).

*JCAM Software Library*

**DEFINT** Uses double exponential transformation of Mori to compute definite integral automatically to user specified accuracy.

*NAG Subprogram Library*

**D01AHF** Computes a definite integral over a finite range to a specified relative accuracy using a method described by Patterson.

**D01AJF** Is a general-purpose integrator which calculates an approximation to the integral of a function F(x) over a finite interval (A,B).

**D01ARF** Computes definite and indefinite integrals over a finite range to a specified relative or absolute accuracy, using a method described by Patterson.

**D01BDF** Calculates an approximation to the integral of a function over a finite interval (A,B). It is non-adaptive and as such is recommended for the integration of smooth functions. These exclude integrands with singularities, derivative singularities or high peaks on (A,B), or which oscillate too strongly on (A,B).

*NMS Subprogram Library*

**Q1DA** Automatic integration of a user-defined function of one variable. Special features include randomization and singularity weakening.

*PORT Subprogram Library*

**ODEQ** Finds the integral of a set of functions over the same interval by using the differential equation solver ODES1. For smooth functions.

**QUAD** Finds the integral of a general user defined EXTERNAL function by an adaptive technique to given absolute accuracy.

**RQUAD** Finds the integral of a general user defined EXTERNAL function by an adaptive technique. Combined absolute and relative error control.

*Scientific Desk PC Subprogram Library*

• **H2A1** Automatically evaluates the definite integral of a user defined function of one variable.

**H2A1U** Automatically evaluates the definite integral of a user defined function of one variable.

*SCRUNCH Subprogram Library*

**SIMP** Calculates an estimate of the definite integral of a user supplied function by adaptive quadrature. In BASIC.

## H2a1a2 : Nonautomatic 1-D finite interval quadrature, integrand available via user-defined procedure

*CMLIB Library (QUADPKS Sublibrary)*

**QK15** Evaluates integral of given function on an interval with a 15 point Gauss Kronrod formula and returns error estimate.

**QK21** Evaluates integral of given function on an interval with a 21 point Gauss Kronrod formula and returns error estimate.

**QK31** Evaluates integral of given function on an interval with a 31 point Gauss Kronrod formula and returns error estimate.

**QK41** Evaluates integral of given function on an interval with a 41 point Gauss Kronrod formula and returns error estimate.

**QK51** Evaluates integral of given function on an interval with a 51 point Gauss Kronrod formula and returns error estimate.

**QK61** Evaluates integral of given function on an interval with a 61 point Gauss Kronrod formula and returns error estimate.

*NAG Subprogram Library*

**D01BAF** Computes an estimate of the definite integral of a function of known analytical form, using a Gaussian quadrature formula with a specified number of abscissae. Formulae are provided for a finite interval (Gauss-Legendre), a semi-infinite interval (Gauss-Laguerre, Gauss-Rational), and an infinite interval (Gauss-Hermite).

*NMS Subprogram Library*

**QK15** Evaluates integral of given function on an interval with a 15 point Gauss Kronrod formula and returns error estimate.

**H2a1b2 : Nonautomatic 1-D finite interval quadrature, integrand available only on a grid**

*NAG Subprogram Library*

**D01GAF** Integrates a function which is specified numerically at four or more points, over the whole of its specified range, using third-order finite-difference formulae with error estimates, according to a method due to Gill and Miller.

*NMS Subprogram Library*

**PCHQA** Integrates piecewise cubic from A to B given N-arrays X,F,D. Usually used in conjunction with PCHEZ to form cubic, but can be used independently, especially if the abscissas are equally spaced.

*PORT Subprogram Library*

**CSPQU** Finds the integral of a function defined by pairs (x,y) of input points. The x's can be unequally spaced. Uses spline interpolation.

*Scientific Desk PC Subprogram Library*

**H2A1T** Computes the integral of the array f between x(i) and x(j), given n points in the plane (x(k),f(k)), k=1,. . .,n.

**H2a2a1 : Automatic 1-D finite interval quadrature (user need only specify required accuracy) (special integrand including weight functions, oscillating and singular integrands, principal value integrals, splines, etc.), integrand available via user-defined procedure**

*CMLIB Library (BSPLINE Sublibrary)*

**BFQAD** Integrates function times derivative of B-spline from X1 to X2. The B-spline is in B representation.

**PFQAD** Computes integral on (X1,X2) of product of function and the ID-th derivative of B-spline which is in piecewise polynomial representation.

*CMLIB Library (QUADPKS Sublibrary)*

• **QAGP** Automatic adaptive integrator, allows user to specify location of singularities or difficulties of integrand, uses extrapolation.

• **easy-to-use version**

**QAGPE** Automatic adaptive integrator for function with user specified endpoint singularities, provides more information that QAGP.

• **QAWC** Cauchy principal value integrator, using adaptive Clenshaw Curtis method (real Hilbert transform).

**QAWCE** Cauchy principal value integrator, provides more information than QAWC (real Hilbert transform).

• **QAWO** Automatic adaptive integrator for integrands with oscillatory sine or cosine factor.

**QAWOE** Automatic integrator for integrands with explicit oscillatory sine or cosine factor, provides more information than QAWO.

• **QAWS** Automatic integrator for functions with explicit algebraic and/or logarithmic endpoint singularities.

**QAWSE** Automatic integrator for integrands with explicit algebraic and/or logarithmic endpoint singularities, provides more information than QAWS.

**QMOMO** Computes integral of k-th degree Chebyshev polynomial times one of a selection of functions with various singularities.

*IMSL MATH/LIBRARY Subprogram Library*

**QDAGP** Integrate a function with singularity points given.

**QDAWC** Integrate a function F(x)/(x-c) in the Cauchy principal value sense.

**QDAWO** Integrate a function containing a sine or a cosine.

**QDAWS** Integrate a function with algebraic-logarithmic singularities.

*NAG Subprogram Library*

**D01AKF** Is an adaptive integrator, especially suited to oscillating, non-singular integrands, which calculates an approximation to the integral of a function F(x) over a finite interval (A,B).

**D01ALF** Is a general purpose integrator which calculates an approximation to the integral of a function F(x) over a finite interval (A,B), where the integrand may have local singular behaviour at a finite number of points within the integration interval.

**D01ANF** Calculates an approximation to the cosine or the sine transform of a function G over (A,B), i.e. the integral of G(x)cos($\omega$x) or G(x)sin($\omega$x) over (A,B) (for a user-specified value of $\omega$).

**D01APF** Is an adaptive integrator which calculates an approximation to the integral of a function G(x)W(x) over (A,B) where the weight function W has end-point singularities of algebraico-logarithmic type (see input parameter KEY).

**D01AQF** Calculates an approximation to the Hilbert transform of a function G(x) over (A,B), i.e., the integral of G(x)/(x-c) over (A,B), for user-specified values of A,B,C.

*PORT Subprogram Library*

**BQUAD** Adaptively integrates functions which have discontinuities in their derivatives. User can specify these points.

H2a2a2 : **Nonautomatic 1-D finite interval quadrature (special integrand including weight functions, oscillating and singular integrands, principal value integrals, splines, etc.), integrand available via user-defined procedure**

*CMLIB Library (QUADPKS Sublibrary)*

**QC25C** Uses 25 point Clenshaw-Curtis formula to estimate integral of F(x)W(x) where W(x)=1/(x-c).

**QC25F** Clenshaw-Curtis integration rule for function with cos or sin factor, also uses Gauss Kronrod formula.

**QC25S** Estimates integral of function with algebraic-logarithmic singularities using 25 point Clenshaw-Curtis formula and gives error estimate.

**QK15W** Evaluates integral of given function times arbitrary weight function on interval with 15 point Gauss Kronrod formula and gives error estimate.

**H2a2b1 : Automatic 1-D finite interval quadrature (user need only specify required accuracy) (special integrand including weight functions, oscillating and singular integrands, principal value integrals, splines, etc.), integrand available only on a grid**

*CMLIB Library (BSPLINE Sublibrary)*

**BSQAD** Computes the integral of a B-spline from X1 to X2. The B-spline must be in B representation.

**PPQAD** Computes the integral of a B-spline from X1 to X2. The B-spline must be in piecewise polynomial representation.

*IMSL Subprogram Library*

**DCSQDU** Cubic spline quadrature.

*IMSL MATH/LIBRARY Subprogram Library*

**BSITG** Evaluate the integral of a spline, given its B-spline representation.

*NAG Subprogram Library*

**E02AJF** Determines the coefficients in the Chebyshev series representation of the indefinite integral of a polynomial given in Chebyshev series form.

**E02BDF** Computes the definite integral of a cubic spline from its B-spline representation.

*PORT Subprogram Library*

**BSPLI** Obtains the integrals of basis splines, from the left-most mesh point to a specified set of points.

**SPLNI** Integrates a function described previously by an expansion in terms of B-splines. Several integrations can be performed in one call.

*Scientific Desk PC Subprogram Library*

**E3HIN** Evaluates the definite integral of a piecewise cubic Hermite function over an arbitrary interval.

**E3INT** Evaluates the definite integral of a piecewise cubic Hermite function over an interval whose endpoints are data points.

**H2a3a1 : Automatic 1-D semi-infinite interval quadrature (user need only specify required accuracy) (including exp(-x) weight function), integrand available via user-defined procedure**

*Collected Algorithms of the ACM*

**A614** INTHP: a Fortran subroutine for automatic numerical integration in Hp. The functions may have singularities at one or both end-points of an interval. Each of finite, semi-infinite, and infinite intervals are admitted. (See K. Sikorski, F. Stenger, and J. Schwing, ACM TOMS 10 (1984) pp. 152-160.)

**A639** OSCINT: a Fortran subprogram for the automatic integration of some infinitely oscillating tails. That is, the evaluation of the integral from a to infinity of h(x)j(x), where h(x) is ultimately positive, and j(x) is either a circular function (e.g., cosine) or a first-kind Bessel function of fractional order. (See J. Lyness and G. Hines, ACM TOMS 12 (1986) pp. 24-25.)

*CMLIB Library (QUADPKS Sublibrary)*

• **QAGI** Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation.

**QAGIE** Automatic integrator for semi-infinite or infinite intervals and general integrands, provides more information than QAGI.

• **QAWF** Automatic integrator for Fourier integrals on $(a,\infty)$ with factors $\sin(\omega x)$, $\cos(\omega x)$ by integrating between zeros.

**QAWFE** Automatic integrator for Fourier integrals, with $\sin(\omega x)$ factor on $(a,\infty)$, provides more information than QAWF.

*IMSL MATH/LIBRARY Subprogram Library*

**QDAGI** Integrate a function over an infinite or semi-infinite interval.

**QDAWF** Compute a Fourier integral.

*JCAM Software Library*

**DEHINT** Uses double exponential transformation of Mori to compute semi-infinite range integral automatically to user specified accuracy.

*NAG Subprogram Library*

**D01AMF** Calculates an approximation to the integral of a function F(x) over an infinite or semi-infinite interval (A,B).

*NMS Subprogram Library*

**QAGI** Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation.

## H2a3a2 : Nonautomatic 1-D semi-infinite interval quadrature (including exp(-x) weight function), integrand available via user-defined procedure

*CMLIB Library (QUADPKS Sublibrary)*

**QK15I** Evaluates integral of given function on semi-infinite or infinite interval with a transformed 15 point Gauss Kronrod formula and gives error estimate.

*NAG Subprogram Library*

**D01BAF** Computes an estimate of the definite integral of a function of known analytical form, using a Gaussian quadrature formula with a specified number of abscissae. Formulae are provided for a finite interval (Gauss-Legendre), a semi-infinite interval (Gauss-Laguerre, Gauss-rational), and an infinite interval (Gauss-Hermite).

## H2a4a1 : Automatic 1-D infinite interval quadrature (user need only specify required accuracy) (including $\exp(-x^2)$ weight function), integrand available via user-defined procedure

*Collected Algorithms of the ACM*

**A614** INTHP: a Fortran subroutine for automatic numerical integration in Hp. The functions may have singularities at one or both end-points of an interval. Each of finite, semi-infinite, and infinite intervals are admitted. (See K. Sikorski, F. Stenger, and J. Schwing, ACM TOMS 10 (1984) pp. 152-160.)

*CMLIB Library (QUADPKS Sublibrary)*

• **QAGI** Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation.

**QAGIE** Automatic integrator for semi-infinite or infinite intervals and general integrands, provides more information than QAGI.

*NAG Subprogram Library*

**D01AMF** Calculates an approximation to the integral of a function F(x) over an infinite or semi-infinite interval (A,B).

*NMS Subprogram Library*

**QAGI** Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation.

**H2a4a2 : Nonautomatic 1-D infinite interval quadrature (including exp(-$x^2$) weight function), integrand available via user-defined procedure**

*CMLIB Library (QUADPKS Sublibrary)*

**QK15I** Evaluates integral of given function on semi-infinite or infinite interval with a transformed 15 point Gauss Kronrod formula and gives error estimate.

*NAG Subprogram Library*

**D01BAF** Computes an estimate of the definite integral of a function of known analytical form, using a Gaussian quadrature formula with a specified number of abscissae. Formulae are provided for a finite interval (Gauss-Legendre), a semi-infinite interval (Gauss-Laguerre, Gauss-rational), and an infinite interval (Gauss-Hermite).

**H2b1a1 : Automatic n-D quadrature (user need only specify required accuracy) on one or more hyper-rectangular regions, integrand available via user-defined procedure**

*CMLIB Library (ADAPT Sublibrary)*

**ADAPT** Computes the definite integral of a user specified function over a hyper-rectangular region in dimension 2 through 20. User specifies tolerance. A restarting feature is useful for continuing a computation without wasting previous function values.

*IMSL Subprogram Library*

**DBLIN** Numerical integration of a function of two variables.

**DMLIN** Numerical integration of a function of several variables over a hyper-rectangle (Gaussian method).

*IMSL MATH/LIBRARY Subprogram Library*

**QAND** Integrate a function on a hyper-rectangle.

• **easy-to-use version**

**TWODQ** Compute a two-dimensional iterated integral using internal calls to a one dimensional automatic integrator.

*NAG Subprogram Library*

**D01DAF** Attempts to evaluate a double integral to a specified absolute accuracy by repeated applications of the method described by Patterson.

**D01EAF** Computes approximations to the integrals of a vector of similar functions, each defined over the same multi-dimensional hyper-rectangular region. The routine uses an adaptive subdivision strategy, and also computes absolute error estimates.

**D01FCF** Attempts to evaluate a multidimensional integral (up to 15 dimensions), with constant and finite limits, to a specified relative accuracy, using an adaptive subdivision strategy.

**D01GBF** Returns an approximation to the integral of a function over a hyper-rectangular region, using a Monte-Carlo method. An approximate relative error estimate is also returned. This routine is suitable for low accuracy work.

### H2b1a2 : Nonautomatic n-D quadrature on one or more hyper-rectangular regions, integrand available via user-defined procedure

*NAG Subprogram Library*

**D01FBF** Computes an estimate of a multidimensional integral (from 1 to 20 dimensions), given the analytic form of the integrand and suitable Gaussian weights and abscissae.

**D01FDF** Calculates an approximation to a definite integral in up to 30 dimensions, using the method of Sag and Szekeres. The region of integration is an n-sphere, or by built-in transformation via the unit n-cube, any product region.

**D01GCF** Calculates an approximation to a definite integral in up to 20 dimensions, using the Korobov-Conroy number theoretic method.

### H2b1b2 : Nonautomatic n-D quadrature on one or more hyper-rectangular regions, integrand available only on a grid

*IMSL Subprogram Library*

**DBCQDU** Bicubic spline quadrature.

*IMSL MATH/LIBRARY Subprogram Library*

**BS2IG** Evaluate the integral of a tensor-product spline on a rectangular domain, given its tensor-product B-spline representation.

**BS3IG** Evaluate the integral of a tensor-product spline in three dimensions over a three-dimensional rectangle, given its tensor-product B-spline representation.

### H2b2a1 : Automatic n-D quadrature on a nonrectangular region (user need only specify required accuracy), integrand available via user-defined procedure

*Collected Algorithms of the ACM*

**A584** CUBTRI: a Fortran subroutine for adaptive cubature over a triangle. (See D.P. Laurie, ACM TOMS 8 (1982) pp. 210-218.)

**A612** TRIEX: a Fortran subroutine for integration over a triangle. Uses an adaptive subdivisional strategy with global acceptance criteria and incorporates the epsilon algorithm to speed convergence. (see E. de Doncker and I. Robinson, ACM TOMS 10 (1984) pp. 17-22.)

*CMLIB Library (TWODQ Sublibrary)*

**TWODQ** Automatic (adaptive) integration of a user specified function f(x,y) on one or more triangles to a prescribed relative or absolute accuracy. Two different quadrature formulas are available within TWODQ. This enables a user to integrate functions with boundary singularities.

*NAG Subprogram Library*

**D01JAF** Attempts to evaluate an integral over an n-dimensional sphere (n=2, 3, or 4), to a user specified absolute or relative accuracy, by means of a modified Sag-Szekeres method. The routine can handle singularities on the surface or at the centre of the sphere, and returns an error estimate.

*Scientific Desk PC Subprogram Library*

**H2B2A** Computes the two-dimensional integral of a function f over a region consisting of n triangles.

**H2b2a2 : Nonautomatic n-D quadrature on a nonrectangular region, integrand available via user-defined procedure**

*JCAM Software Library*

**DTRIA** Computes an approximation to the double integral of f(u,v) over a triangle in the uv-plane by using an $n^2$ point, generalized Gauss-Legendre product rule of polynomial degree precision 2n-2. From: "Computation of Double Integrals over a Triangle", by F.G. Lether, Algorithm 007, J. Comp. Appl. Math. 2(1976), 219-224.

*NAG Subprogram Library*

**D01PAF** Returns a sequence of approximations to the integral of a function over a multi-dimensional simplex, together with an error estimate for the last approximation.

**H2c : Service routines for quadrature (compute weight and nodes for quadrature formulas)**

*Collected Algorithms of the ACM*

**A647** Fortran subprograms for the generation of sequences of quasirandom vectors with low discrepancy. Such sequences may be used to reduce error bounds for multidimensional integration and global optimization. (See B.L. Fox, ACM TOMS 12 (1986) pp. 362-376.)

**A655** IQPACK: Fortran routines for the stable evaluation of the weights and nodes of interpolatory and Gaussian quadratures with prescribed simple or multiple knots. (See S. Elhay and J. Kautsky, ACM TOMS 13 (1987) pp. 399-415.)

**A659** A Fortran implementation of Sobol's quasirandom sequence generator for multivariate quadrature and optimization. (See P. Bratley and B. L. Fox, ACM TOMS 14 (1988) pp. 88-100.)

*IMSL MATH/LIBRARY Subprogram Library*

**FQRUL** Compute a Fejer quadrature rule with various classical weight functions.

**GQRCF** Compute a Gauss, Gauss-Radau or Gauss-Lobatto quadrature rule given the recurrence coefficients for the monic polynomials orthogonal with respect to the weight function.

**GQRUL** Compute a Gauss, Gauss-Radau or Gauss-Lobatto quadrature rule with various classical weight functions.

**RECCF** Compute recurrence coefficients for various monic polynomials.

**RECQR** Compute recurrence coefficients for monic polynomials given a quadrature rule.

*NAG Subprogram Library*

**D01BBF** Returns the weights and abscissae appropriate to a Gaussian quadrature formula with a specified number of abscissae. The formulae provided are Gauss-Legendre, Gauss-rational, Gauss-Laguerre and Gauss-Hermite.

**D01BCF** Returns the weights (normal or adjusted) and abscissae for a Gaussian integration rule with a specified number of abscissae. Six different types of Gauss rule are allowed.

*PORT Subprogram Library*

**GAUSQ** Finds the abscissae and weights for Gauss quadrature on the interval (a,b) for a general weight function with known moments.

**GQ0IN** Finds the abscissae and weights for Gauss Laguerre quadrature on the interval $(0,+\infty)$.

**GQM11** Finds the abscissae and weights for Gauss Legendre quadrature on the interval (-1,1).

# I: Differential and Integral Equations

Differential and integral equations are the basis of most mathematical models of continuous processes, and hence the solution of these equations is a very important problem found in many applications. This class is divided into three parts which reflect the most general types of such equations — ordinary differential equations (ODEs), partial differential equations (PDEs), and integral equations.

## I1: Ordinary differential equations

Physical laws are often posed in the form of systems of ordinary differential equations. Most programs for solving ODE's operate only on systems of first order equations of the form

$$y' = f(t, y)$$

or

$$Ay' = f(t, y)$$

where $y$ and $f$ are vectors and $A$ is a square matrix. Most higher order systems can be reduced to one of these forms by a change of variable.

An initial value problem is a system of ODE's and a vector of numbers which specify the solution $y$ at a specific (initial) time. The problem is to determine the solution at subsequent times. An initial value problem is *stiff* if the physical system contains time constants varying over several decades and the solution is desired in a range where the fastest components have died out. A boundary value problem is similar except that the data are not all given at a single point, e.g., some components may be specified at $t_1$ and others at $t_2$ etc.

Programs for solving initial or boundary value problems usually approximate the solution at a discrete set of points which are chosen dynamically by the integrator. These points may not correspond to the user's output points, rather they are selected to get to the end as efficiently as possible. This requires an interpolation procedure (invisible to the user). The solution methods are often implicit and hence ultimately require the solution of systems of linear algebraic equations. Boundary value problems are more difficult and the programs are less reliable. Current techniques include shooting (solving a sequence of initial value problems), collocation (forcing an approximate solution to satisfy the ODE's at selected points), integral equations, invariant imbedding and finite differences.

Most programs require an error tolerance to be specified by the user. If the program terminates normally the implication is that the solution is given to within that tolerance. In practice this is often true. However, the input tolerance is usually used only to control local errors. There is no attempt to control accumulation of error over a long sequence of integration steps.

In Fortran subprograms that allow the user to specify the name of a subroutine to evaluate the differential equations be sure that this name appears

1. in an EXTERNAL statement in the main program,
2. in the CALL statement which invokes the ODE solver, and
3. as the name of a SUBROUTINE.

Another frequent programming error is to confuse the array Y which is input to the main ODE program with the array of the same name in the user-supplied subroutine to evaluate the derivatives.

### References

[CSD79] B. Child, M. Scott, J. W. Daniel, et al., editors. *Codes for Boundary-Value Problems in Ordinary Differential Equations.* Springer-Verlag, New York, 1979.

[Gla79] I. Gladwell. Initial value routines in the NAG library. *ACM Transactions on Mathematical Software*, 5(4):386–400, 1979.

[Kel76] H. B. Keller. *Numerical Solution of Two Point Boundary Value Problems.* SIAM, Philadelphia, 1976.

[Pre75] P. M. Prenter. *Splines and Variational Methods*, chapter 6. John Wiley & Sons, New York, 1975.

[SG75] L. F. Shampine and M. K. Gordon. *Computer Solution of Ordinary Differential Equations: The Initial Value Problem.* W. H. Freeman and Co., San Francisco, 1975.

[SG79] L. F. Shampine and C. W. Gear. A user's view of solving stiff ordinary differential equations. *SIAM Review*, 21(1):1–17, 1979.

[SWD76] L. F. Shampine, H. A. Watts, and S. M. Davenport. Solving nonstiff ordinary differential equations — the state of the art. *SIAM Review*, 18(3):376–411, 1976.

## I1a : Initial value problems for ordinary differential equations

*PLOD Interactive Program*

**PLOD** An easy to use interactive system on a personal computer for the solution of initial value problems for ordinary differential equations. The user can change initial conditions, interval, parameters etc., and examine various plots on the terminal. Little programming needed.

## I1a1a : One-step methods (e.g., Runge-Kutta) for general, nonstiff or mildly stiff initial value problems for ordinary differential equations

*Collected Algorithms of the ACM*

**A497** DMRODE: a subprogram for the automatic integration of functional differential equations, such as retarded ordinary differential equations, Volterra integro-differential equations, and difference differential equations. (See K.W. Neves, ACM TOMS 1 (1975) pp. 369-371.)

**A504** GERK: a Fortran subprogram to solve nonlinear systems of ordinary differential equations when it is important to have a global error estimate. Integrations are performed on different mesh spacings and global extrapolation is applied to provide an estimate of the global error in the more accurate solution. The integrations are done using Runge-Kutta-Fehlberg methods of 4th and 5th order. (See L.F. Shampine and H.A. Watts, ACM TOMS 2 (1976) pp. 200-203.)

**A553** M3RK: a Fortran subroutine for solving initial value problems for nonlinear first-order systems of ordinary differential equations which originate from semi-discretization of parabolic partial differential equations. M3RK is based on stabilized, explicit three-step Runge-Kutta formulas of order one and two, and degree 2 through 12. (See J.G. Verwer, ACM TOMS 6 (1980) pp. 236-239.)

*CMLIB Library (DEPAC Sublibrary)*

**DERKF** Solves a system of first order ordinary differential equations with arbitrary initial conditions by a Runge-Kutta method.

*IMSL Subprogram Library*

**DVERK** Differential equation solver – Runge-Kutta-Verner fifth and sixth order method.

*IMSL MATH/LIBRARY Subprogram Library*

**IVPRK** Solve an initial-value problem for ordinary differential equations using the Runge-Kutta-Verner fifth- and sixth-order method.

*NAG Subprogram Library*

• **D02BAF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method.

• **D02BBF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, and returns the solution at points specified by the user.

**D02BDF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, and computes a global error estimate check. A stiffness check is also available.

**D02BGF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, until a specified component attains a given value.

**D02BHF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, until a user-specified function of the solution is zero.

**D02PAF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method. A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended only to experienced users.

**D02YAF** Integrates a system of first-order ordinary differential equations over one step, using Merson's Runge-Kutta method.

*Scientific Desk PC Subprogram Library*

**I1A1A** Integrates a system of neqn first order ordinary differential equations of the form dy(i)/dt = f(t,y(1),y(2),...,y(neqn)), where the y(i) are given at t (Runge-Kutta-Fehlberg method).

*SCRUNCH Subprogram Library*

**RKF45** Runge-Kutta-Felberg method for the integration of a first order system of ordinary differential equations. In BASIC.

## I1a1b : Multistep methods (e.g., Adams predictor-corrector) for general, nonstiff or mildly stiff initial value problems for ordinary differential equations

*Collected Algorithms of the ACM*

**A658** ODESSA: A Fortran ordinary differential equation solver (a modification of LSODE) with explicit simultaneous sensitivity analysis. (See J. R. Leis and M. A. Kramer, ACM TOMS 14 (1988) pp. 61-67.)

*CMLIB Library (CDRIV Sublibrary)*

• **CDRIV1** Numerical integration of complex initial value problems for ordinary differential equations, Gear stiff formulas. Easy to use.

**CDRIV2** Numerical integration of complex initial value problems for ordinary differential equations, Gear stiff and Adams formulas, root finding.

**CDRIV3** Numerical integration of complex initial value problems for ODEs, Gear and Adams formulas, implicit equations, sparse Jacobians, root finding.

• **easy-to-use version**

*CMLIB Library (DEPAC Sublibrary)*

**DEABM** Solves a system of first order ordinary differential equations with arbitrary initial conditions by a predictor-corrector method.

*CMLIB Library (SDASSL Sublibrary)*

**SDASSL** Solves the system of differential/algebraic equations of the form g(t,y,y′)=0, with given initial values.

*CMLIB Library (SDRIV Sublibrary)*

• **SDRIV1** Numerical integration, initial value problems, ordinary differential equations, Gear stiff Formulas. Easy to Use.

**SDRIV2** Numerical integration, initial value problems, ordinary differential equations, Gear/Adams Formulas.

**SDRIV3** Numerical integration, initial value problems, ordinary differential equations, implicit Equations, sparse Jacobians.

*IMSL Subprogram Library*

**DGEAR** Differential equation solver – variable order Adams predictor-corrector method or Gear's method.

*IMSL MATH/LIBRARY Subprogram Library*

**IVPAG** Solve an initial-value problem for ordinary differential equations using an Adams-Moulton or Gear method.

*NAG Subprogram Library*

• **D02CAF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method.

• **D02CBF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method, and returns the solution at points specified by the user.

**D02CGF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method, until a specified component attains a given value.

**D02CHF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method, until a user-specified function of the solution is zero.

**D02QAF** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method. A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended only to experienced users.

*NMS Subprogram Library*

**SDRIV2** Numerical Integration, Initial Value Problems, Ordinary Differential Equations, Gear /Adams Formulas.

*Scientific Desk PC Subprogram Library*

**I1A2** Numerical integration of initial value problems for ordinary differential equations, implicit equations, sparse Jacobians, root finding.

• **I1A2E** Numerical integration of initial value problems for ordinary differential equations, Gear stiff formulas; easy to use.

**I1A2F** Numerical integration of initial value problems for ordinary differential equations, Gear/Adams formulas, root finding.

| | |
|---|---|
| **I1a1c :** | **Extrapolation methods (e.g., Bulirsch-Stoer) for general, nonstiff or mildly stiff initial value problems for ordinary differential equations** |

*IMSL Subprogram Library*

**DREBS** Solve an initial-value problem for ordinary differential equations using the Bulirsch-Stoer extrapolation method.

*IMSL MATH/LIBRARY Subprogram Library*

**IVPBS** Solve an initial-value problem for ordinary differential equations using the Bulirsch-Stoer extrapolation method.

*PORT Subprogram Library*

• **ODES** Solves an initial value problem for a system of ordinary differential equations. Easy to use.

**ODES1** Solves an initial value problem for a system of ordinary differential equations. Allows great flexibility and user control.

| | |
|---|---|
| **I1a2 :** | **Stiff and mixed algebraic-ordinary differential equations** |

*Collected Algorithms of the ACM*

**A534** STINT: A Fortran subprogram for integrating a set of first order ordinary differential equations using stiffly stable, cyclic composite linear multistep methods. (See J.M. Tendler, T.A. Bickart, and Z. Picel, ACM TOMS 4 (1978) pp. 399-403.)

**A658** ODESSA: A Fortran ordinary differential equation solver (a modification of LSODE) with explicit simultaneous sensitivity analysis. (See J. R. Leis and M. A. Kramer, ACM TOMS 14 (1988) pp. 61-67.)

*CMLIB Library (CDRIV Sublibrary)*

• **CDRIV1** Numerical integration of complex initial value problems for ordinary differential equations, Gear stiff formulas, Easy to use.

**CDRIV2** Numerical integration of complex initial value problems for ordinary differential equations, Gear stiff and Adams formulas, root finding.

**CDRIV3** Numerical integration of complex initial value problems for ODEs, Gear and Adams formulas, Implicit equations, Sparse Jacobians, root finding.

*CMLIB Library (DEPAC Sublibrary)*

**DEBDF** Solves a system of first order stiff ordinary differential equations with arbitrary initial conditions by Gear's method.

*CMLIB Library (SDRIV Sublibrary)*

• **SDRIV1** Numerical Integration, Initial Value Problems, Ordinary Differential Equations, Gear Stiff Formulas, Easy to Use.

**SDRIV2** Numerical Integration, Initial Value Problems, Ordinary Differential Equations, Gear /Adams Formulas.

**SDRIV3** Numerical Integration, Initial Value Problems, Ordinary Differential Equations, Implicit Equations, Sparse Jacobians.

*IMSL Subprogram Library*

**DGEAR** Differential equation solver – variable order Adams predictor-corrector method or Gear's method.

• **easy-to-use version**

*NAG Subprogram Library*

- **D02EAF** Integrates a stiff system of first-order differential equations over a range with suitable initial conditions, using a variable-order variable-step method implementing the Backward Differentiation Formulae.
- **D02EBF** Integrates a stiff system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order, variable-step method implementing the Backward Differentiation Formulae, and returns the solution at points specified by the user.
- **D02EGF** Integrates a stiff system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step method implementing the Backward Differentiation Formulae, until a specified component attains a given value.
- **D02EHF** Integrates a stiff-system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order, variable-step method implementing the Backward Differentiation Formulae, until a user specified function of the solution is zero.

**D02EJF** Integrates a stiff system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order, variable-step method implementing the Backward Differentiation Formulae, until a user-specified function, if supplied, of the solution is zero, and returns the solution at points specified by the user, if desired.

**D02NBF** Forward communication routine for integrating stiff systems of explicit ordinary differential equations when the Jacobian is a full matrix.

**D02NCF** Forward communication routine for integrating stiff systems of explicit ordinary differential equations when the Jacobian is a banded matrix.

**D02NDF** Forward communication routine for integrating stiff systems of explicit ordinary differential equations when the Jacobian is a sparse matrix.

**D02NGF** Forward communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations when the Jacobian is a full matrix.

**D02NHF** Forward communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations when the Jacobian is a banded matrix.

**D02NJF** Forward communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations when the Jacobian is a sparse matrix.

**D02NMF** Reverse communication routine for integrating stiff systems of explicit ordinary differential equations.

**D02NNF** Reverse communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations.

**D02QBF** Integrates a stiff system of first-order ordinary differential equations, over a range with suitable initial conditions, using a variable-order variable-step Gear method. A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended only to experienced users.

**D02QDF** Integrates a stiff system of first order ordinary differential equations, over a range with suitable initial conditions, using a variable-order, variable-step method based on the Backward Differentiation Formulae (BDF). A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended to experienced users only.

*NMS Subprogram Library*

**SDRIV2** Numerical integration, initial value problems, ordinary differential equations, Gear/Adams formulas.

*Scientific Desk PC Subprogram Library*

**I1A2** Numerical integration of initial value problems for ordinary differential equations, implicit equations, sparse Jacobians, root finding.

• **I1A2E** Numerical integration of initial value problems for ordinary differential equations, Gear stiff formulas; easy to use.

**I1A2F** Numerical integration of initial value problems for ordinary differential equations, Gear/Adams formulas, root finding.

## I1b1 : Linear multipoint boundary value problems for ordinary differential equations

*CMLIB Library (BVSUP Sublibrary)*

**BVSUP** Solves boundary value problems for a linear system of ODEs using superposition, orthogonalization, and variable step integration.

*NAG Subprogram Library*

**D02GBF** Solves a general linear two-point boundary value problem for a system of ordinary differential equations using a deferred correction technique.

**D02JAF** Solves a regular linear two-point boundary value problem for a single N(th) order ordinary differential equation by a Chebyshev series using collocation and least squares.

**D02JBF** Solves a regular linear two-point boundary value problem for a system of ordinary differential equations by Chebyshev series using collocation and least squares.

**D02TGF** Solves a system of linear ordinary differential equations by least-squares fitting of a series of Chebyshev polynomials using collocation.

## I1b2 : Nonlinear multipoint boundary value problems for ordinary differential equations

*Collected Algorithms of the ACM*

**A569** COLSYS: Fortran subroutine for solving nonlinear multi-point boundary value problems for mixed order systems of ordinary differential equations. Based upon spline collocation at Gaussian points using a B-spline basis. Approximate solutions are computed on a sequence of automatically selected meshes until a user-specified set of tolerances is satisfied. (See U. Ascher, J. Christiansen, and R.D. Russell, ACM TOMS 7 (1981) pp. 223-229.)

*IMSL Subprogram Library*

**DTPTB** Solve a system of ordinary differential equations with boundary conditions at two points, using a multiple shooting method.

**DVCPR** Solve a system of ordinary differential equations with boundary conditions at two points, using a variable order, variable step size finite difference method with deferred corrections.

*IMSL MATH/LIBRARY Subprogram Library*

**BVPFD** Solve a system of differential equations with boundary conditions at two points, using a variable order, variable step-size finite-difference method with deferred corrections.

**BVPMS** Solve a system of differential equations with boundary conditions at two points, using a multiple shooting method.

*NAG Subprogram Library*

**D02AGF** Solves the two-point boundary-value problem for a system of ordinary differential equations, using initial value techniques and Newton iteration; it generalizes subroutine D02HAF to include the case where parameters other than boundary values are to be determined.

• **D02GAF** Solves the two-point boundary-value problem with assigned boundary values for a system of ordinary differential equations, using a deferred correction technique and a Newton iteration.

• **easy-to-use version**

- • **D02HAF** Solves the two-point boundary-value problem for a system of ordinary differential equations.
- **D02HBF** Solves the two-point boundary-value problem for a system of ordinary differential equations, using initial value techniques (D02PAF) and Newton iteration; it generalizes subroutine D02HAF to include the case where parameters other than boundary values are to be determined.
- **D02RAF** Solves the two-point boundary-value problem with general boundary conditions for a system of ordinary differential equations, using a deferred correction technique and Newton iteration.
- **D02SAF** Solves a two-point boundary-value problem for a system of first order ordinary differential equations with boundary conditions, combined with additional algebraic equations. It uses initial value techniques and a modified Newton iteration in a shooting and matching method.

## I1b3 : Eigenvalue (e.g., Sturm-Liouville) multipoint boundary value problems for ordinary differential equations

*Collected Algorithms of the ACM*

- **A537** CHARMA: a Fortran subprogram for calculating the characteristic values of Mathieu's differential equation for odd or even solutions. (See W.R. Leeb, ACM TOMS 5 (1979) pp. 112-117.)

*NAG Subprogram Library*

- **D02AGF** Solves the two-point boundary-value problem for a system of ordinary differential equations, using initial value techniques and Newton iteration; it generalizes subroutine D02HAF to include the case where parameters other than boundary values are to be determined.
- **D02HBF** Solves the two-point boundary-value problem for a system of ordinary differential equations, using initial value techniques (D02PAF) and Newton iteration; it generalizes subroutine D02HAF to include the case where parameters other than boundary values are to be determined.
- • **D02KAF** Finds a specified eigenvalue of a regular second-order Sturm-Liouville system defined on a finite range, using a Pruefer transformation and a shooting method.
- **D02KDF** Finds a specified eigenvalue of a regular or singular second-order Sturm-Liouville system on a finite or infinite range, using a Pruefer transformation and a shooting method. Provision is made for discontinuities in the coefficient functions or their derivatives.
- **D02KEF** Finds a specified eigenvalue of a regular singular second-order Sturm-Liouville system on a finite or infinite range, using a Pruefer transformation and a shooting method. It also reports values of the eigenfunction and its derivatives. Provision is made for discontinuities in the coefficient functions or their derivatives.

## I1c : Service routines for ordinary differential equations (e.g., interpolation of solutions, error handling, test programs)

*Collected Algorithms of the ACM*

- **A546** SOLVEBLOK: a Fortran subprogram for solving almost block diagonal linear systems. Such matrices arise naturally in piecewise polynomial interpolation or approximation and in finite element methods for two-point boundary value problems. (See C. de Boor and R. Weiss, ACM TOMS 6 (1980) pp. 88-91.)
- **A603** COLROW and ARCECO: Fortran subroutines for solving certain almost block diagonal linear systems by modified alternate row and column elimination. Such systems arise when solving boundary-value problems for ordinary differential equations. COLROW is designed for systems whose blocks all have the same dimension; ARCECO is designed for systems whose blocks may have different dimensions. (See J.C. Diaz, G. Fairweather, and P. Keast, ACM TOMS 9 (1983) pp. 376-380.)

| | |
|---|---|
| **A648** | NSDTST and STDTST: Fortran routines for assessing the performance of initial value solvers for stiff or nonstiff systems. (See W. H. Enright and J. D. Pryce, ACM TOMS 13 (1987) pp. 28-34.) |

*NAG Subprogram Library*

| | |
|---|---|
| **D02NRF** | Enquiry routine for communicating with D02NMF or D02NNF when supplying columns of a sparse Jacobian matrix. |
| **D02NSF** | Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if full matrix linear algebra is required. |
| **D02NTF** | Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if banded matrix linear algebra is required. |
| **D02NUF** | Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if sparse matrix linear algebra is required. |
| **D02NVF** | Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if Backward Differentiation Formulae (BDF) are to be used. |
| **D02NWF** | Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if the BLEND formulae are to be used. |
| **D02NXF** | Optional output routine which the user may call, on exit from an integrator in the D02N subchapter, if sparse matrix linear algebra has been selected. |
| **D02NYF** | Diagnostic routine which the user may call either after any user specified exit or after a mid-integration error exit from any of the integrators in the D02N subchapter. |
| **D02NZF** | Setup routine which must be called, if optional inputs need resetting, prior to a continuation call to any of the integrators in the D02N subchapter. |
| **D02QQF** | Sets up interrupts for use in D02QDF. |
| **D02XAF** | Interpolates the system of first-order ordinary differential equations from information provided by the Runge-Kutta-Merson routine D02PAF. |
| **D02XBF** | Interpolates one component of the solution of a system of first-order ordinary differential equations from information provided by the Runge-Kutta-Merson routine D02PAF. |
| **D02XGF** | Interpolates the solution of a system of first-order ordinary differential equations from information provided by the Adams routine D02QAF or the Gear routine D02QBF. |
| **D02XHF** | Interpolates one component of the solution of a system of first-order ordinary differential equations from information provided by the Adams routine D02QAF or the Gear routine D02QBF. |
| **D02XJF** | Interpolates components of the solution of a system of first-order ordinary differential equations from information provided by the integrators in the D02N subchapter (or by the routine D02QDF). |
| **D02XKF** | Interpolates components of the solution of a system of first-order ordinary differential equations from information provided by the integrators in the D02N subchapter (or by the routine D02QDF). It provides $C^1$ interpolation suitable for general use. |
| **D02ZAF** | Calculates the weighted norm of the local error estimate from inside a MONITR routine called from an integrator in the D02N subchapter. |

*PORT Subprogram Library*

| | |
|---|---|
| **ODESE** | Standard error subprogram for the routine ODES1. |
| **ODESH** | Default HANDLE routine for ODES. Used to access the results at the end of each integration time step. |

## I2: Partial differential equations

Partial differential equations are an important tool for those who model continuous processes in all areas of science. Such complex models are rarely solvable using analytic techniques; instead, they provide some of the most challenging problems in all of scientific computing. Successful numerical methods must forge an effective synthesis of techniques from such diverse areas as approximation theory, numerical quadrature, and the numerical solution of linear and nonlinear algebraic equations.

The most common problems are *second order*, i.e., the highest derivative is a second partial derivative, and may be classified as either elliptic, parabolic or hyperbolic, although the most complex systems are combinations of these. A solution is required on a one, two or three dimensional domain, which may be bounded (with irregular boundaries adding geometrical headaches) or not.

*Elliptic* equations model steady-state phenomena, with the solution determined by conditions specified on the boundaries of the domain (Laplace's equation, $u_{xx} + u_{yy} = 0$, is the prototype). *Parabolic* problems add the element of time, with the solution at future times dependent upon the given solution at some initial time (the heat equation, $u_t = u_{xx}$, is the prototype). *Hyperbolic* problems are also of the initial-boundary value problem type, but are characterized by the finite propagation speed of data (the wave equation, $u_{tt} = u_{xx}$, is the prototype).

Because of the great diversity of problem characteristics, most programs for solving partial differential equations have been aimed at a specific problem in a specific applications area, with little possibility of easy extension to other problems. Thus, the state of general purpose software for partial differential equations is still in its infancy. The exceptions are in the areas of separable elliptic equations on rectangular domains and parabolic system solvers in one space dimension.

Most software begins with some finite dimensional approximation of the spatial part of the differential equation and boundary conditions. Two basic techniques are used — finite differences and finite elements. In *finite differences*, derivatives are directly approximated by difference quotients, leading to a system of algebraic equations whose solution yields values of the unknown quantities at a finite set of points. In *finite elements*, the solution is represented as a finite sum of known functions, each of which is zero on most of the domain. Variational techniques are then used to obtain a system of algebraic equations which determine the unknown coefficients of this sum. Finite elements have been tremendously successful in such application areas as structural engineering where their ability to easily conform to complex geometries is essential. In other areas such as fluid dynamics, finite differences remain popular due to their inherent simplicity. The computational complexity of the problem increases nonlinearly with dimension, and storage and computation times may be prohibitive for all but the coarsest of approximations in three dimensions.

Software which provides efficient and reliable solutions to separable elliptic problems is available. These are single linear elliptic equations which are defined as the sum of two one-dimensional equations, one depending only upon $x$ and one depending only upon $y$. The domain must be rectangular (in Cartesian, polar, or surface spherical coordinates) with simple boundary conditions. In this case very fast techniques related to the numerical separation of variables may be applied. Laplace's equation is easily reformulated so that integral equation techniques may be applied. This is especially attractive for problems on complicated domains. Several programs based on such techniques are available.

Most successful general purpose software packages for parabolic equations are based on the *method of lines.* Here, the spatial part of the differential equations is first approximated by either finite differences or finite elements; what remains is an initial value problem for a system of ordinary differential equations which is solved by existing general purpose software for that problem.

## References

[Ame77] Ames. W. F. *Numerical Methods for Partial Differential Equations.* Academic Press, New York, 2nd edition, 1977.

[BS83] R. F. Boisvert and R. A. Sweet. Mathematical software for elliptic boundary value problems. In W. Cowell, editor, *Sources and Development of Mathematical Software.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[CP76] G. G. Carrier and C. E. Pearson. *Partial Differential Equations.* Academic Press, New York, 1976.

[GW79] I. Gladwell and R. Wait, editors. *A Survey of Numerical Methods for Partial Differential Equations.* Oxford University Press, London, 1979.

[MG80] A. R. Mitchell and D. F. Griffiths. *The Finite Difference Method in Partial Differential Equations.* John Wiley & Sons, New York, 1980.

[MS80] M. Machura and R. A. Sweet. A survey of software for partial differential equations. *ACM Transactions on Mathematical Software*, 6(4):461–488, 1980.

[MW77] A. R. Mitchell and R. Wait. *The Finite Element Method in Partial Differential Equations.* John Wiley & Sons, New York, 1977.

[Roa76] P. J. Roache. *Computational Fluid Dynamics.* Hermosa Publishers, Albuquerque, New Mexico, revised edition, 1976.

## I2a1 : Parabolic partial differential equations

*Collected Algorithms of the ACM*

**A553** M3RK: a Fortran subroutine for solving initial value problems for nonlinear first-order systems of ordinary differential equations which originate from semi-discretization of parabolic partial differential equations. M3RK is based on stabilized, explicit three-step Runge-Kutta formulas of order one and two, and degree 2 through 12. (See J.G. Verwer, ACM TOMS 6 (1980) pp. 236-239.)

## I2a1a : Parabolic partial differential equations in one spatial dimension

*Collected Algorithms of the ACM*

**A494** PDEONE: solution of systems of nonlinear parabolic partial differential equations in one space dimension using the method of lines. (See R.F. Sincovec and N.K. Madsen, ACM TOMS 1 (1975) pp. 261-263.)

**A540** PDECOL: a Fortran subprogram for solving coupled systems of nonlinear partial differential equations in one space and one time dimension. The solution method uses finite element collocation based upon piecewise polynomials for spatial discretization. The time discretization is performed by general-purpose software for ordinary initial value problems. (See N.K. Madsen and R.F. Sincovec, ACM TOMS 5 (1979) pp. 326-351.)

*IMSL Subprogram Library*

**DPDES** Solve a system of partial differential equations of the form $U_t = F(x, t, U, U_x, U_{xx})$ using the method of lines with cubic Hermite polynomials.

*IMSL MATH/LIBRARY Subprogram Library*

**MOLCH** Solve a system of partial differential equations of the form $U_t = F(x, t, U, U_x, U_{xx})$ using the method of lines with cubic Hermite polynomials.

*NAG Subprogram Library*

• **D03PAF** Integrates a single linear or nonlinear parabolic partial differential equation in one space variable, using the method of lines and Gear's method.

• **D03PBF** Integrates a system of linear or nonlinear parabolic partial differential equations in one space variable, using the method of lines and Gear's method.

• **easy-to-use version**

**D03PGF** Integrates a system of nonlinear parabolic partial differential equations in one space variable, using the method of lines and Gear's method. This routine provides quite general facilities; for simpler versions see D03PAF (for a single equation), or D03PBF (for simple systems).

*PDELIB Library (MOL1D Sublibrary)*

**MOL1D** Solves systems of linear or nonlinear initial-boundary-value problems in one space dimension. Can solve hyperbolic equations with or without discontinuities, parabolic equations (including reaction-diffusion equations). Uses the method of lines based on equi-spaced finite differences. Graphical output available.

*PDELIB Library (PDECOL Sublibrary)*

**PDECOL** Solves general nonlinear systems of initial-boundary-value problems in one space dimension with general boundary conditions. Spatial derivatives may be of at most second order. Uses method of lines based on collocation of B-spline basis functions.

## I2a1b : Parabolic partial differential equations in two or more spatial dimensions

*Collected Algorithms of the ACM*

**A565** PDETWO/PSETM/GEARB: a Fortran package for solving time-dependent coupled systems of nonlinear partial differential equations which are defined over a two-dimensional rectangular region. (See D.K. Melgaard and R.F. Sincovec, ACM TOMS 7 (1981) pp. 126-135.)

**A621** BDMG: a Fortran subprogram with low storage requirements for two-dimensional nonlinear parabolic differential equations on rectangular spatial domains with mixed linear boundary conditions. (See B.P. Sommeijer and P.J. van der Houven, ACM TOMS 10 (1984) pp. 378-396.)

*PDELIB Library (PDETWO Sublibrary)*

**PDETWO** Solves general nonlinear systems of initial-boundary-value problems in two spatial dimensions with quasi-linear boundary conditions. Uses the method of lines based upon finite differences on a user-specified rectangular mesh.

## I2a2 : Hyperbolic partial differential equations

*Collected Algorithms of the ACM*

**A540** PDECOL: a Fortran subprogram for solving coupled systems of nonlinear partial differential equations in one space and one time dimension. The solution method uses finite element collocation based upon piecewise polynomials for spatial discretization. The time discretization is performed by general-purpose software for ordinary initial value problems. (See N.K. Madsen and R.F. Sincovec, ACM TOMS 5 (1979) pp. 326-351.)

**A565** PDETWO/PSETM/GEARB: a Fortran package for solving time-dependent coupled systems nonlinear partial differential equations which are defined over a two-dimensional rectangular region. (See D.K. Melgaard and R.F. Sincovec, ACM TOMS 7 (1981) pp. 126-135.)

*IMSL Subprogram Library*

**DPDES** Solve a system of partial differential equations of the form $U_t = F(x, t, U, U_x, U_{xx})$ using the method of lines with cubic Hermite polynomials.

*PDELIB Library (MOL1D Sublibrary)*

**MOL1D** Solves systems of linear or nonlinear initial-boundary-value problems in one space dimension. Can solve hyperbolic equations with or without discontinuities, parabolic equations (including reaction-diffusion equations). Uses the method of lines based on equi-spaced finite differences. Graphical output available.

| I2b1a : | Second order linear elliptic boundary value problems |
|---|---|

*ELLPACK Program Library*

**ELLPACK** Solves linear elliptic partial differential equations in general domains in two dimensions and in boxes; a variety of boundary conditions are handled. Users write programs in the ELLPACK language (Fortran extension) which allows them to declare elliptic problems and to select from a large library of modules to solve them numerically. Results can be tabulated or plotted; the solution is also available as a Fortran function for postprocessing.

| I2b1a1a : | Poisson (Laplace) or Helmholtz equation on a rectangular domain (or topologically rectangular in the coordinate system) |
|---|---|

*Collected Algorithms of the ACM*

**A527** GMA, GMAS, and KPICK: Fortran subroutines for linear systems arising from 5-point discretizations of separable or constant coefficient elliptic boundary-value problems on rectangular domains. A Dirichlet, Neumann, or mixed boundary condition may be independently specified on each side of the rectangle, or periodic boundary conditions may be specified on opposing sides. Implements the generalized marching algorithm. (See R.E. Bank, ACM TOMS 4 (1978) pp. 165-176.)

**A541** FISHPAK: Fortran subroutines for solving separable elliptic partial differential equations. Drivers are available for the Helmholtz equation in Cartesian, polar, surface spherical coordinates, cylindrical and interior spherical coordinates. In addition, subprograms for solving systems of linear equations resulting from finite difference approximations to general separable problems are included. (See P.N. Swarztrauber and R.A. Sweet, ACM TOMS 5 (1979) pp. 352-364.)

**A543** FFT9: a Fortran subroutine for the Dirichlet problem for the Helmholtz equation on a rectangle. The program is based upon 4th and 6th order accurate 9-point finite difference approximations and fast Fourier solution techniques. (See E.N. Houstis and T.S. Papatheodorou, ACM TOMS 5 (1979) pp. 490-493.)

**A651** HFFT: Fortran routines for solving the Helmholtz equation on bounded two- or three-dimensional rectangular domains. (See R. F. Boisvert, ACM TOMS 13 (1987) pp. 235-249.)

*CMLIB Library (FISHPAK Sublibrary)*

**HSTCRT** Solves the Helmholtz or Poisson equations in two dimensions in Cartesian coordinates on a staggered grid.

**HSTCSP** Solves a modified Helmholtz equation in spherical coordinates with axisymmetry using a staggered grid.

**HSTCYL** Solves a modified Helmholtz equation in cylindrical coordinates on a staggered grid.

**HSTPLR** Solves the Helmholtz or Poisson equation in polar coordinates on a staggered grid.

**HSTSSP** Solves the Helmholtz or Poisson equation in spherical coordinates on the surface of a sphere using a staggered grid.

**HW3CRT** Solves the Helmholtz or Poisson equation in three dimensions using Cartesian coordinates.

**HWSCRT** Solves the Helmholtz or Poisson equation in two dimensions in Cartesian coordinates.

**HWSCSP** Solves a modified Helmholtz equation in spherical coordinates with axisymmetry.

**HWSCYL** Solves a modified Helmholtz equation in cylindrical coordinates.

**HWSPLR** Solves the Helmholtz or Poisson equation in polar coordinates.

**HWSSSP** Solves the Helmholtz or Poisson equation in spherical coordinates on the surface of a sphere.

*CMLIB Library (VHS3 Sublibrary)*

**HS3CRT** Sets up and solves the standard seven-point finite difference approximation on a staggered grid to the Helmholtz equation in Cartesian coordinates with a variety of possible boundary conditions.

*IMSL MATH/LIBRARY Subprogram Library*

**FPS2H** Solve Poisson's or Helmholtz's equation on a two-dimensional rectangle using a fast Poisson solver based on the HODIE finite-difference scheme.

**FPS3H** Solve Poisson's or Helmholtz's equation on a three-dimensional box using a fast Poisson solver based on the HODIE finite-difference scheme.

## I2b1a1b : Poisson (Laplace) or Helmholtz equation on a nonrectangular domain

*Collected Algorithms of the ACM*

**A572** HELM3D: a Fortran subroutine for solving the Dirichlet problem for the Helmholtz equation on general bounded three-dimensional regions. Based upon second-order accurate finite differences; the resulting linear system of equations is reduced to a capacitance matrix equation that is solved approximately by a conjugate gradient method. (See D.P. O'Leary and O. Widlund, ACM TOMS 7 (1981) pp. 239-246.)

**A593** CMMEXP, CMMIMP, and CMMSIX: Fortran subroutines for solving the Helmholtz equation on bounded nonrectangular planar regions with Dirichlet or Neumann boundary conditions. Solution is based upon the Fourier method extended to nonrectangular regions using the capacitance matrix method. (See W. Proskurowski, ACM TOMS 9 (1983) pp. 117-124.)

**A629** LAPLAC: a Fortran subroutine for the interior Dirichlet problem for Laplace's equation on a general three dimensional domain. Based on integral equation techniques. (See K.E. Atkinson, ACM TOMS 11 (1985) pp. 85-96.)

*NAG Subprogram Library*

**D03EAF** Solves Laplaces's equation in two dimensions for an arbitrary domain bounded internally or externally by one or more closed contours, given the value of either the unknown function or its normal derivative (into the domain) at each point of the boundary.

## I2b1a2 : Other separable second order linear elliptic boundary value problems

*Collected Algorithms of the ACM*

**A527** GMA, GMAS, and KPICK: Fortran subroutines for linear systems arising from 5-point discretizations of separable or constant coefficient elliptic boundary-value problems on rectangular domains. A Dirichlet, Neumann, or mixed boundary condition may be independently specified on each side of the rectangle, or periodic boundary conditions may be specified on opposing sides. Implements the generalized marching algorithm. (See R.E. Bank, ACM TOMS 4 (1978) pp. 165-176.)

*CMLIB Library (FISHPAK Sublibrary)*

**SEPELI** Solves separable elliptic boundary value problems on a rectangle.

**SEPX4** Solves separable elliptic boundary value problems on a rectangle with constant coefficients in one direction.

| **I2b1a3 :** | **Nonseparable second order linear elliptic boundary value problems** |
|---|---|

*Collected Algorithms of the ACM*

**A637** GENCOL: a Fortran subprogram for linear second-order elliptic problems with general linear boundary conditions on non-rectangular two-dimensional domains. Solves the problem using collocation with bicubic Hermite polynomials. (See E.N. Houstis, W.F. Mitchell, and J.R. Rice, ACM TOMS 11 (1985) pp. 379-412 and 413-415.)

**A638** INTCOL and HERMCOL: Fortran subprograms for linear second-order elliptic problems on rectangular two-dimensional domains. HERMCOL allows general linear boundary conditions while INTCOL required uncoupled boundary conditions. Problem are solved using collocation with bicubic Hermite polynomials. (See E.N. Houstis, W.F. Mitchell, and J.R. Rice, ACM TOMS 11 (1985) pp. 379-412 and 416-418.)

| **I2b4 :** | **Service routines for elliptic boundary value problems** |
|---|---|

*Collected Algorithms of the ACM*

**A499** CONOPT: a subprogram which determines the contour scanning path for a two-dimensional region. The path is designed to help accelerate the propagation of edge effects when solving two-dimensional partial differential equations using iterative methods. (See W. Kinsner and E.D. Torre, ACM TOMS 2 (1976) pp. 82-86.)

**A625** A Fortran subprogram which relates a general two-dimensional domain to a rectangular grid laid over it. (See J.R. Rice, ACM TOMS 10 (1984) pp. 443-452 and 453-462.)

| **I2b4a :** | **Domain triangulation for elliptic boundary value problems** *(search also class P)* |
|---|---|

*NAG Subprogram Library*

**D03MAF** Places a triangular mesh over a given 2-dimensional region. The region may have any shape, including one with holes.

| **I2b4b :** | **Solution of discretized elliptic equations** |
|---|---|

*Collected Algorithms of the ACM*

**A512** FACTOR, RHS, and SOLVE: Fortran subroutines for solving symmetric positive definite periodic quindiagonal systems of linear equations. (See A. Benson, and D.J. Evans, ACM TOMS 3 (1977) pp. 96-103.)

**A527** GMA, GMAS, and KPICK: Fortran subroutines for linear systems arising from 5-point discretizations of separable or constant coefficient elliptic boundary-value problems on rectangular domains. A Dirichlet, Neumann, or mixed boundary condition may be independently specified on each side of the rectangle, or periodic boundary conditions may be specified on opposing sides. Implements the generalized marching algorithm. (See R.E. Bank, ACM TOMS 4 (1978) pp. 165-176.)

**A541** FISHPAK: Fortran subroutines for solving separable elliptic partial differential equations. Drivers are available for the Helmholtz equation in Cartesian, polar, surface spherical coordinates, cylindrical and interior spherical coordinates. In addition, subprograms for solving systems of linear equations resulting from finite difference approximations to general separable problems are included. (See P.N. Swarztrauber and R.A. Sweet, ACM TOMS 5 (1979) pp. 352-364.)

*CMLIB Library (FISHPAK Sublibrary)*

**BLKTRI** Solves block tridiagonal systems of linear algebraic equations arising from the discretization of separable elliptic partial differential equations.

**CBLKTR** Solves certain complex block tridiagonal systems of linear equations arising from the discretization of separable elliptic partial differential equations.

**CMGNBN** Solves certain complex block tridiagonal systems of linear equations arising from Helmholtz or Poisson equations in 2 dimensional Cartesian coordinates.

**GENBUN** Solves certain block tridiagonal systems of linear equations arising from Helmholtz or Poisson equations in two Cartesian coordinates.

**POIS3D** Solves block tridiagonal linear systems of algebraic equations arising from the discretization of separable elliptic partial differential equations in 3D.

**POISTG** Solves block tridiagonal linear systems of algebraic equations arising from the discretization of separable elliptic partial differential equations.

*CMLIB Library (VHS3 Sublibrary)*

**PSTG3D** Solves certain block tridiagonal systems of linear algebraic equations that arise in finite difference approximations to three-dimensional Helmholtz equations on a staggered grid.

*NAG Subprogram Library*

• **D03EBF** Uses the Strongly Implicit Procedure to calculate the solution to a system of simultaneous algebraic equations of five point molecule form on a two-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example (r,theta), can be used, being equivalent to a rectangular box.)

• **D03ECF** Uses the Strongly Implicit Procedure to calculate the solution to a system of simultaneous algebraic equations of seven point molecule form on a three-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example, can be used if it is equivalent to a rectangular box.)

**D03EDF** Solves 7-diagonal systems of linear equations which arise from the discretisation of an elliptic partial differential equation on a rectangular region. This routine uses a multigrid technique.

**D03UAF** Performs at each call one iteration of the Strongly Implicit Procedure. It is used to calculate on successive calls a sequence of approximate corrections to the current estimate of the solution when solving a system of simultaneous algebraic equations for which the iterative up-date matrix is of five point molecule form on a two-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example (r,theta), can be used, being equivalent to a rectangular box.)

**D03UBF** Performs at each call one iteration of the Strongly Implicit Procedure. It is used to calculate on successive calls the sequence of approximate corrections to the solution when solving a system of simultaneous algebraic equations for which the iterative up-date matrix is of seven-point molecule form on a three-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example, can be used if it is equivalent to a rectangular box.)

## I3: Integral Equations

An integral equation is a functional equation in which the unknown function occurs as an essential part of an integrand. Integral equations are termed *first kind* or *second kind* depending on whether the unknown function appears only inside or both inside and outside the integral. *Fredholm* equations are defined on a fixed interval, while *Volterra* equations are defined on a semi-infinite region. Some examples of integral equations follow; $u$ is the unknown function here.

$$\int_0^1 K(x,y)u(y)dy = g(x) \qquad 0 \le x \le 1 \tag{1}$$

$$u(x) - \int_0^1 K(x,y)u(y)dy = g(x) \qquad 0 \le x \le 1 \tag{2}$$

$$\int_0^x K(x,y)u(y)dy = g(x) \qquad x \ge 0 \tag{3}$$

$$u(x) - \int_0^x K(x,y)u(y)dy = g(x) \qquad x \ge 0 \tag{4}$$

Equations (1) and (3) are first kind; (2) and (4) are second kind. Equations (1) and (2) are Fredholm; (3) and (4) are Volterra. The function $K(x,y)$ is called the *kernel.*

Second kind equations are much easier to solve, both computationally and mathematically. First kind equations are often very sensitive numerically and special care must be exercised or the results will be meaningless. Easier problems correspond to those with more singular kernels; very smooth kernels are particularly difficult to handle.

## References

[Bak77] C. T. H. Baker. *The Numerical Treatment of Integral Equations.* Oxford University Press, New York, 1977.

[DM85] L. M. Delves and J. L. Mohamed. *Computational Methods for Integral Equations.* Cambridge University Press, Cambridge, 1985.

[Mus58] N. I. Muskhelishvili. *Singular Integral Equations: Boundary Problems of Functions Theory and their Applications to Mathematical Physics.* Wolters-Noordhoff Publishing, Groningen, The Netherlands, revised edition, 1958.

[Nob77] B. Noble. The numerical solution of integral equations. In D. Jacobs, editor, *The State of the Art in Numerical Analysis*, pages 915–966. Academic Press, New York, 1977.

**I3 : Integral equations**

*Collected Algorithms of the ACM*

**A503** IESIMP and IEGAUS: Fortran subprograms for solving one-dimensional linear Fredholm integral equations of the second kind. The routines are based on the Nystrom method using Simpson's and Gauss quadrature formulas, respectively. The resulting linear systems are solved using an iterative method. (See K. Atkinson, ACM TOMS 2 (1976) pp. 196-199.)

**A627** VE1: a Fortran subroutine for solving Volterra integral equations. (See J.M. Bownds and L. Applebaum, ACM TOMS 11 (1985) pp. 58-65.)

*NAG Subprogram Library*

**D05AAF** Solves a linear, non-singular Fredholm equation of the second kind with a split kernel.

**D05ABF** Solves any linear non-singular Fredholm integral equation of the second kind with a smooth kernel.

# J: Integral Transforms

Programs in this class compute integral transforms, the most common of which is the Fourier transform. FFT programs allow transformation to/from spectral space. The programs are most efficient when $n$ is highly composite, a power of two being the most propitious choice. Although most programs will work correctly for any $n$, it may take 200–300% more computing for an $n = 127$ transform than an $n = 128$. On the other hand, adding zero value data points can introduce spurious effects in the spectrum. The program selected must not only satisfy the physical model requirements (pure cosine transform, complex transform, etc.) but should also be appropriate for the intended computer. FFT programs can be made extra efficient by taking advantage of special machine hardware (e.g. vectorization capability) or by writing segments in assembly language. Even in Fortran, implementations on the same machine can easily differ 30% in running time.

## References

[Bri74] E. O. Brigham. *The Fast Fourier Transform.* Prentice-Hall, Englewood Cliffs, NJ, 1974.

[DSPC79] Digital Signal Processing Committee. *Programs for Digital Signal Processing.* IEEE Press, New York, 1979.

| J : | **Integral transforms** |
|---|---|

*Collected Algorithms of the ACM*

A588 HANKEL: a Fortran subroutine for the fast evaluation of complex Hankel transforms of orders 0 and 1 using related and lagged convolutions. (See W.L. Anderson, ACM TOMS 8 (1982) pp. 369-370.)

| J1a1 : | **One-dimensional real fast Fourier transforms** |
|---|---|

*Collected Algorithms of the ACM*

A545 CMFFT and RMFFT: Fortran subprograms for computing real and complex fast Fourier transforms of a one-dimensional or multidimensional data set. The program is designed to minimize I/O for the case where the data lie on mass storage. (See D. Fraser, ACM TOMS 5 (1979) pp. 500-517.)

*CMLIB Library (FFTPACK Sublibrary)*

• EZFFTB Backward real discrete (fast) Fourier transform. Performs Fourier synthesis. Easy to use version.

• EZFFTF Forward real discrete (fast) Fourier transform. Performs Fourier analysis. Easy to use version.

RFFTB Computes real periodic sequence from real Fourier coefficients. Performs Fourier synthesis.

RFFTF Computes Fourier coefficients of real periodic sequence (fast). Performs Fourier analysis.

RFFTI Initialize WSAVE array for SUBROUTINE RFFTF and RFFTB.

*CMLIB Library (VFFT Sublibrary)*

VRFFTB Synthesis (backward transform) of multiple real periodic sequences from their Fourier coefficients.

VRFFTF Analysis (forward transform) of multiple real periodic sequences into Fourier coefficients.

*CMLIB Library (VSFFT Sublibrary)*

**VSRFTB** Synthesis (backward transform) of multiple Fourier coefficients into real staggered grid sequences.

**VSRFTF** Analysis (forward transform) of multiple real staggered grid sequences into Fourier coefficients.

*IMSL Subprogram Library*

**FFTRC** Fast Fourier transform of a real valued sequence.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**FFTRB** Compute the real periodic sequence from its Fourier coefficients.

**FFTRF** Compute the Fourier coefficients of a real periodic sequence.

*MAGEV Cyber 205 Subprogram Library*

**FFT1D** One-dimensional fast Fourier transform of one or several real or complex vectors whose lengths can be expressed as $n=2^p3^q5^r$, where p, q, and r are non-negative integers and $p \geq 2$. The user can choose among several options.

*NAG Subprogram Library*

**C06EAF** Calculates the discrete Fourier transform of a sequence of N real data values. (No extra workspace required.)

**C06FAF** Calculates the discrete Fourier transform of a sequence of N real data values (using a work array for extra speed).

**C06FPF** Computes the discrete Fourier transforms of M sequences, each containing N real data values. This routine is designed to be particularly efficient on vector processors.

**C06FQF** Computes the discrete Fourier transforms of M Hermitian sequences, each containing N complex data values. This routine is designed to be particularly efficient on vector processors.

*NMS Subprogram Library*

**EZFFTB** Backward FFT of N data points in AZERO,A(N/2),B(N/2). Returns data in R(N).

**EZFFTF** Forward FFT of N data points R(N). Returns A(N/2) and B(N/2) as cosine and sine coefficients, AZERO as mean.

*PORT Subprogram Library*

**FFTR** Mixed radix fast Fourier transform to find the transform of 2N real data points.

**FFTRI** Finds the inverse Fourier transform using Fourier coefficients assumed to arise from real data in the time domain.

**RLTR** An auxiliary routine for use together with FFT to transform 2N real data points. Uses less storage than FFTR.

*Scientific Desk PC Subprogram Library*

**J1A3B** Computes a real periodic sequence from its Fourier coefficients. Inverse FFT.

**J1A3F** Fast Fourier transform (FFT) of a real periodic sequence.

• **J1AEB** Computes a real periodic sequence from its Fourier coefficients. Inverse Fourier transform. This routine is a simplified version of J1A3B. It is not as fast as J1A3B since scaling and initialization are computed for each transform.

• **J1AET** Computes the Fourier coefficients of a real periodic sequence. Fast Fourier transform. This routine is a simplified version of J1A3F. It is not as fast as J1A3F since scaling and initialization are computed for each transform.

• **easy-to-use version**

*STARPAC Subprogram Library*

**FFTLEN** Compute the minimum extended series length for using the Singleton FFT; return the extended series length (no printed output).

**FFTR** Compute the Fourier coefficients of an input series of real observations; return the Fourier coefficients (no printed output).

## J1a2 : One-dimensional complex fast Fourier transforms

*Collected Algorithms of the ACM*

**A545** CMFFT and RMFFT: a Fortran subprograms for computing real and complex fast Fourier transforms of a one-dimensional or multidimensional data. The program is designed to minimize I/O for the case where the data lies on mass storage. (See D. Fraser, ACM TOMS 5 (1979) pp. 500-517.)

*CMLIB Library (FFTPACK Sublibrary)*

**CFFTB** Backward complex discrete (fast) Fourier transform. Performs Fourier synthesis.

**CFFTF** Forward complex discrete (fast) Fourier transform. Performs Fourier analysis.

*IMSL Subprogram Library*

**FFT2C** Computes the fast Fourier transform of a complex valued sequence of length equal to a power of two.

**FFTCC** Compute the fast Fourier transform of a complex valued sequence.

*IMSL MATH/LIBRARY Subprogram Library*

**FFT2D** Compute Fourier coefficients of a complex periodic two-dimensional array.

**FFTCB** Compute the complex periodic sequence from its Fourier coefficients.

**FFTCF** Compute the Fourier coefficients of a complex periodic sequence.

*MAGEV Cyber 205 Subprogram Library*

**FFT1D** One-dimensional fast Fourier transform of one or several real or complex vectors whose lengths can be expressed as $n=2^p 3^q 5^r$, where p, q, and r are non-negative integers and $p \geq 2$. The user can choose among several options.

*NAG Subprogram Library*

**C06EBF** Calculates the discrete Fourier transform of a Hermitian sequence of N complex data values. (No extra workspace required.)

**C06ECF** Calculates the discrete Fourier transform of a sequence of N complex data values. (No extra workspace required.)

**C06FBF** Calculates the discrete Fourier transform of a Hermitian sequence of N complex data values (using a work array for extra speed).

**C06FCF** Calculates the discrete Fourier transform of a sequence of N complex data values (using a work array for extra speed).

**C06FFF** Computes the discrete Fourier transform of one variable in a multivariate sequence of complex data values.

**C06FRF** Computes the discrete Fourier transforms of M sequences, each containing N complex data values. This routine is designed to be particularly efficient on vector processors.

**C06GQF** Forms the complex conjugates of M Hermitian sequences, each containing N data values.

**C06GSF** Takes M Hermitian sequences, each containing N data values, and forms the real and imaginary parts of the M corresponding complex sequences.

*NMS Subprogram Library*

**CFFTB** Backward complex FFT of complex C(N). Returns result in C.

**CFFTF** Forward complex FFT of complex C(N). Returns result in C.

*PORT Subprogram Library*

**FFT** FFT of complex data sequence (forward or inverse) any number of points. Useful for multivariate transforms. Uses only real arithmetic.

**FFTC** Mixed radix fast Fourier transform of complex data. Two arrays used for complex data.

**FFTCI** Inverse fast Fourier transform, given the Fourier coefficients in the frequency domain.

*Scientific Desk PC Subprogram Library*

**J1A2B** Backward complex discrete Fourier transform, FFT.

**J1A2F** Forward complex discrete Fourier transform, FFT.

## J1a3 : One-dimensional trigonometric (sine, cosine) fast Fourier transforms

*Collected Algorithms of the ACM*

**A649** FOURCO: A Fortran package for computing the trigonometric Fourier coefficients of a smooth function using Lyness's algorithm. (See G. Giunta and A. Murli, ACM TOMS 13 (1987) pp. 97-107.)

*CMLIB Library (FFTPACK Sublibrary)*

**COSQB** Fast Fourier transform of quarter wave data. Computes a sequence from cosine series representation. Fourier synthesis.

**COSQF** Fast Fourier transform of quarter wave data. Fourier analysis. Computes coefficients in cosine series with odd wave numbers.

**COST** Discrete fast cosine transform of even sequence X(I).

**SINQB** Fast Fourier transform of quarter wave data. Backward fast sine transform. Performs Fourier synthesis.

**SINQF** Computes fast Fourier transform of quarter wave data. Forward fast sine transform. Performs Fourier analysis.

**SINT** Computes fast Fourier sine transform of an odd sequence X(I).

*CMLIB Library (VFFT Sublibrary)*

**VCOSQB** Backward fast Fourier cosine transform of multiple quarter-wave sequences. That is, cosine series representations with only odd wave numbers.

**VCOSQF** Forward fast Fourier cosine transform of multiple quarter-wave sequences. That is, cosine series representations with only odd wave numbers.

**VCOST** Discrete Fourier cosine transform of multiple even sequences.

**VSINQB** Backward fast Fourier sine transform of multiple quarter-wave sequences. That is, sine series representations with only odd wave numbers.

**VSINQF** Forward fast Fourier sine transform of multiple quarter-wave sequences. That is, sine series representations with only odd wave numbers.

**VSINT** Discrete Fourier sine transform of multiple odd sequences.

*CMLIB Library (VSFFT Sublibrary)*

**VSCOSB** Synthesis (backward transform) of multiple Fourier cosine coefficients into real staggered grid sequences.

**VSCOSF** Analysis (forward transform) of multiple real staggered grid sequences into Fourier cosine coefficients.

**VSCOSQ** Analysis or synthesis (forward or backward transform) of multiple real staggered grid sequences to or from Fourier cosine quarter-wave sequences.

**VSSINB** Synthesis (backward transform) of multiple Fourier sine coefficients into real staggered grid sequences.

**VSSINF** Analysis (forward transform) of multiple real staggered grid sequences into Fourier sine coefficients.

**VSSINQ** Analysis or synthesis (forward or backward transform) of multiple real staggered grid sequences to or from Fourier sine quarter-wave coefficients.

*IMSL Subprogram Library*

**FFTSC** Sine and cosine transforms of a real valued sequence.

*IMSL MATH/LIBRARY Subprogram Library*

**FCOST** Discrete Fourier cosine transformation of an even sequence.

**FSINT** Discrete Fourier cosine transformation of an odd sequence.

**QCOSB** Compute a sequence from its cosine Fourier coefficients with only odd wave numbers.

**QCOSF** Compute the coefficients of the cosine Fourier transform with only odd wave numbers.

**QSINB** Compute a sequence from its sine Fourier coefficients with only odd wave numbers.

**QSINF** Compute the coefficients of the sine Fourier transform with only odd wave numbers.

*Scientific Desk PC Subprogram Library*

**J1A3C** Discrete Fourier cosine transform of an even sequence X.

**J1A3D** Backward fast Fourier transform (FFT) of quarter wave data. That is, computes a sequence from its representation in terms of a sine series with odd wave numbers.

**J1A3E** Fast Fourier transform (FFT) of quarter wave data. That is, computes the coefficients in a sine series representation with only odd wave numbers.

**J1A3G** Fast Fourier transform (FFT) of quarter wave data. That is, computes the coefficients in a cosine series representation with only odd wave numbers.

**J1A3H** Backward fast Fourier transform of quarter wave data. That is, computes a sequence from its representation in terms of a cosine series with odd wave numbers.

**J1A3S** Discrete Fourier sine transform (FFT) of an odd sequence X.

## J1b : Multidimensional fast Fourier transforms

*Collected Algorithms of the ACM*

**A545** CMFFT and RMFFT: Fortran subprograms for computing real and complex fast Fourier transforms of a one-dimensional or multidimensional dataset . The program is designed to minimize I/O for the case where the data lie on mass storage. (See D. Fraser, ACM TOMS 5 (1979) pp. 500-517.)

*IMSL Subprogram Library*

**FFT3D** Fast Fourier transform of a complex valued 1, 2 or 3 dimensional array.

*IMSL MATH/LIBRARY Subprogram Library*

**FFT2B** Inverse Fourier transform of a complex periodic two-dimensional array.

**FFT2D** Compute Fourier coefficients of a complex periodic two-dimensional array.

*NAG Subprogram Library*

**C06FJF** Computes the multi-dimensional discrete Fourier transform of a multivariate sequence of complex data values.

*NMS Subprogram Library*

**CFFT2D** Two dimensional fast Fourier transform (FFT), forward or reverse, of a complex n-by-m matrix F.

*PORT Subprogram Library*

**FFT** FFT of complex data sequence (forward or inverse) any number of points. Useful for multivariate transforms. Uses only real arithmetic.

*Scientific Desk PC Subprogram Library*

**J1B2T** Two dimensional fast Fourier transform (FFT), forward or reverse, of a complex n-b-m matrix F.

## J2 : Convolutions

*Collected Algorithms of the ACM*

**A588** HANKEL: a Fortran subroutine for the fast evaluation of complex Hankel transforms of orders 0 and 1 using related and lagged convolutions. (See W.L. Anderson, ACM TOMS 8 (1982) pp. 369-370.)

*IMSL Subprogram Library*

**VCONVO** Vector convolution.

*NAG Subprogram Library*

**C06EKF** Calculates the circular convolution or correlation of two real vectors of period N. (No extra workspace is required.)

**C06FKF** Calculates the circular convolution or correlation of two real vectors of period N (using a work array for extra speed).

## J3 : Laplace transforms

*Collected Algorithms of the ACM*

**A619** DLAINV: a Fortran subroutine for the automatic numerical inversion of the Laplace transform using the Durbin formula in combination with the epsilon algorithm. (See R. Piessens and R. Huysmans, ACM TOMS 10 (1984) pp. 348-353.)

**A662** MODUL1 and MODUL2: Fortran routines for the numerical inversion of the Laplace transform based on Weeks' method. (See B. S. Garbow, G. Giunta, and J. N. Lyness, ACM TOMS 14 (1988) pp. 171-176.)

*IMSL Subprogram Library*

**FLINV** Inverse Laplace transform of a user supplied complex function.

*IMSL MATH/LIBRARY Subprogram Library*

**INLAP** Inverse Laplace transform of a complex function.

*NAG Subprogram Library*

**C06LAF** Estimates values of the inverse Laplace transform of a given function using a Fourier series approximation. Real and imaginary parts of the function, and a bound on the exponential order of the inverse, are required.

*Scientific Desk PC Subprogram Library*

**J3LD** Computes an approximation to the inverse Laplace transform F(T) for the value of the independent variable equal to T. The Durbin formula and the epsilon algorithm are used. The (hopeful) accuracy claim is $|F(T)-RESULT| \leq \max(EPSABS, EPSREL \times |F(T)|)$.

## J4 : Hilbert transforms

*CMLIB Library (QUADPKS Sublibrary)*

• **QAWC** Cauchy principal value integrator, using adaptive Clenshaw Curtis method (real Hilbert transform).

**QAWCE** Cauchy Principal value integrator, provides more information than QAWC (real Hilbert transform).

**QC25C** Uses 25 point Clenshaw-Curtis formula to estimate integral of F(x)W(x) where W(x)=1/(x-c).

*NAG Subprogram Library*

**D01AQF** Calculates an approximation to the Hilbert transform of a function G(x) over (A,B), i.e., the integral of G(x)/(x-c) over (A,B), for user-specified values of A,B,C.

# K: Approximation

Any computer calculation which tries to model a continuous process involves some sort of approximation. In this class we consider only those programs which have curve or surface fitting as their ultimate goal. Computer-aided design (CAD) is an important application of this software. Here one would like a simple mathematical function which represents the shape of an object under study. Such a function might be used, for example, by a numerically controlled milling machine. Another example is in computer graphics, where one wants a visually pleasing curve to approximate given data. Function approximation is a third example. Here one has a complicated mathematical function which must be replaced by a simpler one providing an approximation with a guaranteed maximum error.

The nature of an approximating curve or surface is determined by two fundamental choices: the choices of norm and form. The *norm* is the means for measuring the distance of an approximating function $g(x)$ from the data $(x_i, y_i)$, $i = 1, \ldots, n$. In most cases we want to determine the free parameters of the approximating function so that this distance is minimized. Some important choices are:

(a) Least squares: minimize $\sum_{i=1}^{n}(y_i - g(x_i))^2$

(b) Least absolute value: minimize $\sum_{i=1}^{n} \mid y_i - g(x_i) \mid$

(c) Minimax: minimize $\max_{1 \leq i \leq n} \mid y_i - g(x_i) \mid$

(d) Orthogonal distance: minimize $\sum_{i=1}^{n} \text{distance}(y_i, g(x))$

Interpolation is a special case of approximation. Here there are at least as many free parameters in $g(x)$ as there are data points, and we require that $y_i - g(x_i) = 0$ for $i = 1, \ldots, n$, that is, $g(x)$ passes through the data points. Interpolation is useful when fitting a curve or surface to data which are known exactly, while least squares is more appropriate for data with inherent error. Interpolation software is found in class E.

The classical *form* for approximating functions is the polynomial. Unfortunately, polynomials are inadequate for many applications: low degree polynomials lack flexibility, while high degree polynomials can fluctuate wildly, and hence lack physical interpretation. An important breakthrough of the 1960's and 70's were piecewise polynomials (splines). These functions consist of low degree polynomial pieces joined smoothly at a set of breakpoints or knots. Piecewise polynomials are both flexible and stable, and have been used with enormous success by engineers and scientists in areas such as the aerospace and automobile industries for years. Easy to use software for computations with piecewise polynomials is now abundant.

Both polynomials and piecewise polynomials are examples of linear approximating functions. That is, they can be written in the form

$$g(x) = a_1 f_1(x) + a_2 f_2(x) + \ldots + a_n f_n(x)$$

where the *basis* functions $f_i$ are fixed and the coefficients $a_i$ are determined from the data. (Spline functions can be written in this form using the so-called B-representation.) Nonlinear approximating functions such as rational functions (the quotient of two polynomials) or sum of exponentials are also often useful, though more difficult to handle computationally.

A wide variety of norms and fitting functions are represented in existing software for approximation. A wealth of software is available for the general least squares approximation problem. Least absolute value and minimax computations are much more difficult and less software is available for these.

Most software for approximation utilizes polynomial or piecewise polynomial fitting functions, although some is also available for trigonometric and rational functions. The ultimate flexibility is offered by routines that work with user-defined fitting functions. In many cases two subprograms are needed to solve an approximation problem: one to determine the coefficients of the approximating function from the data and the second to evaluate the fitted function at one or more points chosen by the user. Occasionally subroutines for evaluating derivatives or definite integrals of the approximation function are also provided.

## References

[BR74] R. E. Barnhill and R. F. Riesenfeld. *Computer Aided Geometric Design.* Academic Press, New York, 1974.

[Che66] E. W. Cheney. *Introduction to Approximation Theory.* McGraw-Hill, New York, 1966.

[Dav75] P. J. Davis. *Interpolation and Approximation.* Dover Publications, New York, 1975.

[deB78] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, New York, 1978.

[Riv69] T. J. Rivlin. *An Introduction to the Approximation of Functions.* Blaisdell, Waltham, Massachusetts, 1969.

[Sch81] L. L. Schumaker. *Spline Functions: Basic Theory.* John Wiley & Sons, New York, 1981.

## K1: Least squares approximation

In least squares curve fitting a set of $m$ data values $(x_i, y_i), i = 1, \ldots, m$ are approximated by a function $g(x)$ which depends on $n$ unknown parameters $a_1, a_2, \ldots, a_n$. The set of parameter values is found which minimizes

$$\sum_{i=1}^{m} w_i(y_i - g(x_i))^2$$

where the $w_i$ are suitably chosen positive weights. In statistics this is known as the regression problem. Software which performs least squares approximation with statistical analysis of the results is found in class L8. This class also overlaps with class G (Optimization) where software for minimizing sums of squares can be found.

There is an important distinction between linear and nonlinear least squares approximation. In the former the unknown parameters appear as linear coefficients of the known fitting functions, whereas in the latter they may be appear in any position. For example, in the fitting function

$$y = ae^{bx} + ce^{dx}$$

$a$ and $c$ are linear parameters while $b$ and $d$ enter nonlinearly. This concept of linearity refers only to the unknown parameters and not to the fitting functions. These functions can be highly nonlinear functions of the $x$ values, as in the example above. It is also possible to impose side conditions or constraints on least squares fits. For example, one might force the fitted function to pass through a data point, or to have non-negative slope at a given point. Linear least squares approximation problems have an equivalent matrix formulation, and hence algorithms for these problems are special cases of those for solving overdetermined systems of linear equations (described in class D9).

In the nonlinear case iterative methods must be employed, and these are special cases of the more general optimization routines described in class G. Typically, such algorithms require derivatives of the function with respect to the parameters evaluated at the data points and the current parameter estimates. Some modules allow the user the option of either providing these derivatives or having them estimated by finite differences.

## References

[deB78] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, New York, 1978.

[DS83] J. E. Dennis and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[LH74] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems.* Prentice-Hall, Englewood Cliffs, NJ, 1974.

**K1a1a1 : Unconstrained linear least squares piecewise polynomial approximation on univariate data (curve fitting)**

*Collected Algorithms of the ACM*

**A525** ADAPT: a Fortran subroutine for approximating a user-defined function by a piecewise polynomial of specified smoothness and degree. The user selects the accuracy required of the approximant as well as the norm by which the error is to be measured. (See J.R. Rice, ACM TOMS 4 (1978) pp. 82-94.)

*CMLIB Library (FC Sublibrary)*

**FC** Fits piecewise polynomial to discrete data with equality and inequality constraints.

*IMSL Subprogram Library*

**ICSFKU** Least squares approximation by cubic splines with fixed knots.

**ICSVKU** Least squares approximation by cubic splines with variable knots.

*IMSL MATH/LIBRARY Subprogram Library*

**BSLSQ** Compute a B-spline least squares spline approximation to given data.

**BSVLS** Compute the variable knot B-spline least squares approximation to given data.

*NAG Subprogram Library*

**E02BAF** Computes a weighted least-squares approximation to an arbitrary set of data points by a cubic spline with knots prescribed by the user. Cubic spline interpolation can also be carried out.

*PORT Subprogram Library*

**DL2SF** Fits discrete data with a B-spline of order K, by least squares.

**DL2SW** Fits discrete data with a B-spline of order k, by weighted least squares.

**L2SFF** Obtains a weighted least square expansion of a known function in terms of B-splines of order K, at given mesh points.

**L2SFH** Obtains a weighted least square expansion of a known function in and its derivatives in terms of B-splines of order K at given mesh points.

**K1a1a2 : Unconstrained linear least squares polynomial approximation on univariate data (curve fitting)**

*IMSL MATH/LIBRARY Subprogram Library*

**RLINE** Fit a line to a set of data points using least squares.

*IMSL STAT/LIBRARY Subprogram Library*

**RPOLY** Analyze a polynomial regression model.

*NAG Subprogram Library*

**E02ADF** Computes weighted least-squares polynomial approximations to an arbitrary set of data points.

**E02AFF** Computes the coefficients of a polynomial, in its Chebyshev series form, which interpolates (passes exactly through) data at a special set of points. Least-squares polynomial approximations can also be obtained.

| K1a1a3 : | **Unconstrained linear least squares approximation on univariate data (curve fitting)** |
|---|---|

*CMLIB Library (ODRPACK Sublibrary)*

• **SODR** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated.

**SODRC** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated.

*IMSL Subprogram Library*

**IFLSQ** Least squares approximation with user supplied functions.

*IMSL MATH/LIBRARY Subprogram Library*

**FNLSQ** Least squares approximation with user-supplied basis functions.

| K1a1b : | **Unconstrained linear least squares approximation on multivariate data (surface fitting)** |
|---|---|

*Collected Algorithms of the ACM*

**A634** CONST and EVAL: Fortran subroutines for fitting multinomials in a least-squares sense. (See R.H. Bartels and J.J. Jezioranski, ACM TOMS 11 (1985) pp. 218-228.)

*IMSL MATH/LIBRARY Subprogram Library*

**BSLS2** Compute a two-dimensional tensor-product spline approximant using least squares, returning the tensor product B-spline coefficients.

*NAG Subprogram Library*

**E02CAF** Forms an approximation to the weighted, least-squares Chebyshev series surface fit to data arbitrarily distributed on lines parallel to one independent co-ordinate axis.

**E02DAF** Forms a minimal, weighted least-squares bicubic spline surface fit with prescribed knots to a given set of data points.

| K1a2a : | **Linearly constrained linear least squares approximation** |
|---|---|

*CMLIB Library (FC Sublibrary)*

**FC** Fits piecewise polynomial to discrete data with equality and inequality constraints.

*IMSL STAT/LIBRARY Subprogram Library*

**RLEQU** Fit a multivariate linear regression model with linear equality restrictions H×BETA = G imposed on the regression parameters given results from IMSL routine RGIVN after IDO = 1 and IDO = 2 and prior to IDO = 3.

*NAG Subprogram Library*

**E02AGF** Computes constrained weighted least-squares polynomial approximations in Chebyshev series form to an arbitrary set of data points. The values of the approximations and any number of their derivatives can be specified at selected points.

**K1b1a1 : Unconstrained nonlinear least squares approximation by smooth functions, user provides no derivatives**

*Collected Algorithms of the ACM*

**A573** NL2SOL: an adaptive nonlinear least-squares algorithm. (See J.E. Dennis, D.M. Gay, and R.E. Welsch, ACM TOMS 7 (1981) pp. 367-383.)

*CMLIB Library (NL2SN Sublibrary)*

**NL2SN** Minimizes a nonlinear sum of squares using residual values only.

*CMLIB Library (ODRPACK Sublibrary)*

• **SODR** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated.

**SODRC** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated.

*CMLIB Library (SNLS1E Sublibrary)*

**SNLS1** Minimizes the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. Flexible usage, including various options for providing Jacobian. Covariance matrix is available via the subroutine SCOV.

• **SNLS1E** Minimizes the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. An easy to use driver for SNLS1. The covariance matrix is available by calling the subroutine SCOV.

*IMSL Subprogram Library*

**ZXSSQ** Minimum of the sum of squares of m functions in n variables using a finite difference Levenberg-Marquardt algorithm.

*IMSL MATH/LIBRARY Subprogram Library*

**BCLSF** Solve a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.

**BCONF** Minimize a function of N variables subject to bounds on the variables using a quasi-Newton method and a finite-difference gradient.

**UNLSF** Solve a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.

*IMSL STAT/LIBRARY Subprogram Library*

**RNLIN** Fit a nonlinear regression model.

*INVAR Interactive Program Library*

**INVAR1** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results. Line printer graphics only.

**INVAR2** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results and DISSPLA graphics.

• **easy-to-use version**

*NAG Subprogram Library*

**E04FCF** Is a comprehensive algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables (M≥N). No derivatives are required.

• **E04FDF** Is an easy-to-use algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables (M≥N). No derivatives are required.

*Scientific Desk PC Subprogram Library*

• **G1B2E** Minimizes the sum of squares of M nonlinear functions in N variables using a modification of the Levenberg-Marquardt algorithm from Minpack. The Jacobian can be approximated by G1B2E or supplied by the user. (Easy to use.) User supplied Jacobian can be checked by G4C, and covariance matrix can be obtained by G4F.

**G1B2U** Minimizes the sum of squares of M nonlinear functions in N variables using a modification of the Levenberg-Marquardt algorithm from Minpack. The Jacobian can be supplied by the user or approximated by G1B2U. User supplied Jacobian can be checked by G4C, covariance matrix can be computed by G4F.

**L8G1A** Determines an $r$ for the equation $Y_i=b_0+b_1X_i^r$ so that $Y_i=b_0+b_1Z_i$, where $Z_i=X_i^r$ plots as a straight line. The iteration scheme used for $r$ is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $r$.

**L8G1E** Determines an $a$ for the equation $Y_i=b_0+b_1a^{X_i}$ so that $Y_i=b_0+b_1Z_i$, where $Z_i = a^{X_i}$ plots as a straight line. The iteration scheme used for $a$ is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $a$.

## K1b1a2 : Unconstrained nonlinear least squares approximation by smooth functions, user provides first derivatives

*CMLIB Library (NL2SN Sublibrary)*

**NL2S1** Minimizes a nonlinear sum of squares using both residual and gradient values supplied by the user.

*CMLIB Library (ODRPACK Sublibrary)*

• **SODR** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated.

**SODRC** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated.

*CMLIB Library (SNLS1E Sublibrary)*

**SNLS1** Minimizes the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. Flexible usage, including various options for providing Jacobian. Covariance matrix is available via the subroutine SCOV.

• **SNLS1E** Minimizes the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. An easy to use driver for SNLS1. The covariance matrix is available by calling the subroutine SCOV.

*IMSL MATH/LIBRARY Subprogram Library*

**BCLSJ** Solve a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

**UNLSJ** Solve a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

*IMSL STAT/LIBRARY Subprogram Library*

**RNLIN** Fit a nonlinear regression model.

*INVAR Interactive Program Library*

**INVAR1** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results. Line printer graphics only.

**INVAR2** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results and DISSPLA graphics.

*NAG Subprogram Library*

**E04GBF** Is a comprehensive quasi-Newton algorithm for finding an unconstrained minimum of a sum of squares of M non-linear functions in N variables (M≥N). First derivatives are required.

• **E04GCF** Is an easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables (M≥N). First derivatives are required.

**E04GDF** Is a comprehensive modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of M non-linear functions in N variables (M≥N). First derivatives are required. The routine is intended for functions which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities).

• **E04GEF** Is an easy-to-use modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables (M≥N). First derivatives are required.

*NASHLIB Subprogram Library*

**A23** Use Marquardt algorithm to find minimum of a sum of squares (nonlinear least squares). Illustrates use of algorithm A23MRT. (See J.C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimization*, John Wiley, New York, 1979, pages 175-177.)

*Scientific Desk PC Subprogram Library*

• **G1B2E** Minimizes the sum of squares of M nonlinear functions in N variables using a modification of the Levenberg-Marquardt algorithm from Minpack. The Jacobian can be approximated by G1B2E or supplied by the user. (Easy to use.) User supplied Jacobian can be checked by G4C, and covariance matrix can be obtained by G4F.

**G1B2U** Minimizes the sum of squares of M nonlinear functions in N variables using a modification of the Levenberg-Marquardt algorithm from Minpack. The Jacobian can be supplied by the user or approximated by G1B2U. User supplied Jacobian can be checked by G4C, covariance matrix can be computed by G4F.

## K1b1a3 : Unconstrained nonlinear least squares approximation by smooth functions, user provides first and second derivatives

*NAG Subprogram Library*

**E04HEF** Is a comprehensive modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of M non-linear functions in N variables (M≥N). First and second derivatives are required.

• **E04HFF** Is an easy-to-use modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables (M≥N). First and second derivatives are required.

## K1b2a : Linearly constrained nonlinear least squares approximation

*IMSL MATH/LIBRARY Subprogram Library*

**BCLSF** Solve a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian.

**BCLSJ** Solve a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian.

**BCONF** Minimize a function of N variables subject to bounds on the variables using a quasi-Newton method and a finite-difference gradient.

## K2 : Minimax ($L_\infty$) approximation

*Collected Algorithms of the ACM*

**A501** APPROX/EXCH: subprograms for computing the best polynomial approximation to a discrete one-dimensional data set in the Chebyshev (minimax) sense. (See J.C. Simpson, ACM TOMS 2 (1976) pp. 95-97.)

**A525** ADAPT: a Fortran subroutine for approximating a user-defined function by a piecewise polynomial of specified smoothness and degree. The user selects the accuracy required of the approximant as well as the norm by which the error is to be measured. (See J.R. Rice, ACM TOMS 4 (1978) pp. 82-94.)

**A604** EXTREM: a Fortran subprogram for the calculating extremal polynomials. If L is a linear functional on polynomials of degree n or less, then p is extremal if it is a polynomial of (Chebyshev) norm one at which L takes on its norm. (See F.W. Sauer, ACM TOMS 9 (1983) pp. 381-383.)

*IMSL Subprogram Library*

**IRATCU** Rational weighted Chebyshev approximation of a continuous function.

**RLLMV** Perform linear regression using the minimax criterion.

*IMSL MATH/LIBRARY Subprogram Library*

**RATCH** Compute a rational weighted Chebyshev approximation to a continuous function on an interval.

*IMSL STAT/LIBRARY Subprogram Library*

**RLMV** Fit a multiple linear regression model using the minimax criterion.

*NAG Subprogram Library*

**E02ACF** Calculates a minimax polynomial fit to a set of data points.

*PORT Subprogram Library*

**BURAM** Finds the best uniform rational approximation to a given function on a specified mesh.

**BURM1** Finds the best uniform rational approximation to a given function on a specified mesh, starting from a given initial approximation.

## K3 : Least absolute value ($L_1$) approximation

*Collected Algorithms of the ACM*

**A525** ADAPT: a Fortran subroutine for approximating a user-defined function by a piecewise polynomial of specified smoothness and degree. The user selects the accuracy required of the approximant as well as the norm by which the error is to be measured. (See J.R. Rice, ACM TOMS 4 (1978) pp. 82-94.)

**A564** L1GNR: a Fortran subroutine for generating test problems for discrete linear L-sub-1 approximation problems. (See K.L. Hoffman and D.R. Shier, ACM TOMS 6 (1980) pp. 615-617.)

*IMSL Subprogram Library*

**RLLAV** Perform linear regression using the least absolute values criterion.

*IMSL STAT/LIBRARY Subprogram Library*

**RLAV** Fit a multiple linear regression model using the least absolute values criterion.

## K4 : Approximation by other analytic functions (e.g., Taylor polynomial, Pade)

*Collected Algorithms of the ACM*

**A510** STL2: a Fortran subprogram which determines a piecewise linear approximation within specified tolerances of given data points which is composed of the fewest line segments. The approximant need not be continuous, and distinct tolerances may be specified for each data point. (See D.G. Wilson, ACM TOMS 2 (1976) pp. 388-391.)

**A525** ADAPT: a Fortran subroutine for approximating a user-defined function by a piecewise polynomial of specified smoothness and degree. The user selects the accuracy required of the approximant as well as the norm by which the error is to be measured. (See J.R. Rice, ACM TOMS 4 (1978) pp. 82-94.)

**A579** CPSC: a Fortran subroutine for evaluating the leading coefficients in a power series expansion of an analytic function. (See B. Fornberg, ACM TOMS 7 (1981) pp. 542-547.)

*NAG Subprogram Library*

**E02RAF** Calculates the coefficients in a Pade approximant to a function from its user-supplied Maclaurin expansion.

## K5 : Smoothing

*Collected Algorithms of the ACM*

**A642** CUBGCV: a Fortran subprogram for the fast O(n) computation of a cubic smoothing spline fitted to n noisy data points, with the degree of smoothing chosen to minimize the expected mean square error at the data points when the variance of the error in the data is known, or the generalized cross validation when it is unknown. The data may be unequally spaced and nonuniformly weighted. Bayesian point error estimates are also calculated. (See M.F. Hutchinson, ACM TOMS 12 (1986) pp. 150-153.)

*IMSL Subprogram Library*

**ICSMOU** One-dimensional data smoothing by error detection.

**ICSSCU** Cubic spline data smoother.

• **ICSSCV** Cubic spline data smoother.

• **easy-to-use version**

*IMSL MATH/LIBRARY Subprogram Library*

**CSSCV** Compute a smooth cubic spline approximation to noisy data using cross-validation to estimate the smoothing parameter.

**CSSED** Smooth one-dimensional data by error detection.

**CSSMH** Compute a smooth cubic spline approximation to noisy data.

## K6a1 : Evaluation of fitted functions

*CMLIB Library (BSPLINE Sublibrary)*

**BSPEV** Calculates the value of a spline and its derivatives at X from its B representation.

**BVALU** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its B representation.

**PPVAL** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its piecewise polynomial representation.

*IMSL Subprogram Library*

**IBCEVL** Evaluation of a bicubic spline.

**ICSEVU** Evaluation of a cubic spline.

*IMSL MATH/LIBRARY Subprogram Library*

**BSVAL** Evaluate a spline, given its B-spline representation.

**CSVAL** Evaluate a cubic spline.

**PPVAL** Evaluate a piecewise polynomial.

*NAG Subprogram Library*

**E02AEF** Evaluates a polynomial from its Chebyshev series representation.

**E02AKF** Evaluates a polynomial from its Chebyshev series representation, allowing an arbitrary index increment for accessing the array of coefficients.

**E02BBF** Evaluates a cubic spline from its B-spline representation.

**E02BCF** Evaluates a cubic spline and its first three derivatives from its B-spline representation.

**E02CBF** Evaluates a bivariate polynomial from the rectangular array of coefficients in its double Chebyshev series representation.

**E02DBF** Calculates values of a bicubic spline from its B-spline representation.

**E02RBF** Evaluates a rational function at a user-supplied point, give the numerator and denominator coefficients.

*PORT Subprogram Library*

**CSPFE** Evaluates a cubic spline function which has already been fit to n input data pairs (x,y) by CSPFI.

**SPLN1** Evaluates a function and derivatives described previously by an expansion in terms of B-splines.

**SPLN2** Evaluates a function described by a previously determined expansion in B-splines. More flexible than SPLN1.

**SPLND** Evaluates at a given set of points a function described by a previously determined expansion in terms of B-splines.

**SPLNE** Evaluates, at a set of points, a function described by a previously determined expansion in terms of B-splines.

| K6a2 : | Evaluation of derivatives of fitted functions |
|---|---|

*CMLIB Library (BSPLINE Sublibrary)*

**BSPEV** Calculates the value of a spline and its derivatives at X from its B representation.

**BVALU** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its B representation.

**PPVAL** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its piecewise polynomial representation.

*IMSL Subprogram Library*

**DBCEVL** Bicubic spline mixed partial derivative evaluator.

**DCSEVU** Cubic spline first and second derivative evaluator.

*IMSL MATH/LIBRARY Subprogram Library*

**BSDER** Evaluate the derivative of a spline, given its B-spline representation.

**CSDER** Evaluate the derivative of a cubic spline.

**PPDER** Evaluate the derivative of a piecewise polynomial.

*NAG Subprogram Library*

**E02AHF** Determines the coefficients in the Chebyshev series representation of the derivative of a polynomial given in Chebyshev series form.

**E02BCF** Evaluates a cubic spline and its first three derivatives from its B-spline representation.

*PORT Subprogram Library*

**SPLN1** Evaluates a function and derivatives described previously by an expansion in terms of B-splines.

**SPLN2** Evaluates a function described by a previously determined expansion in B-splines. More flexible than SPLN1.

**SPLND** Evaluates at a given set of points a function described by a previously determined expansion in terms of B-splines.

| K6a3 : | Quadrature involving fitted functions |
|---|---|

*CMLIB Library (BSPLINE Sublibrary)*

**BFQAD** Integrates function times derivative of B-spline from X1 to X2. The B-spline is in B representation.

**BSQAD** Computes the integral of a B-spline from X1 to X2. The B-spline must be in B representation.

**PFQAD** Computes integral on (X1,X2) of product of function and the ID-th derivative of B-spline which is in piecewise polynomial representation.

**PPQAD** Computes the integral of a B-spline from X1 to X2. The B-spline must be in piecewise polynomial representation.

*IMSL Subprogram Library*

**DCSQDU** Cubic spline quadrature.

*IMSL MATH/LIBRARY Subprogram Library*

**CSITG** Evaluate the integral of a cubic spline.

**PPITG** Evaluate the integral of a piecewise polynomial.

*NAG Subprogram Library*

**E02AJF** Determines the coefficients in the Chebyshev series representation of the indefinite integral of a polynomial given in Chebyshev series form.

**E02BDF** Computes the definite integral of a cubic spline from its B-spline representation.

*NMS Subprogram Library*

**PCHQA** Integrates piecewise cubic from A to B given N-arrays X,F,D. Usually used in conjunction with PCHEZ to form cubic, but can be used independently, especially if the abscissas are equally spaced.

*PORT Subprogram Library*

**BSPLI** Obtains the integrals of basis splines, from the left-most mesh point to a specified set of points.

**SPLNI** Integrates a function described previously by an expansion in terms of B-splines. Several integrations can be performed in one call.

## K6b : Grid or knot generation

*PORT Subprogram Library*

**ILUMB** Given a basic mesh, this subdivides each interval into the same number of uniformly spaced points for B-spline use.

**ILUMD** Given a basic mesh, this subdivides each interval into the same number or uniformly spaced points for B-spline use.

**IMNPB** Creates a B-spline mesh from an array of fitting points, using at least n fitting points in each mesh interval.

**IPUMB** Given a basic mesh, this subdivides each interval. Number of points per interval can vary, but uniform in each subdivision.

**IPUMD** Given a basic mesh, this subdivides each interval with a variable number of points. Points are uniform in each interval.

**IUMB** Given interval endpoints, this generates a uniform mesh for B-spline use.

**IUMD** Given interval endpoints, this generates a uniform mesh.

**LUMB** Given a basic mesh, this subdivides each interval uniformly for B-spline use. Multiplicities are allowed.

**LUMD** Given a basic mesh, this subdivides each interval into the same number of uniformly spaced points.

**MNPB** Creates a B-spline mesh from an array of fitting points, using at least n fitting points in each interval.

**PUMB** Given a basic mesh, this subdivides each interval into a uniform but variable number of points. Multiplicities can occur.

**PUMD** Given a basic mesh, this subdivides each interval into a uniform but variable number of points.

**UMB** Given interval endpoints, this generates a uniform mesh, with needed multiplicities for B-spline use.

**UMD** Given interval endpoints, this generates a uniform mesh of distinct points.

| K6c : | Manipulation of basis functions (e.g., evaluation, change of basis) |
|---|---|

*CMLIB Library (BSPLINE Sublibrary)*

**BSPPP** Converts from B representation of B-spline to piecewise polynomial representation.

**BSPVD** Calculates value and derivatives of order less than NDERIV of all B-spline basis functions which do not vanish at X.

**BSPVN** Calculates the value of all (possibly) nonzero B-spline basis functions at X of a given order.

**INTRV** Computes the index into a knot or breakpoint sequence corresponding to a given point X.

*IMSL MATH/LIBRARY Subprogram Library*

**BSCPP** Convert a spline in B-spline representation to piecewise polynomial representation.

*PORT Subprogram Library*

**BSPL1** Evaluates, at a given set of points in a specified mesh interval, basis splines together with selected orders of derivatives.

**BSPLD** Evaluates at a given set of points in a specified mesh interval, basis splines and their derivatives.

**BSPLN** Evaluates at a given set of points in a specified mesh interval, all the basis splines which are nonzero in that interval.

| K6d : | Other service routines for approximation |
|---|---|

*Collected Algorithms of the ACM*

**A564** L1GNR: a Fortran subroutine for generating test problems for discrete linear L-sub-1 approximation problems. (See K.L. Hoffman and D.R. Shier, ACM TOMS 6 (1980) pp. 615-617.)

*NAG Subprogram Library*

**E02ZAF** Sorts two-dimensional data into rectangular panels.

**E04NDF** Supplies optional parameters to NAG Fortran Library routine E04NCF from an external file.

**E04NEF** Supplies individual optional parameters to NAG Fortran Library routine E04NCF.

**E04YCF** Returns estimates of elements of the variance-covariance matrix of the estimated regression coefficients for a nonlinear least squares problem. The estimates are derived from the Jacobian of the function f(x) at the solution.

*PORT Subprogram Library*

**EEBSF** Estimates the error in a given B-spline fit to a function by refining the mesh.

**EEBSI** Estimates the error in a given B-spline fit to a function by refining the mesh intervals selected by user.

**EESFF** Finds the maximum absolute error in a given B-spline fit to a function.

**EESFI** Finds the maximum absolute error in a given B-spline fit to a function on a set of user selected intervals.

*Scientific Desk PC Subprogram Library*

**G4F** Calculates the covariance matrix for a nonlinear data fitting problem. Intended to be used after a successful return from either G1B2U or G1B2E.

# L: Statistics, Probability

The organization of this class follows that of many academic statistics programs. In the software in the first three subclasses, no statistical assumptions are made about the data being analyzed, but rather elementary summary statistics are calculated (L1), elementary transformations are made (L2), and the data are plotted (L3). Class L4 provides software for analysis of univariate and multivariate statistical data based on elementary statistical assumptions. Classes L5 and L6 provide two basic statistical tools — random number generation and statistical function evaluation. Classes L7 through L16 provide software for more advanced multivariate statistical data analysis. Class L17 contains some statistical data sets which are frequently cited in the literature.

Within each class L subclass which is further partitioned (e.g., L1), the next level of subclasses is based on the dimension of the data set; an exception is class L8 where the distinction is the form of the regression function. There is also parallelism between different class L subclasses (classes L1 and L4 are one example). In addition, some classes are reserved for future software, e.g., software for display of four-dimensional data using color for the fourth dimension will be documented in class L3d.

Two rules were used in classifying software that handles one-, two-, and higher-dimensional data. Since analysis of one-dimensional data is so frequently performed, software which handles both one- and higher-dimensional data is classified in both the one-dimensional class and the higher-dimensional class. Software that handles two- and higher-dimensional data is classified only in the higher-dimensional class, and the class for two-dimensional data refers users to the higher-dimensional class; similarly for three-dimensional data.

Much of the available software can automatically handle data sets in which there are missing values or weighted data. The classification scheme doesn't explicitly identify that software except when the volume warrants.

## References


[And84] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis.* John Wiley & Sons, New York, 2nd edition, 1984.

[BD77] P. J. Bickel and K. A. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics.* Holden-Day Inc., Oakland, CA, 1977.

[BD87] P. Brockwell and R. Davis. *Time Series: Theory and Methods.* Springer-Verlag, New York, 1987.

[BHH78] G. E. P. Box, W. G. Hunter, and J. S. Hunter. *Statistics for Experimenters.* John Wiley & Sons, New York, 1978.

[BL84] V. Barnett and T. Lewis. *Outliers in Statistical Data.* John Wiley & Sons, New York, 2nd edition, 1984.

[Cha84] C. Chatfield. *The Analysis of Time Series.* Chapman and Hall, London, 3rd edition, 1984.

[Fel67] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. John Wiley & Sons, 3rd edition, 1967.

[HC78] R. V. Hogg and A. T. Craig. *Introduction to Mathematical Statistics.* Macmillan, New York, 1978.

[Hun83] J. Hunter. *Mathematical Techniques of Applied Probability.* Academic Press, New York, 1983. (two volumes).

[HW73] M. Hollander and D. A. Wolfe. *Nonparametric Statistical Methods.* Wiley-Interscience, New York, 1973.

[JW88] R. Johnson and D. Wichern. *Applied Multivariate Statistical Analysis.* Prentice-Hall, Englewood Cliffs, NJ, 2nd edition, 1988.

[KJ82] S. Kotz and N. Johnson. *Encyclopedia of Statistical Sciences.* John Wiley & Sons, New York, 1982.

[KT75] S. Karlin and H. Taylor. *A First Course in Stochastic Processes.* Academic Press, New York, 2nd edition, 1975.

[Leh83] E. L. Lehmann. *Theory of Point Estimation.* John Wiley & Sons, New York, 1983.

[Leh86] E. L. Lehmann. *Testing Statistical Hypotheses.* John Wiley & Sons, New York, 2nd edition, 1986.

[MT77] F. Mosteller and J. W. Tukey. *Data Analysis and Regression: A Second Course in Statistics.* Addison-Wesley, Reading, MA, 1977.

[NWK88] J. Neter, W. Wasserman, and M. Kutner. *Applied Linear Regression Models.* Richard D. Irwin, Inc., Homewood, Illinois, 2nd edition, 1988.

[RW79] R. H. Randles and D. A. Wolfe. *Introduction to the Theory of Nonparametric Statistics.* Wiley-Interscience, New York, 1979.

[Tuk77] J. W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, Reading, MA, 1977.

[Woo75] M. Woodroofe. *Probability with Applications.* McGraw-Hill, New York, 1975.

## L1: Data Summarization

Data summarization is often the first step in the statistical analysis of a data set because it provides initial insight through a small number of summary statistics.

The software documented in this class is first organized by the dimension of the data set — one-dimensional (L1a), two-dimensional (L1b), and multi-dimensional (L1c). In both L1b and L1c, the software is for pairwise summarization, e.g., each off-diagonal entry in the sample variance-covariance matrix is the covariance of a pair of variables.

While the software in this class is purely for summarizing data, some summary statistics have broader statistical properties. Indeed, different statistical assumptions are the reason for many of the summary statistics. For example, the sample mean is a good estimator of the mean of normally distributed data but the sample median is preferred for some other distributions. On the other hand, many EDA (exploratory data analysis) statistics are not generally incorporated in more advanced statistical analyses; examples include hinges and other components of letter value summaries. Summary statistics such as those classified here are the fundamental tools used in classes L4 and L7 through L16.

### References

[BD77] P. J. Bickel and K. A. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics.* Holden-Day Inc., Oakland, CA, 1977.

[HC78] R. V. Hogg and A. T. Craig. *Introduction to Mathematical Statistics.* Macmillan, New York, 1978.

[HW73] M. Hollander and D. A. Wolfe. *Nonparametric Statistical Methods.* Wiley-Interscience, New York, 1973.

[MT77] F. Mosteller and J. W. Tukey. *Data Analysis and Regression: A Second Course in Statistics.* Addison-Wesley, Reading, MA, 1977.

[Tuk77] J. W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, Reading, MA, 1977.

## L1a1 : Summary statistics for one-dimensional data

*BMDP Program Library*

**P1D** Prints univariate statistics (mean, standard deviation, standard error of mean, coefficient of variation, extreme values, extreme z-scores, range) for each variable. Options: statistics for each level of each grouping variable, sorting, printing all cases OR only cases with values missing or values outside specified limits.

**P2D** For each variable, prints frequency and percent for each distinct value; mean, median, mode, standard deviation, standard errors of mean and median, skewness, kurtosis, half interquartile range; histogram, and stem-and-leaf plot. Options: initially round or truncate, three robust location estimates.

*DATAPLOT Interactive System*

**SUMMARY** Generate summary statistics for a one-dimensional data set. Generated statistics include location statistics (mean, median, midmean, midrange), dispersion statistics (range, standard deviation, average absolute deviation, minimum and maximum, lower and upper quartiles, lower and upper hinges), randomness statistics (autocorrelation), and distributional statistics (8 types).

*IMSL Subprogram Library*

**BDLTV** Produce letter value summary.

**BEIUGR** Estimation of basic statistical parameters using ungrouped data.

*IMSL STAT/LIBRARY Subprogram Library*

**LETTR** Produce a letter value summary.

*MINITAB Interactive System*

**LVALS** Prints letter-value display – median, hinges, eighths, etc., and optionally saves results.

**MINITAB** Vector summarization commands include COUNT, SUM, MEAN, MAX, MIN, MEDIAN, N, NMISS (number of missing values), STDEV, SSQ (sum of squares), DESCRIBE (N, MEAN, MEDIAN, STDEV, MAX, MIN, 5% trimmed mean, quartiles) for columns or rows (use prefix R, e.g., RMEAN) of data in the Minitab worksheet.

**TABLE** Produces and prints one-way, two-way, and multi-way tables of counts with 20 optional subcommands for summarizing (e.g., cell mean, standard deviation), marginals, performing chi-square tests for each 2-way table, handling missing values, and selecting forms of input and output.

*NAG Subprogram Library*

**G01AAF** Calculates the mean, standard deviation, coefficients of skewness and kurtosis, and the maximum and minimum values for a set of ungrouped data. Weighting may be used.

*SAS Program Library*

**MEANS** Produces univariate descriptive statistics for numeric variables in an entire data set or for groups of observations in the data set. Options: weights, missing values.

**SUMMARY** Produces univariate descriptive statistics for numeric variables and saves the output. Options: weights, missing values.

**UNIVARIATE** Produces simple descriptive statistics for numeric variables including extreme values and quantiles. Options: distribution plots, frequency table, missing values, weights, normality tests.

*Scientific Desk PC Subprogram Library*

**L1A1C** Computes the mean and second moment about the mean for the sample contained in x. On option, the third and fourth moments can be computed.

*SPSS Program Library*

**CONDESCRIPTIVE** Computes univariate summary statistics (mean, standard deviation, minimum, maximum) for continuous variables and computes standardized variables. Options: standard error of mean, variance, kurtosis, skewness, range, sum, z-scores, missing values.

*STARPAC Subprogram Library*

• **STAT** Compute and print 53 statistics describing the input data.

**STATS** Compute and optionally print 53 statistics describing input data; return statistics.

**STATW** Compute and print 53 statistics describing weighted input data.

**STATWS** Compute and optionally print 53 statistics describing weighted input data; return statistics.

## L1a1a : Location statistics

*DATAPAC Subprogram Library*

**LOC** Computes 4 location estimates (midrange, mean, midmean, and median) of the data in the input vector X.

**MEAN** Computes the sample mean of the data in the input vector X.

**MEDIAN** Computes the sample median of the data in the input vector X.

**MIDM** Computes the sample midmean, i.e., the sample 25% (on each side) trimmed mean of the data in the input vector X.

**MIDR** Computes the sample midrange of the data in the input vector X.

**TRIM** Computes the sample trimmed mean of the data in the input vector X.

**WIND** Computes the sample Windsorized mean of the data in the input vector X.

## L1a1b : Dispersion statistics

*DATAPAC Subprogram Library*

**RANGE** Computes the sample range of the data in the input vector X.

**RELSD** Computes the sample relative standard deviation of the data in the input vector X.

**SCALE** Computes 3 estimates of the scale (variation, scatter, dispersion) of the data in the input vector X.

**SD** Computes the sample standard deviation (with denominator N-1) of the data in the input vector X.

**VAR** Computes the sample variance (with denominator N-1) of the data in the input vector X.

## L1a1c : Shape statistics

*DATAPAC Subprogram Library*

**STMOM3** Computes the sample standardized third central moment of the data in the input vector X.

**STMOM4** Computes the sample standardized fourth central moment of the data in the input vector X.

• **easy-to-use version**

## L1a1d : Frequency and cumulative frequency statistics

*BMDP Program Library*

**P4D** Counts frequency of each number, letter, or symbol in single-column fields (A1 format). Options: input case label variables in A4 format, diagnostic printing useful in preliminary data screening. Specified characters may be replaced by blanks or symbols.

*DATAPAC Subprogram Library*

**COUNT** Computes the number of observations between XMIN and XMAX (inclusively) in the input vector X.

**FREQ** Computes the sample frequency and sample cumulative frequency for the data in the input vector X.

**PROPOR** Computes the sample proportion which is the proportion of data between XMIN and XMAX (inclusively) in the input vector X.

**SAMPP** Computes the sample 100P percent point (where P is between 0.0 and 1.0, exclusively) of the data in the input vector X.

*SPSS Program Library*

**FREQUENCIES** Produces table of frequency counts and percentages for values of individual variables. Options: bar charts, histograms, percentiles, univariate summary statistics, missing values.

## L1a1e : Tie statistics

*IMSL Subprogram Library*

**NMTIE** Tie statistics, given a sample of observations.

*IMSL STAT/LIBRARY Subprogram Library*

**NTIES** Compute tie statistics for a sample of observations.

## L1a3 : Summary statistics for grouped data

*IMSL Subprogram Library*

**BEGRPS** Moments estimation for grouped data with and without Sheppards corrections.

**BEIGRP** Estimation of basic statistical parameters using grouped data.

*IMSL STAT/LIBRARY Subprogram Library*

**GRPES** Compute basic statistics from grouped data.

*NAG Subprogram Library*

**G01ADF** Calculates the mean, standard deviation and coefficients of skewness and kurtosis for data grouped on a frequency distribution.

| **L1b :** | **Summary statistics for two-dimensional data** *(search also class L1c)* |
|---|---|

*DATAPAC Subprogram Library*

**CORR** Computes the sample correlation coefficient between the sets of data in the input vectors X and Y.

**SPCORR** Computes the Spearman rank correlation coefficient between the two sets of data in the input vectors X and Y.

*MINITAB Interactive System*

**CTABLE** Prints a coded two-way table, each cell of which is coded with one character for features MAXIMUM, MINIMUM, or EXTREME, and codes for values between the hinges, between hinges and inner fences, between inner and outer fences, and beyond the outer fences.

*NAG Subprogram Library*

**G01ABF** Computes the means, standard deviations, corrected sums of squares and products, maximum and minimum values, and the product-moment correlation coefficient for two variables. Unequal weighting may be given.

| **L1c1 :** | **Summary statistics for multi-dimensional data** |
|---|---|

*BMDP Program Library*

**P1D** Prints univariate statistics (mean, standard deviation, standard error of mean, coefficient of variation, extreme values, extreme z-scores, range) for each variable. Options: statistics for each level of each grouping variable, sorting, printing all cases OR only cases with values missing or values outside specified limits.

**P2D** For each variable, prints frequency and percent for each distinct value; mean, median, mode, standard deviation, standard errors of mean and median, skewness, kurtosis, half interquartile range; histogram, and stem-and-leaf plot. Options: initially round or truncate, three robust location estimates.

*CMLIB Library (CLUSTER Sublibrary)*

**TWO** Computes overall mean and covariance matrices of a data set with appropriate labels for the cases and variables.

*IMSL Subprogram Library*

**BECOR** Estimates of means, standard deviations, and correlation coefficients (out-of-core version).

**BECORI** Estimates of means, standard deviations, and correlation coefficients (in-core version).

**BECOVM** Means and variance-covariance matrix.

**BECOVW** Means and variance-covariance or correlation matrix from data possibly containing missing observations, with weighting on option.

**BESTAT** Computations of basic univariate statistics from data possibly containing missing values, with weighting on option.

*IMSL STAT/LIBRARY Subprogram Library*

**CSTAT** Compute cell frequencies, cell means, and cell sums of squares for multivariate data.

**UVSTA** Compute basic univariate statistics.

*MINITAB Interactive System*

**MINITAB** Vector summarization commands include COUNT, SUM, MEAN, MAX, MIN, MEDIAN, N, NMISS (number of missing values), STDEV, SSQ (sum of squares), DESCRIBE (N, MEAN, MEDIAN, STDEV, MAX, MIN, 5% trimmed mean, quartiles) for columns or rows (use prefix R, e.g., RMEAN) of data in the Minitab worksheet

**TABLE** Produces and prints one-way, two-way, and multi-way tables of counts with 20 optional subcommands for summarizing (e.g., cell mean, standard deviation), marginals, performing chi-square tests for each 2-way table, handling missing values, and selecting forms of input and output.

*NAG Subprogram Library*

**G02BAF** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in the array X.

**G02BDF** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in the array X.

**G02BGF** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in specified columns of the array X.

**G02BKF** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in specified columns of the array X.

*SAS Program Library*

**MEANS** Produces univariate descriptive statistics for numeric variables in an entire data set or for groups of observations in the data set. Options: weights, missing values.

**SUMMARY** Produces univariate descriptive statistics for numeric variables and saves the output. Options: weights, missing values.

**TABULATE** Produces hierarchical tables of descriptive statistics from compositions of classification variables, analysis variables, and statistics keywords. Options: weights, missing values.

**UNIVARIATE** Produces simple descriptive statistics for numeric variables including extreme values and quantiles. Options: distribution plots, frequency table, missing values, weights, normality tests.

*Scientific Desk PC Subprogram Library*

**L1E1B** Computes the mean, variance estimates, and correlation coefficients for the n observations on the m variables stored in matrix A.

*SPSS Program Library*

**AGGREGATE** Provides summary statistics (sum, mean, standard deviation, percent of observations with specified range) for the values of one variable across groups of sequential observations. Option: missing values.

**BREAKDOWN** Computes means, standard deviations, variances and subpopulation sizes for a dependent variable over subgroups defined by as many as five independent variables. Options: one-way ANOVA, test for linearity, missing values.

**CONDESCRIPTIVE** Computes univariate summary statistics (mean, standard deviation, minimum, maximum) for continuous variables and computes standardized variables. Options: standard error of mean, variance, kurtosis, skewness, range, sum, z-scores, missing values.

**CROSSTABS** Produces tables that are the joint distribution of two or more variables that have a limited number of distinct values. Handles integer, continuous, and character data. Options: summary statistics, several measures of association, missing values.

**REPORT** Produces case listings, univariate summary statistics, absolute and relative frequencies for subpopulations with user control of output appearance. Options: page lengths, column widths, margins, titles, footnotes, labels, concatenation of variables and literals for display, functions of summary statistics, missing values.

**L1c1b : Covariance and correlation statistics**

*IMSL Subprogram Library*

**BECVL** Variances and covariances of linear functions (out-of-core version).

**BECVLI** Variances and covariances of linear functions (in-core version).

*IMSL STAT/LIBRARY Subprogram Library*

**CORVC** Compute the variance-covariance or correlation matrix.

*MINITAB Interactive System*

**CORRELATION** Calculates the Pearson product moment correlation coefficient between two or more pairs of vectors, handles missing values, and optionally saves results.

*NAG Subprogram Library*

**G02BNF** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X; the data array X is overwritten, and on exit it contains the ranks of the observations.

**G02BQF** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X; the data array X is preserved, and the ranks of the observations are not available on exit from the routine.

*SAS Program Library*

**CORR** Computes correlation coefficients between variables, including Pearson product-moment and weighted product-moment correlations. Can also compute Spearman's rank-order correlation, Kendall's tau-b, and Hoeffding's measurement of dependence. Options: some univariate descriptive statistics, missing values.

*SPSS Program Library*

**PEARSON CORR** Produces matrices of Pearson product-moment correlation coefficients with one- or two-tailed significance levels. Options: univariate statistics, covariances, cross-product deviations, missing values.

**L1c1d : Frequency and cumulative frequency statistics for multi-dimensional data**

*BMDP Program Library*

**P4D** Counts frequency of each number, letter, or symbol in single-column fields (A1 format). Options: input case label variables in A4 format, diagnostic printing useful in preliminary data screening. Specified characters may be replaced by blanks or symbols.

*SPSS Program Library*

**FREQUENCIES** Produces table of frequency counts and percentages for values of individual variables. Options: bar charts, histograms, percentiles, univariate summary statistics, missing values.

**L1c2 :** **Summary statistics for multi-dimensional data containing missing values** *(search also class L1c1)*

*BMDP Program Library*

**P8D** Four methods to compute covariance and correlation matrices when data contain missing values or values out of range. Options: weights, summary statistics, save results, pairwise t-tests based on the pattern of incomplete data.

**PAM** Describes pattern of invalid values (missing or out of range) for multivariate data. Options: weights, grouping, estimates covariance and correlation matrices by one of three methods (including maximum likelihood), replace invalid values using means or one of several regression procedures, plots, save results.

*IMSL Subprogram Library*

**BEMMI** Estimates of means, standard deviations, correlation coefficients, and coefficients of skewness and kurtosis from a data matrix containing missing observations (in-core version).

**BEMMO** Estimates of means, std. devs., correlation coefficients, and coefficients of skewness and kurtosis from a data matrix containing missing observations (out-of-core version).

*NAG Subprogram Library*

**G02BBF** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable.

**G02BCF** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing.

**G02BEF** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable.

**G02BFF** Computes means and standard deviations of variables, sums of squares and cross-products about zero and correlation-like coefficients for a set of data in the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing.

**G02BHF** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in specified columns of the array X, omitting completely any cases with a missing observation for any variable (either over all variables in the data array or over only those variables in the selected subset).

**G02BJF** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in specified columns of the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing.

**G02BLF** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in specified columns of the array X, omitting completely any cases with a missing observation for any variable (either over all variables in the data array or over only those variables in the selected subset).

**G02BMF** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in specified columns of the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing.

**G02BPF** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable; the data array X is overwritten, and on exit contains the ranks of the observations.

**G02BRF** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable; the data array X is preserved, and the ranks of the observations are not available on exit from the routine.

**G02BSF** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing; the data array X is preserved, and the ranks of the observations are not available on exit from the routine.

## L2: Data Manipulation

This class is for elementary manipulation of data. Examples are transforming data for normality, centering variables for regression, tallying data for categorical data analysis, and extracting subsets to delete outliers. In one case manipulation is actually creation — of indicator variables, primarily for use in regression.

Class L10a documents data manipulation software for time series analysis; class N6 documents general-purpose sorting and ranking software.

### References

[Atk85] A. Atkinson. *Plots, Transformations and Regression.* Oxford University Press, Oxford, 1985.

[HC78] R. V. Hogg and A. T. Craig. *Introduction to Mathematical Statistics.* Macmillan, New York, 1978.

[NWK88] J. Neter, W. Wasserman, and M. Kutner. *Applied Linear Regression Models.* Richard D. Irwin, Inc., Homewood, Illinois, 2nd edition, 1988.

| **L2 :** | **Data manipulation** |
|---|---|

*BMDP Program Library*

**PDM** An interactive data manipulation system which transforms (re-expresses data, selects cases, edits data, and computes various results), sorts, checks data (for minimum, maximum, missing, etc.), reports summary information including histograms, prints graphical displays, constructs new files from existing files in several ways, and constructs new records from existing records in several ways. The Data Manager can be used with both BMDP files and raw data files.

| **L2a :** | **Data transformation** *(search also classes L10a1, N6, and N8)* |
|---|---|

*BMDP Program Library*

**P1S** At each pass through the data, computes univariate statistics (choose means, standard deviations, geometric means, harmonic means, extreme values), and transforms or edits the data using statistics computed in the previous pass. Options: printing, save results.

*CMLIB Library (CLUSTER Sublibrary)*

**SCALE** Discretizes the data into classes.

**STAND** Standardizes a data matrix such that each variable has mean zero and unit variance. Missing values are ignored.

*DATAPAC Subprogram Library*

**DISCR2** Discretizes the data in the vector X into NUMCLA classes (using midpoint).

**DISCR3** Discretizes the data in the vector X into NUMCLA classes (using class numbers).

**DISCRE** Discretizes the data of the vector X according to class width.

**REPLAC** Replaces (with the value XNEW) all observations in the vector X which are inside the interval [XMIN, XMAX].

*DATAPLOT Interactive System*

**LET** Carries out general transformations of variables and assigns values to variables or parameters for function evaluation. Options: statistics (mean, median, autocorrelation, etc.), mathematical operations (sum, root, derivative, integral, etc.), random number generation (from 24 distributions and families), sequencing, patterning.

**LET FUNCTION** Creates functions. Options: intrinsic functions, concatenation or build-up of functions, definition before or after the parameters and variables used in them are created.

*IMSL Subprogram Library*

**BDTRGI** Transgeneration of the columns of a matrix (in-core version).

**BDTRGO** Transgeneration of the columns of a matrix (out-of-core version).

**FTRDIF** Transformations, differences and seasonal differences of a time series for model identification.

**RLGQMI** Centering of independent variable settings and generation of centered square and cross product terms – in-core version.

**RLGQMO** Centering of independent variable settings and generation of uncentered square and cross product terms – out-of-core version.

**RLPOL** Generate orthogonal polynomials with associated constants AA and BB.

*IMSL STAT/LIBRARY Subprogram Library*

**BCTR** Perform a forward or an inverse Box-Cox (power) transformation.

**GCSCP** Generate centered variables, squares, and crossproducts.

**OPOLY** Generate orthogonal polynomials with respect to a specified interval and weights.

**RANKS** Compute the ranks, normal scores, or exponential scores for a vector of observations.

*MINITAB Interactive System*

**CENTER** Centers data to mean 0, standard deviation 1. Optionally can select location and scale or minimum and maximum.

**DIFFERENCES** Computes all differences X – Y for each value X in one vector and each value Y in a second vector (useful for nonparametric tests and confidence intervals).

**LAG** Computes lagged observations in a time series.

**MINITAB** Vector transformation commands include ADD, SUBTRACT, MULTIPLY, DIVIDE, RAISE, SIN, COS, TAN, ASIN, ACOS, ATAN, LOGE, LOGTEN, EXPONENTIAL, ANTILOG, ABSOLUTE VALUE, ROUND, SIGNS, SQRT, INDICATOR, RECODE, SUBSTITUTE, CONVERT, PARSUM, PARPRODUCT, and LET (to combine commands).

**NSCORES** Calculates normal scores.

**RANK** Ranks the values in a vector. Ties are assigned the average rank.

**WALSH** Calculates $(X_i + X_j)/2$ and stores these averages and their indices (useful for nonparametric tests and confidence intervals).

*SAS Program Library*

**RANK** Computes ranks for one or more numeric variables across the observations of a data set. Options: group continuous data into ranges, fractional ranks, normal scores (Blom, Tukey, or van der Waerden), Savage (exponential) scores.

**STANDARD** Standardizes some or all of the variables in a data set to a given mean and standard deviation. Options: weights, replace missing values with variable mean.

*Scientific Desk PC Subprogram Library*

**L8G1A** Determines an $r$ for the equation $Y_i = b_0 + b_1 X_i^R$ so that $Y_i = b_0 + b_1 Z_i$, where $Z_i = X_i^r$ plots as a straight line. The iteration scheme used for R is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $r$.

**L8G1E** Determines an $a$ for the equation $Y_i = b_0 + b_1 a^{X_i}$ so that $Y_i) = b_0 + b_1 Z_i$, where $Z_i = a^{X_i}$ plots as a straight line. The iteration scheme used for R is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $a$.

*SPSS Program Library*

**MULT RESPONSE** Combines elementary variables into multiple dichotomy groups and multiple response groups to produce univariate and multivariate tables. Options: cell content and percentages, missing values, stub and banner tables.

*STARPAC Subprogram Library*

**CENTER** Subtract the series mean from each observation of a series; return the centered series (no printed output).

## L2b : Tally data

*DATAPLOT Interactive System*

**TABULATE** Produce a table consisting of counts for each distinct value of a variable. Options include calculating either the counts, the means, the standard deviations, or the ranges of one variable for each distinct value of another variable.

*IMSL Subprogram Library*

**BDCOU1** Tally of observations into a one-way frequency table.

**BDCOU2** Tally of observations into a two-way frequency table.

**BDTAB** Computations of frequencies of multivariate data.

**BDTWT** Computations of a two-way frequency table.

**GTDDU** D-square tally.

**GTPL** Poker test tally of hand types and statistics.

**GTPR** Tally of coordinates of pairs (or lagged pairs) of random numbers.

**GTRTN** Tally of number of runs up and down.

**GTTRT** Tally for triplets test.

*IMSL STAT/LIBRARY Subprogram Library*

**FREQ** Tally multivariate observations into a multi-way frequency table.

**OWFRQ** Tally observations into a one-way frequency table.

**TWFRQ** Tally observations into a two-way frequency table.

*NAG Subprogram Library*

**G01AEF** Constructs a frequency distribution of a variable, according to either user-supplied, or routine-calculated class boundary values.

**L2c : Subset extraction**

*DATAPAC Subprogram Library*

**DELETE** Deletes all observations in the vector X which are inside the interval [XMIN, XMAX].

**RETAIN** Retains all observations in the vector X which are inside the interval [XMIN, XMAX].

**SUBSE1** Carries over into Y all observations of vector X for which the corresponding elements in vector D are in the interval [DMIN, DMAX].

**SUBSE2** Carries over into Y all observations of vector X for which the corresponding elements in vector D1 are in the inclusive interval [D1MIN,D1MAX] and also for which the corresponding elements in D2 are in the interval [D2MIN,D2MAX].

**SUBSET** Retain all observations in vector X for which the corresponding elements in vector D are in the interval [DMIN,DMAX].

*MINITAB Interactive System*

**MINITAB** Subsetting commands include PICK, CHOOSE, and OMIT for selecting or deleting entries in a vector in a Minitab worksheet.

*NAG Subprogram Library*

**G02CEF** Takes selected elements from two vectors (typically vectors of means and standard deviations) to form two smaller vectors, and selected rows and columns from two matrices (typically either matrices of sums of squares and cross-products of deviations from means and Pearson product-moment correlation coefficients, or matrices of sums of squares and cross-products about zero and correlation-like coefficients) to form two smaller matrices, allowing re-ordering of elements in the process.

*STARPAC Subprogram Library*

**SAMPLE** Sample (extract) every NSth observation from a series; return sampled series (no printed output).

**L2d : Merge data sets**

*MINITAB Interactive System*

**JOIN** Merges constants and/or vectors into vectors.

**L2e : Construct new variables (e.g., indicator variables)**

*DATAPLOT Interactive System*

**LET** Carries out general transformations of variables and assigns values to variables or parameters for function evaluation. Options: statistics (mean, median, autocorrelation, etc.), mathematical operations (sum, root, derivative, integral, etc.), random number generation (from 24 distributions and families), sequencing, patterning.

*MINITAB Interactive System*

**SET** Create a constant vector or a vector of integers in increments of 1 or more or with other patterns.

## L3: Elementary Statistical Graphics

The software documented in this class is for general-purpose statistical graphics. This includes histograms (and their bar and pie chart variants) and many types of frequency plots and scatter diagrams. It also includes newer EDA (exploratory data analysis) methods such as box-plots (for displaying a comprehensive set of summary statistics in an intuitively easily understood manner) and stem-and-leaf plots, a variant of the histogram. Graphics associated with a particular type of analysis is in the class for that type of analysis (for example, cluster analysis graphics are in class L14c).

This software is first organized by the dimension of the data set being plotted (not the dimension of the plot). Thus, for example, class L3a for one-dimensional data contains subclass L3a6 for the two-dimensional plot of $X_i$ vs. $i$. Three-dimensional data are displayed on two-dimensional surfaces.

Most of the software in class L3e handles multi-dimensional data in a pairwise fashion. For example, scatter diagrams for multi-dimensional data are superimposed scatter diagrams for pairs of variables or are matrices of such scatter diagrams.

## References

[Cle85] W. S. Cleveland. *The Elements of Graphing Data.* Wadsworth, Monterey, CA, 1985.

[Mey75] S. L. Meyer. *Data Analysis for Scientists and Engineers.* John Wiley & Sons, New York, 1975.

[MT77] F. Mosteller and J. W. Tukey. *Data Analysis and Regression: A Second Course in Statistics.* Addison-Wesley, Reading, MA, 1977.

[Sch83] C. F. Schmid. *Statistical Graphics: Design Principles and Practices.* Wiley-Interscience, New York, 1983.

[Tuf83] E. R. Tufte. *The Visual Display of Quantitative Information.* Graphics Press, Cheshire, Connecticut, 1983.

[tuk77] J. W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, Reading, MA, 1977.

### L3a1 : Histograms

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*CMLIB Library (CLUSTER Sublibrary)*

**HIST** Prints a univariate histogram.

*DATAPAC Subprogram Library*

**HIST** Produces 2 histograms (with differing class widths) of the data in the input vector X.

**PLOTU** Produces 4 plots: data plot ($X_i$ vs. i), autoregression plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

*DATAPLOT Interactive System*

**4-PLOT** Produces 4 plots: $X_i$ vs. $i$, lag plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

**HISTOGRAM** Generates one of the following plots: histogram, relative histogram, cumulative histogram, relative cumulative histogram. Optional control of upper and lower limits and/or class width.

*IMSL Subprogram Library*

**USHHST** Print a horizontal histogram.

**USHST** Print a vertical histogram.

*IMSL STAT/LIBRARY Subprogram Library*

**HHSTP** Print a horizontal histogram.

**VHSTP** Print a vertical histogram.

*MINITAB Interactive System*

**HISTOGRAM** Prints a histogram of the values in each of one or more vectors, with optional user-specification of the first midpoint and the interval width.

*NAG Subprogram Library*

**G01AJF** Prints a histogram on a character printing device, with user control over size, positioning, and the range of data values included.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

*SPSS Program Library*

**FREQUENCIES** Produces table of frequency counts and percentages for values of individual variables. Options: bar charts, histograms, percentiles, univariate summary statistics, missing values.

*STARPAC Subprogram Library*

• **HIST** Compute and print a histogram and summary statistics, with automatic selection of number of cells and cell boundaries.

**HISTC** Compute and print a histogram and summary statistics with user control of number of cells and cell boundaries.

## L3a2 : Frequency, cumulative frequency, and percentile plots

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*DATAPLOT Interactive System*

**FREQUENCY PLOT** Generates one of the following plots: frequency, relative frequency, cumulative frequency, relative cumulative frequency. Optional control of upper and lower limits and/or class width.

**PERCENT POINT PLOT** Generates a percent point plot (cumulative distribution plot with axes interchanged). Optional control of upper and lower limits and/or class width.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

**UNIVARIATE** Produces simple descriptive statistics for numeric variables including extreme values and quantiles. Options: distribution plots, frequency table, missing values, weights, normality tests.

*SPSS Program Library*

**FREQUENCIES** Produces table of frequency counts and percentages for values of individual variables. Options: bar charts, histograms, percentiles, univariate summary statistics, missing values.

## L3a3 : EDA graphics

*DATAPLOT Interactive System*

**BOX PLOT** Plots box plot(s) – median, "body" between lower and upper hinges, tails (between hinges and minimum or maximum) – for either univariate distributional analysis or analysis of 1-factor models.

**HOMOSCEDASTICITY PLOT** Generate a homoscedasticity plot with horizontal axis the subset standard deviation and vertical axis the subset mean.

**I PLOT** Plots I PLOT(s) – median, minimum, and maximum – for either univariate distributional analysis, analysis of one-factor models, or display of uncertainty intervals about estimates. Default plot character for the estimator is "x" and for the top and the bottom of the uncertainty interval is "-".

**STATISTICS PLOT** For one of over 30 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics, generate a plot with the vertical axis containing the value of the statistic for each distinct subset of another variable while the horizontal axis is the subset identifier. Common statistics are the mean, the standard deviation, the linear-intercept, the linear-slope, the linear RESSD, and the linear-correlation.

**STEM AND LEAF DIAGRAM** Generates a stem and leaf plot of a one-dimensional data set.

**SYMMETRY PLOT** Generate a symmetry plot. The horizontal axis contains the sorted $X_i$ for $i = N, N-1, \ldots, N/2$ (or $N/2+1$) while the vertical axis contains the sorted $-X_i$ for $i = 1, 2, \ldots, N/2$ (or $N/2+1$). Symmetric data sets should approximate a 45 degree line.

*IMSL Subprogram Library*

**USBOX** Print a boxplot (k samples).

**USSLF** Print a stem-and-leaf display.

*IMSL STAT/LIBRARY Subprogram Library*

**BOXP** Print boxplots for one or more samples.

**STMLP** Print a stem-and-leaf plot.

*MINITAB Interactive System*

**BOXPLOT** Prints boxplots – median, hinges, inner and outer fences – for one or more levels. Options: form of plots, notches (confidence interval for population medians).

**STEM-&-LEAF** Prints stem-and-leaf display(s), optionally with no trimming of outliers.

## L3a4 : Bar charts

*DATAPLOT Interactive System*

**PARETO PLOT** Generate a Pareto plot with horizontal axis an index variable and vertical axis typically plotted as a bar or a spike with a labelled character and ordered from the largest to the smallest value.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

*SPSS Program Library*

**FREQUENCIES** Produces table of frequency counts and percentages for values of individual variables. Options: bar charts, histograms, percentiles, univariate summary statistics, missing values.

## L3a5 : Pie charts

*DATAPLOT Interactive System*

**PIE CHART** Generates a pie chart for presentation rather that analysis purposes. Optional control of upper and lower limits and/or class widths.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

## L3a6 : Plots of $X_i$ vs. $i$ (including symbol plots)

*BMDP Program Library*

**P1T** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), missing values, remove seasonal means and linear trend, filtering, save results.

*DATAPAC Subprogram Library*

**PLOTU** Produces 4 plots: data plot ($X_i$ vs. $i$), autoregression plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

**PLOTX** Yields a one-page printer plot of $X_i$ vs. $i$.

**PLOTXT** Yields a narrow-width (71-character) plot of $X_i$ vs. $i$.

*DATAPLOT Interactive System*

**4-PLOT** Produces 4 plots: $X_i$ vs. $i$, lag plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

**ERROR BAR PLOT** Generates an $X_i$ vs $i$ plot or a Y vs X plot with user specified error limits. Separate positive and negative error limits can be specified. Error limits are specified for the vertical variable only, not for the horizontal variable.

**PLOT** Plots a single or multi-trace plot of data, functions, or both. Built-in function library includes elementary, trigonometric, logarithmic, and probability functions.

**RUN SEQUENCE PLOT** Generates a run sequence plot (variable vs. dummy index).

*MINITAB Interactive System*

**TSPLOT** Prints a scatter diagram of a time series, optionally using symbols modulo the period. Handles missing values.

*SAS Program Library*

**TIMEPLOT** Produces a line printer plot of one or more variables over time intervals. Options: missing values, user control of plot features.

*STARPAC Subprogram Library*

• **SVP** Print vertical plot of Y versus input order with individual plot symbols specified by user; linear axis; default control values and axis limits; no missing values allowed.

**SVPC** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; user-supplied control values and axis limits; no missing values allowed.

**SVPL** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; default control values and axis limits; no missing values allowed.

**SVPM** Print vertical plot of Y versus input order with individual plot symbols specified by user; linear axis; default control values and axis limits; missing values allowed.

**SVPMC** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; user-supplied control values and axis limits; missing values allowed.

**SVPML** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; default control values and axis limits; missing values allowed.

• **VP** Print vertical plot of Y versus input order; linear axes; default control values and axis limits; no missing values allowed.

**VPC** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; user-supplied control values and axis limits; no missing values allowed.

**VPL** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; default control values and axis limits; no missing values allowed.

**VPM** Print vertical plot of Y versus input order; linear axis; default control values and axis limits; missing values allowed.

**VPMC** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; user-supplied control values and axis limits; missing values allowed.

**VPML** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; default control values and axis limits; missing values allowed.

## L3a7 : Lag plots (e.g., plots of $X_i$ vs. $X_{i-1}$)

*BMDP Program Library*

**P1T** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), missing values, remove seasonal means and linear trend, filtering, save results.

*DATAPAC Subprogram Library*

**PLOTU** Produces 4 plots: data plot ($X_i$ vs. $i$), autoregression plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

**PLOTXX** Yields a one-page printer plot of $X_i$ versus $X_{i-1}$ for testing autocorrelation.

**PLTXXT** Yields a narrow-width (71-character) plot of $X_i$ versus $X_{i-1}$ for testing autocorrelation.

• easy-to-use version

*DATAPLOT Interactive System*

**4-PLOT** Produces 4 plots: $X_i$ vs. $i$, lag plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

**LAG PLOT** Plots a lag plot (for a user-specified lag) for equi-spaced univariate or bivariate time series data.

| **L3b1 :** | **Histograms for two-dimensional data** |
|---|---|

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*CMLIB Library (CLUSTER Sublibrary)*

**MODAL** Produces a bivariate histogram.

*DATAPLOT Interactive System*

**BIHISTOGRAM** Generate a bi-histogram of two-dimensional data.

*IMSL Subprogram Library*

**USHST2** Print a vertical histogram, plotting two frequencies with one bar of the histogram.

*IMSL STAT/LIBRARY Subprogram Library*

**VHS2P** Print a vertical histogram with every bar subdivided into two parts.

| **L3b2 :** | **Frequency and cumulative frequency plots for two-dimensional data** |
|---|---|

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*IMSL Subprogram Library*

**USPDF** Plot of two sample probability distribution functions against their spectra.

*IMSL STAT/LIBRARY Subprogram Library*

**CDF2P** Print a plot of two sample cumulative distribution functions.

| **L3b3a :** | **Y vs. X scatter diagrams** |
|---|---|

*BMDP Program Library*

**P6D** Bivariate (scatter) plots. Options: several variables, or subsets of one variable (symbols identify group membership), on the same plot; prints correlation and linear regression statistics (line is marked on plot frame); user control for plot size, scales, and symbols.

*DATAPAC Subprogram Library*

• **PLOT** Yields a one-page printer plot of $Y_i$ versus $X_i$.

**PLOT6** Yields a one-page printer plot of $Y_i$ versus $X_i$ for specified axis limits.

**PLOTS** Yields a one-page printer plot of $Y_i$ versus $X_i$ for a subset of the data.

**PLOTST** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$ for a subset of the data.

**PLOTT** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$.

*DATAPLOT Interactive System*

**ERROR BAR PLOT** Generates an $X_i$ vs $i$ plot or a Y vs X plot with user specified error limits. Separate positive and negative error limits can be specified. Error limits are specified for the vertical variable only, not for the horizontal variable.

**PLOT** Plots a single or multi-trace plot of data, functions, or both. Built-in function library includes elementary, trigonometric, logarithmic, and probability functions.

*MINITAB Interactive System*

**CPLOT** Prints a scatter diagram which condenses as many as 10 lines of plot into one line and trims extreme x- and y-values. Option: form of output.

**PLOT** Prints a scatter diagram, with optional scale specification.

*NAG Subprogram Library*

**G01AGF** Performs a scatter plot of two variables on a character printing device, with a chosen number of character positions in each direction.

*SAS Program Library*

**PLOT** Produces a Y vs. X line printer scatterplot, a superimposed plot, or a contour plot. Options: missing values, user control of plot features.

*SPSS Program Library*

**PLOT** Produces two-dimensional line-printer plots. Options: format (bivariate scatterplot, contour, overlay, regression), labels, scales, size, symbols, missing values.

*STARPAC Subprogram Library*

• **PP** Print Y versus X scatterplot; linear axes; default control values and axis limits; no missing values allowed.

**PPC** Print Y versus X scatterplot; log or linear axes; user-supplied control values and axis limits; no missing values allowed.

**PPL** Print Y versus X scatterplot; log or linear axes; default control values and axis limits; no missing values allowed.

**PPM** Print Y versus X scatterplot; linear axes; default control values and axis limits; missing values allowed.

**PPMC** Print Y versus X scatterplot; log or linear axes; user-supplied control values and axis limits; missing values allowed.

**PPML** Print Y versus X scatterplot; log or linear axes; default control values and axis limits; missing values allowed.

## L3b3b : Symbol plots

*DATAPAC Subprogram Library*

**PLOT10** Yields a one-page printer plot of $Y_i$ versus $X_i$ for a subset of the data, with special plot characters, and with specified axis limits and labels.

**PLOT7** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plot characters and for specified axis limits.

**PLOT8** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plot characters for a subset of the data with specified axis limits.

**PLOT9** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plot characters and for specified axis limits and axis labels.

• **PLOTC** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plotting characters.

**PLOTCT** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$ with special plotting characters.

**PLOTSC** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special characters for a subset of the data.

**PLTSCT** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$ with special plot characters and a a subset of the data.

*MINITAB Interactive System*

**LPLOT** Prints a letter plot with symbols corresponding to numerical "tag" values. Scale specification is optional.

*STARPAC Subprogram Library*

• **SPP** Print Y versus X scatterplot with individual plot symbols specified by user; linear axes; default control values and axis limits; no missing values allowed.

**SPPC** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axes; user-supplied control values and axis limits; no missing values allowed.

**SPPL** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axes; default control values and axis limits; no missing values allowed.

**SPPM** Print Y versus X scatterplot with individual plot symbols specified by user; linear axes; default control values and axis limits; missing values allowed.

**SPPMC** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axes; user-supplied control values and axis limits; missing values allowed.

**SPPML** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axis; default control values and axis limits; missing values allowed.

## L3b3c : Lag plots (i.e., plots of $X_i$ vs. $Y_{i-j}$)

*BMDP Program Library*

**P1T** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), missing values, remove seasonal means and linear trend, filtering, save results.

*DATAPLOT Interactive System*

**LAG PLOT** Plots a lag plot (for a user-specified lag) for equi-spaced univariate or bivariate time series data.

## L3b4 : EDA plots for two-dimensional data

*DATAPLOT Interactive System*

**STATISTICS PLOT** For one of over 30 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics, generate a plot with the vertical axis containing the value of the statistic for each distinct subset of another variable while the horizontal axis is the subset identifier. Common statistics are the mean, the standard deviation, the linear-intercept, the linear-slope, the linear RESSD, and the linear-correlation.

| **L3c :** | **Graphics for three-dimensional data** *(search also class L3e)* |
|---|---|

*DATAPLOT Interactive System*

**3D-PLOT** Generates a single or multi-surface 3-dimensional plot of data, functions, or both.

*MINITAB Interactive System*

**TPLOT** Prints pseudo three-dimensional plot of y versus x versus z, with symbols indicating the values of z, and with optional scale specification.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

**PLOT** Produces a Y vs. X line printer scatterplot, a superimposed plot, or a contour plot. Options: missing values, user control of plot features.

*SPSS Program Library*

**PLOT** Produces two-dimensional line-printer plots. Options: format (bivariate scatterplot, contour, overlay, regression), labels, scales, size, symbols, missing values.

| **L3e1 :** | **Histograms for multi-dimensional data** |
|---|---|

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*CMLIB Library (CLUSTER Sublibrary)*

**MHIST** Produces multivariate histograms.

*MINITAB Interactive System*

**HISTOGRAM** Prints a histogram of the values in each of one or more vectors, with optional user-specification of the first midpoint and the interval width.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

| **L3e2 :** | **Frequency, cumulative frequency, and percentile plots for multi-dimensional data** |
|---|---|

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

## L3e3a : Superimposed $Y$ vs. $X$ scatter diagrams

*BMDP Program Library*

**P6D** Bivariate (scatter) plots. Options: several variables, or subsets of one variable (symbols identify group membership), on the same plot; prints correlation and linear regression statistics (line is marked on plot frame); user control for plot size, scales, and symbols.

*CMLIB Library (CLUSTER Sublibrary)*

**PLOT** Produces a scatter plot of several Y variables vs. one X variable.

*DATAPLOT Interactive System*

**PLOT** Plots a single or multi-trace plot of data, functions, or both. Built-in function library includes elementary, trigonometric, logarithmic, and probability functions.

*IMSL STAT/LIBRARY Subprogram Library*

**PLOTP** Print a plot of up to ten sets of points.

**SCTP** Print a scatterplot of several groups of data.

*MINITAB Interactive System*

**MPLOT** Prints multiple scatter diagrams on the same axis.

*SAS Program Library*

**PLOT** Produces a Y vs. X line printer scatterplot, a superimposed plot, or a contour plot. Options: missing values, user control of plot features.

*SPSS Program Library*

**PLOT** Produces two-dimensional line-printer plots. Options: format (bivariate scatterplot, contour, overlay, regression), labels, scales, size, symbols, missing values.

*STARPAC Subprogram Library*

• **MPP** Print plot of multiple Y vectors versus a common X vector; linear axes; default control values and axis limits; no missing values allowed.

**MPPC** Print plot of multiple Y vectors versus a common X vector; log or linear axes; user-supplied control values and axis limits; no missing values allowed.

**MPPL** Print plot of multiple Y vectors versus a common X vector; log or linear axes; default control values and axis limits; no missing values allowed.

**MPPM** Print plot of multiple Y vectors versus a common X vector; linear axes; default control values and axis limits; missing values allowed.

**MPPMC** Print plot of multiple Y vectors versus a common X vector; log or linear axes; user-supplied control values and axis limits; missing values allowed.

**MPPML** Print plot of multiple Y vectors versus a common X vector; log or linear axes; default control values and axis limits; missing values allowed.

## L3e3c : Superimposed $X_i$ vs. $i$ scatter diagrams

*BMDP Program Library*

**P1T** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), missing values, remove seasonal means and linear trend, filtering, save results.

*DATAPLOT Interactive System*

**PLOT** Plots a single or multi-trace plot of data, functions, or both. Built-in function library includes elementary, trigonometric, logarithmic, and probability functions.

**YOUDEN PLOT** Generates a Youden plot (to carry out an interlab comparison). The analysis can be either where each lab has made two runs on the same product or one run on two different products.

*SAS Program Library*

**TIMEPLOT** Produces a line printer plot of one or more variables over time intervals. Options: missing values, user control of plot features.

*STARPAC Subprogram Library*

• **MVP** Print vertical plot of multiple vectors versus input order on a linear axis, using default control values and axis limits, with no missing values allowed.

**MVPC** Print vertical plot of multiple vectors versus input order on a log or linear axis, using user-supplied control values and axis limits, with no missing values allowed.

**MVPL** Print vertical plot of multiple vectors versus input order on a log or linear axis, using default control values and axis limits, with no missing values allowed.

**MVPM** Print vertical plot of multiple vectors versus input order on a linear axis, using default control values and axis limits, with no missing values allowed.

**MVPMC** Print vertical plot of multiple vectors versus input order on a log or linear axis, using user-supplied control values and axis limits, with no missing values allowed.

**MVPML** Print vertical plot of multiple vectors versus input order on a log or linear axis, using default control values and axis limits, with no missing values allowed.

## L3e3d : Matrices of bivariate scatter diagrams

*CMLIB Library (CLUSTER Sublibrary)*

**PMANY** Produces a matrix of scatter plots between $Y_i$ and $Y_j$ for for $i$ and $j$ between 1 and n where n is the number of variables. For $i = j$, a histogram for variable i is produced.

*DATAPLOT Interactive System*

**MULTIPLOT** Generates a matrix of bivariate plots; valid with any DATAPLOT plot command.

## L3e4 : EDA graphics for multi-dimensional data

*CMLIB Library (CLUSTER Sublibrary)*

**BOXES** Produces a plot in the form of a 3-dimensional box of three of more variables for each case.

**LINE** Produces a line profile of the variables for each case.

**PROF** Computes and outputs profiles of the variables such that the linear regression lines through each case have the smallest total error.

**RANK** Produces rank profiles by ranking cases for each variable and reordering variables to minimize crossings.

*DATAPLOT Interactive System*

**BOX PLOT** Plots box plot(s) – median, "body" between lower and upper hinges, tails (between hinges and minimum or maximum) – for either univariate distributional analysis or analysis of 1-factor models.

• **easy-to-use version**

**I PLOT** Plots I PLOT(s) – median, minimum, and maximum – for either univariate distributional analysis, analysis of one-factor models, or display of uncertainty intervals about estimates. Default plot character for the estimator is "x" and default for the top and the bottom of the uncertainty interval is "-".

**PROFILE PLOT** Generate a profile plot.

**STAR PLOT** Generate a star plot.

*IMSL Subprogram Library*

**USBOX** Print a boxplot (k samples).

*IMSL STAT/LIBRARY Subprogram Library*

**BOXP** Print boxplots for one or more samples.

*SAS Program Library*

**CHART** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots.

## L4: Elementary Data Analysis

This class differs from class L1 (Data Summarization) in that here the data are hypothesized to have some statistical distribution and estimates or inferences are made about that distribution. Software for more sophisticated statistical analyses are documented in classes L7 through L16.

The organization of this class parallels that of class L1. Thus, the software is distinguished first by the dimension of the data and second by the form of the data (raw, grouped, etc.). Class L4b also distinguishes between pairwise independent data (e.g., the widths of lines on two sets of silicon wafers) and pairwise dependent data (e.g., the widths and the spacings of lines on one set of wafers).

The L4 classification scheme has been substantially revised to incorporate more statistical software, particularly graphics software such as probability plots and probability plot correlation coefficient (PPCC) plots. In the former, the smallest observed value is plotted against the expected value of the smallest observed value in a sample of the same size, and similarly for the next smallest, etc. Probability plots better expose tail behavior than plots of empirical and theoretical distribution functions. PPCC plots are used to find a particular distribution (taking the form of a parameter) in a family of parameterized distributions which best fits the data by plotting the correlation coefficients of the probability plots for a set of parameters for that family of distributions. Graphical or other analyses for determining whether data fit a distribution well may usefully precede estimating distributional parameters and hypothesis tests about those parameters.

Software for nonparametric (or distribution-free) analyses are available when modeling parametric distributions is not appropriate. The statistics used may be different (e.g., the median rather than the mean may have the most useful statistical properties), and newer computationally-intensive methods, such as the bootstrap and the jacknife for estimating the sampling distribution, are available.

## References

[Efr82] B. Efron. *The Jackknife, the Bootstrap and Other Resampling Plans.* SIAM, Philadelphia, 1982.

[Mey75] S. L. Meyer. *Data Analysis for Scientists and Engineers.* John Wiley & Sons, New York, 1975.

[MGH89] R.L. Mason, R.F. Gunst, and J.L. Hess. *Statistical Design and Analysis of Experiments.* John Wiley & Sons, New York, 1989.

[MT77] F. Mosteller and J. W. Tukey. *Data Analysis and Regression: A Second Course in Statistics.* Addison-Wesley, Reading, MA, 1977.

[RW79] R. H. Randles and D. A. Wolfe. *Introduction to the Theory of Nonparametric Statistics.* Wiley-Interscience, New York, 1979.

[Tuk77] J. W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, Reading, MA, 1977.

**L4a1a1 : Plots of empirical and theoretical density and distribution functions**

*DATAPLOT Interactive System*

**ROOTOGRAM** Generate a rootogram plot, i.e. a histogram based on the square roots of the counts of data values in each bin.

*IMSL Subprogram Library*

**USPC** Print a sample pdf, a theoretical pdf and confidence band information; plot these on option.

*IMSL STAT/LIBRARY Subprogram Library*

**CDFP** Print a sample cumulative distribution function (CDF), a theoretical CDF, and confidence band information.

*MINITAB Interactive System*

**ROOTOGRAM** Prints a suspended rootogram, i.e. a histogram which has been fit with a Gaussian distribution based on square roots of the counts of data values in each bin and which uses medians and hinges. Options: specify Gaussian mean and standard deviation, save results.

**L4a1a2b : Beta and binomial probability plots**

*DATAPLOT Interactive System*

**BETA PROBABILITY PLOT** Generates a probability plot for the beta distribution with parameters $\alpha$ and $\beta$.

**BINOMIAL PROBABILITY PLOT** Generates a probability plot for the binomial distribution with parameters P and N.

**L4a1a2c : Cauchy and chi-squared probability plots**

*DATAPAC Subprogram Library*

**CAUPLT** Generates a Cauchy probability plot with median 0 and 75% point 1.

**CHSPLT** Generates a chi-squared probability plot with integer degrees of freedom parameter value $\nu$.

*DATAPLOT Interactive System*

**CAUCHY PROBABILITY PLOT** Generates a probability plot for the Cauchy distribution with median 0 and 75% point 1.

**CHI-SQUARED PROBABILITY PLOT** Generates a probability plot for the chi-squared distribution with integer degrees of freedom $\nu$.

**L4a1a2d : Double exponential probability plots**

*DATAPAC Subprogram Library*

**DEXPLT** Generates a double exponential (Laplace) probability plot with mean 0 and standard deviation $\sqrt{2}$.

*DATAPLOT Interactive System*

**DOUBLE EXPONENTIAL PROBABILITY PLOT** Generates a probability plot for the double exponential distribution with mean 0 and standard deviation $\sqrt{2}$. Alternate name: LAPLACE PROBABILITY PLOT.

| L4a1a2e : | Exponential and extreme value probability plots |
|---|---|

*DATAPAC Subprogram Library*

**EV1PLT** Generates an extreme value type 1 probability plot with mean Euler's number 0.57721566 and standard deviation $\pi/\sqrt{6}$.

**EV2PLT** Generates an extreme value type 2 probability plot with tail length parameter $\gamma$.

**EXPPLT** Generates an exponential probability plot with mean 1 and standard deviation 1.

*DATAPLOT Interactive System*

**EXPONENTIAL PROBABILITY PLOT** Generates a probability plot for the exponential distribution with mean 1 and standard deviation 1.

**EXTREME VALUE TYPE 1 PROBABILITY PLOT** Generate a probability plot for the extreme value type 1 distribution with mean Euler's number 0.57721566 and standard deviation $\pi/\sqrt{6}$.

**EXTREME VALUE TYPE 2 PROBABILITY PLOT** Generates a probability plot for the extreme value type 2 distribution with tail parameter $\gamma$.

*IMSL Subprogram Library*

**USPRP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**PROBP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

| L4a1a2f : | F distribution probability plots |
|---|---|

*DATAPLOT Interactive System*

**F PROBABILITY PLOT** Generates a probability plot for the F-distribution with degrees of freedom parameters $\nu_1$ and $\nu_2$.

| L4a1a2g : | Gamma and geometric probability plots |
|---|---|

*DATAPAC Subprogram Library*

**GAMPLT** Generates a gamma probability plot with tail length parameter $\gamma$, mean $\gamma$, and standard deviation $\sqrt{\gamma}$.

**GEOPLT** Generates a geometric probability plot with parameter P.

*DATAPLOT Interactive System*

**GAMMA PROBABILITY PLOT** Generates a gamma probability plot with tail length parameter $\gamma$, mean $\gamma$, and standard deviation $\sqrt{\gamma}$.

**GEOMETRIC PROBABILITY PLOT** Generates a probability plot for the geometric distribution with parameter P.

## L4a1a2h : Halfnormal probability plots

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*DATAPAC Subprogram Library*

**HFNPLT** Generates a halfnormal probability plot with mean $\sqrt{2/\pi}$ and standard deviation 1.

*DATAPLOT Interactive System*

**HALFNORMAL PROBABILITY PLOT** Generates a halfnormal probability plot with mean $\sqrt{2/\pi}$ and standard deviation 1.

*IMSL Subprogram Library*

**USPRP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**PROBP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

## L4a1a2l : Lambda, logistic, and lognormal probability plots

*DATAPAC Subprogram Library*

**LAMPLT** Generates a (Tukey) lambda distribution probability plot with tail length parameter $\lambda$.

**LGNPLT** Generates a lognormal probability plot with mean $\sqrt{e}$.

**LOGPLT** Generates a logistic probability plot with mean 0 and standard deviation $\pi/\sqrt{3}$.

*DATAPLOT Interactive System*

**LAMBDA PROBABILITY PLOT** Generates a probability plot for the Tukey lambda distribution with tail length parameter $\lambda$.

**LOGISTIC PROBABILITY PLOT** Generates a probability plot for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$.

**LOGNORMAL PROBABILITY PLOT** Generates a probability plot for the log normal distribution with mean $\sqrt{e}$.

*IMSL Subprogram Library*

**USPRP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**PROBP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

| **L4a1a2n :** | **Negative binomial and normal probability plots** |
|---|---|

*BMDP Program Library*

**P5D** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options.

*DATAPAC Subprogram Library*

**NORPLT** Generates a normal (Gaussian) probability plot with mean 0 and standard deviation 1.

**PLOTU** Produces 4 plots: data plot ($X_i$ vs. $i$), autoregression plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

*DATAPLOT Interactive System*

**4-PLOT** Produces 4 plots: $X_i$ vs. $i$, lag plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot.

**NEGATIVE BINOMIAL PROBABILITY PLOT** Generates a probability plot for the negative binomial distribution with parameters P and K.

**NORMAL PROBABILITY PLOT** Generates a probability plot for the standard normal (Gaussian) distribution. Alternate name: GAUSSIAN PROBABILITY PLOT.

*IMSL Subprogram Library*

**USPRP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**PROBP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

*NAG Subprogram Library*

**G01AHF** Performs a scatter plot of a vector against the normal scores for a sample of the same size, on a character printing device, with a chosen number of character positions in each direction.

| **L4a1a2p :** | **Pareto and Poisson probability plots** |
|---|---|

*DATAPAC Subprogram Library*

**PARPLT** Generates a Pareto probability plot with tail length parameter $\gamma$.

**POIPLT** Generates a Poisson probability plot with tail length parameter $\lambda$, mean $\lambda$ and standard deviation $\sqrt{\lambda}$.

*DATAPLOT Interactive System*

**PARETO PROBABILITY PLOT** Generates a probability plot for the Pareto distribution with tail length parameter $\gamma$.

**POISSON PROBABILITY PLOT** Generates a probability plot for the Poisson distribution with tail length parameter $\lambda$, mean $\lambda$ and standard deviation $\sqrt{\lambda}$.

| **L4a1a2s :** | **Semicircular probability plots** |
|---|---|

*DATAPLOT Interactive System*

**SEMI-CIRCULAR PROBABILITY PLOT** Generates a probability plot for the semicircular distribution (with density $f(x) = (2/\pi)(1 + x^2)$ for $x$ in the interval [-1,1]).

| L4a1a2t : | t and triangular probability plots |
|---|---|

*DATAPAC Subprogram Library*

**TPLT** Generates a Student's t probability plot with degrees of freedom parameter $\nu$.

*DATAPLOT Interactive System*

**T PROBABILITY PLOT** Generates a probability plot for the Student's t distribution with degrees of freedom parameter $\nu$.

**TRIANGULAR PROBABILITY PLOT** Generates a probability plot for the triangular distribution (with density $f(x) = 1 - |x|$ for $x$ in the interval [-1,1]).

| L4a1a2u : | Uniform probability plots |
|---|---|

*DATAPAC Subprogram Library*

**UNIPLT** Generates a uniform probability plot on the unit interval (0,1) with mean 0.5 and standard deviation $\sqrt{1/12}$.

*DATAPLOT Interactive System*

**UNIFORM PROBABILITY PLOT** Generates a probability plot for uniform distribution (with density $f(x) = x$ for $x$ on the interval [0,1], mean 0.5 and standard deviation $\sqrt{1/12}$).

| L4a1a2w : | Weibull probability plots |
|---|---|

*DATAPAC Subprogram Library*

**WEIPLT** Generates a Weibull probability plot with tail length parameter $\gamma$.

*DATAPLOT Interactive System*

**WEIBULL PLOT** Generates a Weibull plot to assess the goodness of fit of a data set to a two parameter Weibull distribution.

**WEIBULL PROBABILITY PLOT** Generates a Weibull probability plot with tail length parameter $\gamma$.

*IMSL Subprogram Library*

**USPRP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**PROBP** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution.

| L4a1a3c : | Chi-squared probability plot correlation coefficient plots |
|---|---|

*DATAPLOT Interactive System*

**CHI-SQUARED PPCC PLOT** Generates a probability plot correlation coefficient plot for the chi-squared distribution (plot of probability plot correlation coefficient vs. degrees of freedom parameter $\nu$ for $\nu$ ranging from 1 to 100 or in user-set range).

## L4a1a3e : Extreme value probability plot correlation coefficient plots

*DATAPLOT Interactive System*

**EXTREME VALUE TYPE 2 PPCC PLOT** Generates a probability plot correlation coefficient plot for the extreme value type 2 distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging from 1 to 100 or in user-set range).

## L4a1a3g : Gamma and geometric probability plot correlation coefficient plots

*DATAPLOT Interactive System*

**GAMMA PPCC PLOT** Generates a probability plot correlation coefficient plot for the gamma distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging form 1 to 100 or in user-set range).

**GEOMETRIC PPCC PLOT** Generates a probability plot correlation coefficient plot for the geometric distribution (plot of probability plot correlation coefficient vs. parameter P for P ranging from 0 to 1 or in user-set range).

## L4a1a3l : Lambda probability plot correlation coefficient plots

*DATAPLOT Interactive System*

**LAMBDA PPCC PLOT** Generates a probability plot correlation coefficient plot for Tukey lambda distribution (plot of probability plot correlation coefficient vs. the tail parameter $\lambda$ for $\lambda$ ranging from -2 to 2 or in user-set range).

## L4a1a3n : Normal probability plot correlation coefficient plots

*DATAPLOT Interactive System*

**BOX-COX NORMALITY PLOT** Plots probability plot correlation coefficient vs. Box-Cox transformation parameter $\lambda$ ($\lambda$ ranging from -2 to +2 or in user-set range) – for determining the "best" transformation from the Box-Cox family to normalize data which are commonly neither normal nor symmetric.

## L4a1a3p : Pareto and Poisson probability plot correlation coefficient plots

*DATAPLOT Interactive System*

**PARETO PPCC PLOT** Generates a probability plot correlation coefficient plot for the Pareto distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging from 1 to 100 or in user-set range).

**POISSON PPCC PLOT** Generates a probability plot correlation coefficient plot for the Poisson distribution (plot of probability plot correlation coefficient vs. the tail parameter $\lambda$ for $\lambda$ ranging from 1 to 100 or in user-set range).

## L4a1a3t : t probability plot correlation coefficient plots

*DATAPLOT Interactive System*

**T PPCC PLOT** Generates a probability plot correlation coefficient plot for the t distribution (plot of probability plot correlation coefficient vs. degrees of freedom parameter $\nu$ for $\nu$ ranging from 1 to 100 or in user-set range).

**L4a1a3w : Weibull probability plot correlation coefficient plots**

*DATAPLOT Interactive System*

**WEIBULL PPCC PLOT** Generates a probability plot correlation coefficient plot for the Weibull distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging from 1 to 100 or in user-set range).

**L4a1a4b : Binomial distribution parameter estimates and hypothesis tests**

*IMSL Subprogram Library*

**BELBIN** Interval estimate of the parameter p of the binomial distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**BINES** Estimate the parameter p of the binomial distribution.

**L4a1a4e : Extreme value distribution parameter estimates and hypothesis tests**

*DATAPAC Subprogram Library*

**EXTREM** Performs an extreme value analysis on the data in the input vector X.

**L4a1a4n : Normal distribution parameter estimates and hypothesis tests**

*BMDP Program Library*

**P3D** One-sample t-test to test if one group mean is zero (e.g., matched pairs); two-sample t test with and without equal variances assumption, Levene's test for equal variances, histograms. Options: trimmed t test, Hotelling's T-squared and Mahalanobis' D-squared, within-group correlations, data listing.

*DATAPLOT Interactive System*

**CONFIDENCE LIMITS** Generates confidence limits for the mean, based on normal theory, for a series of confidence levels.

**T-TEST** Performs a two sample t-test for the mean based on normal theory. The analysis is performed with and without equal variances assumption. The sample sizes do not have to be equal.

*IMSL Subprogram Library*

**BEMNON** Location (mean) inferences using a sample from a normal population with known variance.

**BEMSON** Mean and variance inferences using a sample from a normal population.

**BENSON** Variance inferences using a sample from a normal population with known mean.

**GTNOR** Test for normality of random deviates.

*MINITAB Interactive System*

**TINTERVAL** Calculates a t-confidence interval with specified percent confidence.

**TTEST** Performs one- or two-sided t-tests.

**ZINTERVAL** Calculates a z-confidence interval with specified percent confidence and standard deviation.

**ZTEST** Performs a one- or two-sided z-test for a specified standard deviation.

*NAG Subprogram Library*

**G01DDF** Calculates Shapiro and Wilk's W statistic and its significance level for testing Normality.

*STARPAC Subprogram Library*

• **STAT** Compute and print 53 statistics describing the input data.

**STATS** Compute and optionally print 53 statistics describing input data; return statistics.

**STATW** Compute and print 53 statistics describing weighted input data.

**STATWS** Compute and optionally print 53 statistics describing weighted input data; return statistics.

## L4a1a4p : Poisson distribution parameter estimates and hypothesis tests

*IMSL Subprogram Library*

**BELPOS** Interval estimate of the parameter lambda of the Poisson distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**POIES** Estimate the parameter of the Poisson distribution.

## L4a1a4u : Uniform distribution parameter estimates and hypothesis tests

*IMSL Subprogram Library*

**GTMNT** Moments and standardized moments of uniform random numbers.

*Scientific Desk PC Subprogram Library*

**L4A1BU** Tests for uniformity of deviates when samples are small.

**L4A1D** Computes the first four moments about the mean for the sample contained in x, and performs a goodness of fit test for uniformity.

## L4a1a4w : Weibull distribution parameter estimates and hypothesis tests

*DATAPAC Subprogram Library*

**WEIB** Performs a Weibull distribution analysis on the data in the input vector X.

| L4a1a5 : | **Transformation selection (e.g., for normality)** |
|---|---|

*DATAPLOT Interactive System*

**BOX-COX LINEARITY PLOT** Generate a correlation plot for the Box-Cox family of transformations. The horizontal axis is the $\lambda$ parameter for the Box-Cox family of transformations and the vertical axis is the correlation coefficient between the transformed variables.

**BOX-COX NORMALITY PLOT** Plots probability plot correlation coefficient vs. Box-Cox transformation parameter $\lambda$ ($\lambda$ ranging from -2 to +2 or in user-set range) – for determining the "best" transformation from the Box-Cox family to normalize data which are commonly neither normal nor symmetric.

| L4a1a6 : | **Tail and outlier analysis** |
|---|---|

*DATAPAC Subprogram Library*

**NOROUT** Performs a normal outlier analysis on the data in the input vector X.

**TAIL** Performs a symmetric distribution tail length analysis on the data in the input vector X.

| L4a1a7 : | **Tolerance limits** |
|---|---|

*DATAPAC Subprogram Library*

**TOL** Computes normal and distribution-free tolerance limits for the data in the input vector X.

| L4a1b1 : | **Nonparametric estimates and tests regarding location (e.g., median), dispersion, and shape** |
|---|---|

*BMDP Program Library*

**P3S** Computes and prints results from one or more of the following: sign test, Wilcoxon signed-rank test, Mann-Whitney rank-sum test, Kruskall-Wallis one-way ANOVA, Friedman two-way ANOVA, Kendall's coefficient of concordance, Kendall and Spearman rank-correlation coefficients.

*DATAPLOT Interactive System*

**BOOTSTRAP PLOT** Generate a bootstrap plot for a given statistic, where the statistic is selected from among 29 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics or is user defined. The generated bootstrap samples are available for further analysis.

**JACKNIFE PLOT** Generate a jacknife plot for a given statistic, where the statistic is selected from among 29 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics or is user defined. The generated jacknife samples are available for further analysis.

*IMSL Subprogram Library*

**NBSIGN** Sign test (for percentiles).

*IMSL STAT/LIBRARY Subprogram Library*

**SIGNT** Perform a sign test of the hypothesis that a given value is a specified quantile of a distribution.

**SNRNK** Perform a Wilcoxon signed rank test.

*MINITAB Interactive System*

**WINTERVAL** Calculates a one-sample Wilcoxon rank estimate and confidence interval for the center of a symmetric distribution.

**WTEST** Performs one-sample one- or two-sided Wilcoxon signed-rank tests.

## L4a1b2 : Nonparameteric density function estimation

*IMSL Subprogram Library*

**NDKER** Nonparametric probability density function (one dimensional) estimation by the kernel method.

**NDMPLE** Nonparametric probability density function (one dimensional) estimation by the penalized likelihood method.

*IMSL STAT/LIBRARY Subprogram Library*

**DESKN** Perform nonparametric probability density function estimation by the kernel method.

**DESPL** Perform nonparametric probability density function estimation by the penalized likelihood method.

**DESPT** Estimate a probability density function at specified points using linear or cubic interpolation.

## L4a1c : Goodness-of-fit tests

*IMSL Subprogram Library*

**GFIT** Chi-squared goodness of fit test.

**GTCN** Sample size or number of class intervals determination for chi-squared test applications.

**NKS1** Kolmogorov-Smirnov one-sample test.

*IMSL STAT/LIBRARY Subprogram Library*

**CHIGF** Perform a chi-squared goodness of fit test.

**KSONE** Perform a Kolmogorov-Smirnov one-sample test for continuous distributions.

*NAG Subprogram Library*

**G08CAF** Performs the one sample Kolmogorov-Smirnov distribution test.

*Scientific Desk PC Subprogram Library*

**L4A1CC** Performs a chi-squared goodness of fit test that given data are from a hypothesized density function.

**L4A1CD** Performs a chi-squared goodness of fit test that given data fall into categories in a hypothesized ratio.

**L4A1M** Uses the Mann-Wald criterion to determine the number of categories, or the sample size, for performance of a subsequent chi-squared test.

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

**L4a1d : Analysis of a sequence of numbers** *(search also class L10a)*

*DATAPAC Subprogram Library*

**RUNS** Performs a run analysis of the data in the input vector X.

*DATAPLOT Interactive System*

**CONTROL CHART** Plots a specified statistic (mean (default), standard deviation, or range) for replicated data as a function of time or replication group number. Output contains computed statistics, a typical value line and upper and lower control line.

**RUNS** Perform a runs analysis. Generated output includes the number of runs up, the number of runs down, the number of runs up and down, and the total number of runs. A z score is generated for the number of runs equal to I for I equal 1 through 10.

*IMSL Subprogram Library*

**GTD2T** The d-square test.

**GTPST** Pairs test or Goods serial test.

**GTRN** Runs test.

**GTTT** Triplets test.

**NBCYC** Noethers test for cyclical trend.

**NBSDL** Cox and Stuart sign test for trends in dispersion and location.

*IMSL STAT/LIBRARY Subprogram Library*

**DCUBE** Perform a triplets test.

**DSQAR** Perform a D-square test.

**NCTRD** Perform the Noether test for cyclical trend.

**PAIRS** Perform a pairs test.

**RUNS** Perform a runs up test.

**SDPLC** Perform the Cox and Stuart sign test for trends in dispersion and location.

*MINITAB Interactive System*

**RUNS** Performs a two-sided runs test.

*Scientific Desk PC Subprogram Library*

**L4A1DP** Performs the pairs or serial test for uniformity of random deviates by tallying counts of coordinate occurrences and then performing a chi-squared test on the tally matrix.

**L4A1DT** Performs the triplets test for uniformity of random deviates by tallying counts of coordinate occurrences and then performing a chi-squared test on the tally matrix.

**L4B1BR** Performs the runs-up and runs-down test for uniformity of deviates.

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

*STARPAC Subprogram Library*

• **STAT** Compute and print 53 statistics describing the input data.

**STATS** Compute and optionally print 53 statistics describing input data; return statistics.

**STATW** Compute and print 53 statistics describing weighted input data.

**STATWS** Compute and optionally print 53 statistics describing weighted input data; return statistics.

• **easy-to-use version**

## L4a3 : Analysis of grouped and/or censored data

*IMSL Subprogram Library*

**OTMLNR** Maximum likelihood estimation from grouped and/or censored normal data.

*IMSL STAT/LIBRARY Subprogram Library*

**NRCES** Compute maximum likelihood estimates of the mean and variance from grouped and/or censored normal data.

## L4a4 : Analysis of data sampled from a finite population

*IMSL Subprogram Library*

**SSPAND** Simple random sampling with proportion data – inferences regarding the population proportion and total.

**SSPBLK** Stratified random sampling with proportion data – inferences regarding the population proportion and total.

**SSSAND** Simple random sampling with continuous data – inferences regarding the population mean and total.

**SSSBLK** Stratified random sampling with continuous data – inferences regarding the population mean and total.

**SSSCAN** Single stage cluster sampling with continuous data – inferences regarding the population mean and total.

**SSSEST** Two-stage sampling with continuous data and equisized primary units – inferences regarding the population mean and total.

*IMSL STAT/LIBRARY Subprogram Library*

**SMPPR** Compute statistics for inferences regarding the population proportion and total given proportion data from a simple random sample.

**SMPPS** Compute statistics for inferences regarding the population proportion and total given proportion data from a stratified random sample.

**SMPSC** Compute statistics for inferences regarding the population mean and total using single stage cluster sampling with continuous data.

**SMPSR** Compute statistics for inferences regarding the population mean and total, given data from a simple random sample.

**SMPSS** Compute statistics for inferences regarding the population mean and total, given data from a stratified random sample.

**SMPST** Compute statistics for inferences regarding the population mean and total given continuous data from a two-stage sample with equisized primary units.

## L4a5 : Analysis of one-dimensional categorical data

*IMSL Subprogram Library*

**GTPOK** The poker test.

**L4b1a1 : Plots of empirical and theoretical density and distribution functions for two pairwise independent data sets**

*DATAPLOT Interactive System*

**QUANTILE-QUANTILE PLOT** Generate a quantile-quantile plot with horizontal axis the estimated quantiles from data set 1 and vertical axis the estimated quantiles from data set 2.

*IMSL STAT/LIBRARY Subprogram Library*

**CDF2P** Print a plot of two sample cumulative distribution functions.

**L4b1a4 : Parameter estimates and hypothesis tests for two pairwise independent data sets**

*BMDP Program Library*

**P3D** One-sample t-test to test if one group mean is zero (e.g., matched pairs); two-sample t test with and without equal variances assumption, Levene's test for equal variances, histograms. Options: trimmed t test, Hotelling's T-squared and Mahalanobis' D-squared, within-group correlations, data listing.

*DATAPLOT Interactive System*

**T-TEST** Performs a two sample t-test for the mean based on normal theory. The analysis is performed with and without equal variances assumption. The sample sizes do not have to be equal.

*IMSL Subprogram Library*

**BEPAT** Mean and variance inferences using samples from each of two normal populations with unequal variances.

**BEPET** Mean and variance inferences using samples from each of two normal populations with equal variances.

*IMSL STAT/LIBRARY Subprogram Library*

**TWOMV** Compute statistics for mean and variance inferences using samples from two normal populations.

*MINITAB Interactive System*

**TWOSAMPLE** Performs a one- or two-sided two-sample t-test of equality of two population means with variance estimated either from each sample or the pooled sample.

**TWOT** Performs a one- or two-sided two-sample t-test of equality of two population means with variance estimated either from each sample or the pooled sample.

*SAS Program Library*

**TTEST** Computes a t statistic for testing the hypothesis that the means of two groups of observations are equal.

*Scientific Desk PC Subprogram Library*

**L4B1** Performs two sample tests on the mean of samples from normal populations, and estimates the means and variances of the samples.

*SPSS Program Library*

**T-TEST** Produces Student's t-statistic, degrees of freedom, and two-tailed probability for a comparison of two means for independent samples (using pooled- and separate -variance estimates with a test of homogeneity of variances) or for paired samples. Option: missing values.

**L4b1b : Nonparametric analysis of two pairwise independent data sets**

*BMDP Program Library*

**P3S** Computes and prints results from one or more of the following: sign test, Wilcoxon signed-rank test, Mann-Whitney rank-sum test, Kruskall-Wallis one-way ANOVA, Friedman two-way ANOVA, Kendall's coefficient of concordance, Kendall and Spearman rank-correlation coefficients.

*Collected Algorithms of the ACM*

**A516** RANKCI: a Fortran subroutine for obtaining confidence intervals and point estimates based on ranks in the two-sample location problem. (See J.W. McKean and T.A. Ryan, Jr., ACM TOMS 3 (1977) pp. 183-185.)

*DATAPLOT Interactive System*

**BOOTSTRAP PLOT** Generate a bootstrap plot for a given statistic, where the statistic is selected from among 29 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics or is user defined. The generated bootstrap samples are available for further analysis.

**JACKNIFE PLOT** Generate a jacknife plot for a given statistic, where the statistic is selected from among 29 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics or is user defined. The generated jacknife samples are available for further analysis.

*IMSL Subprogram Library*

**NHINC** Includance test.

**NMKN** Kendall's test for correlation (rank correlation coefficient).

**NRWRST** Wilcoxon's rank-sum test.

*IMSL STAT/LIBRARY Subprogram Library*

**CNCRD** Calculate and test the significance of the Kendall coefficient of concordance.

**INCLD** Perform an includance test.

**KENDL** Compute and test Kendall's rank correlation coefficient.

**RNKSM** Perform the Wilcoxon rank sum test.

*MINITAB Interactive System*

**MANN-WHITNEY** Performs one- or two-sided two-sample rank test (a.k.a. Wilcoxon rank test) for the difference between two population medians, and calculates the corresponding point and confidence interval estimates.

*NAG Subprogram Library*

**G08ACF** Performs the median test on two independent samples of possibly unequal size.

**G08ADF** Performs the Mann-Whitney U test on two independent samples of possibly unequal size.

**G08BAF** Performs Mood's and David's tests for dispersion differences between two independent samples of possibly unequal size.

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

| **L4b1c :** | **Goodness-of-fit tests for two pairwise independent data sets** |
|---|---|

*IMSL Subprogram Library*

**NKS2** Kolmogorov-Smirnov two-sample test.

*IMSL STAT/LIBRARY Subprogram Library*

**KSTWO** Perform a Kolmogorov-Smirnov two-sample test.

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

| **L4b3 :** | **Analysis of two pairwise dependent data sets** |
|---|---|

*BMDP Program Library*

**P3S** Computes and prints results from one or more of the following: sign test, Wilcoxon signed-rank test, Mann-Whitney rank-sum test, Kruskall-Wallis one-way ANOVA, Friedman two-way ANOVA, Kendall's coefficient of concordance, Kendall and Spearman rank-correlation coefficients.

*IMSL Subprogram Library*

**NRWMD** Wilcoxon signed rank test.

*NAG Subprogram Library*

**G08AAF** Performs the Sign test on two related samples of size N.

**G08ABF** Performs the Wilcoxon matched pairs signed ranks test on two related samples of size N.

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

| **L4b4 :** | **Analysis of two pairwise dependent grouped data sets** |
|---|---|

*IMSL Subprogram Library*

**BECTR** Tetrachoric correlation coefficient estimation.

**BESRB** Biserial and point-biserial correlation coefficients for a qualitatively dichotomized variable and a numerically measurable and classified variable.

**BESRN** Biserial correlation coefficient for a qualitatively dichotomized variable and a numerically or qualitatively classified variable.

**CBNRHO** Estimation of the bivariate normal correlation coefficient using a contingency table.

*IMSL STAT/LIBRARY Subprogram Library*

**BSCAT** Compute the biserial correlation coefficient for a dichotomous variable and a classification variable.

**BSPBS** Compute the biserial and point-biserial correlation coefficients for a dichotomous variable and a numerically measurable classification variable.

**CTRHO** Estimate the bivariate normal correlation coefficient using a contingency table.

**TETCC** Categorize bivariate data and compute the tetrachoric correlation coefficient.

## L4b5 : Analysis of two-dimensional data sampled from a finite population

*IMSL Subprogram Library*

**SSRAND** Simple random sampling with continuous data – inferences regarding the population mean and total using ratio or regression estimation.

**SSRBLK** Stratified random sampling with continuous data-inferences regarding the population mean and total using ratio or regression estimation.

*IMSL STAT/LIBRARY Subprogram Library*

**SMPRR** Compute statistics for inferences regarding the population mean and total using ratio or regression estimation, or inferences regarding the population ratio given a simple random sample.

**SMPRS** Compute statistics for inferences regarding the population mean and total using ratio or regression estimation given continuous data from a stratified random sample.

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

## L4c1a : Parametric analysis of multiple independent data sets

*BMDP Program Library*

**P3D** One-sample t-test to test if one group mean is zero (e.g., matched pairs); two-sample t test with and without equal variances assumption, Levene's test for equal variances, histograms. Options: trimmed t test, Hotelling's T-squared and Mahalanobis' D-squared, within-group correlations, data listing.

*IMSL Subprogram Library*

**BESTA2** Computations of confidence intervals and other basic statistics using output from IMSL routine BESTAT.

*IMSL STAT/LIBRARY Subprogram Library*

**UVSTA** Compute basic univariate statistics.

*MINITAB Interactive System*

**TINTERVAL** Calculates a t-confidence interval with specified percent confidence.

**TTEST** Performs one- or two-sided t-tests.

*SPSS Program Library*

**PEARSON CORR** Produces matrices of Pearson product-moment correlation coefficients with one- or two-tailed significance levels. Options: univariate statistics, covariances, cross-product deviations, missing values.

*STARPAC Subprogram Library*

**CORR** Compute and print a correlation analysis of a multivariate random sample.

**CORRS** Compute and optionally print a correlation analysis of a multivariate random sample; return variance-covariance matrix.

## L4c1b : Nonparametric analysis of multiple independent data sets

*BMDP Program Library*

**P3S** Computes and prints results from one or more of the following: sign test, Wilcoxon signed-rank test, Mann-Whitney rank-sum test, Kruskall-Wallis one-way ANOVA, Friedman two-way ANOVA, Kendall's coefficient of concordance, Kendall and Spearman rank-correlation coefficients.

*IMSL Subprogram Library*

**NAK1** Kruskal-Wallis test for identical populations.

**NBQT** Cochran q test.

**NMCC** Calculate and test the significance of the Kendall coefficient of concordance.

**NMKTS** K-sample trends test against ordered alternatives.

**NRBHA** Bhapkar v test.

*IMSL STAT/LIBRARY Subprogram Library*

**BHAKV** Perform a Bhapkar V test.

**KRSKL** Perform a Kruskal-Wallis test for identical population medians.

**KTRND** Perform K-sample trends test against ordered alternatives.

**QTEST** Perform a Cochran Q test for related observations.

*MINITAB Interactive System*

**KRUSKAL-WALLIS** Perform Kruskal-Wallis test, based on ranks, of the null hypothesis that there is no difference among K population locations against the alternative of at least one difference. (This is a K-sample generalization of the Mann-Whitney-Wilcoxon test and is a nonparametric alternative to one-way ANOVA.)

**WINTERVAL** Calculates a one-sample Wilcoxon rank estimate and confidence interval for the center of a symmetric distribution.

**WTEST** Performs one-sample one- or two-sided Wilcoxon signed-rank tests.

*NAG Subprogram Library*

**G08DAF** Calculates Kendall's coefficient of concordance on k independent rankings of n objects or individuals.

*SPSS Program Library*

**NONPAR CORR** Produces a matrix of either Spearman's rho or Kendall's tau-b rank-order correlation coefficients with one- or two-tailed significance levels. Options: random sampling, missing values.

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

| L4e : | Analysis of multiple multi-dimensional data sets |
|---|---|

*IMSL Subprogram Library*

**OIND** Wilks test for the independence of k sets of multi-normal variates.

*IMSL STAT/LIBRARY Subprogram Library*

**MVIND** Compute a test for the independence of K sets of multivariate normal variables.

## L5: Function Evaluation

The evaluation of statistical functions and their inverses is a major part of special function evaluation. A discussion of computational issues in the evaluation of these functions can be found in the introduction to class C (Elementary and Special Functions).

The software documented in this class is first organized by the dimension of the distribution — univariate (i.e., one-dimensional) or multivariate (i.e., multi-dimensional). The next level of organization distinguishes the evaluation of (cumulative) distribution functions and (probability) density functions from the evaluation of percent point functions (the inverses of distribution functions) and sparsity functions (the inverses of density functions).

### References

[JK69] N. I. Johnson and S. Kotz. *Distributions in Statistics.* Houghton Mifflin Company, Boston, 1969. (four volumes).

[KG80] W. J. Kennedy and J. E. Gentle. *Statistical Computing.* Marcel Dekker, New York, 1980.

| L5a1b : | Beta and binomial distribution and density functions |
|---|---|

*DATAPAC Subprogram Library*

**BINCDF** Computes the cumulative distribution function value at X for the binomial distribution with parameters P and N.

*IMSL Subprogram Library*

**GTPKP** Probability distribution of n elements into two equi-probable states.

**MDBETA** Beta probability distribution function.

**MDBIN** Binomial probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**BETDF** Beta probability distribution function.

**BINDF** Evaluate the binomial distribution function.

**BINPR** Evaluate the binomial probability function.

*MINITAB Interactive System*

**BINOMIAL** Prints table of binomial probabilities and cumulative distribution function, and optionally saves results.

*NAG Subprogram Library*

**G01BDF** Returns the probability, I(x)(a,b) associated with the lower tail of the Beta distribution of the first kind with parameters a and b, through the routine name.

*Scientific Desk PC Subprogram Library*

**L5A1B** Incomplete beta cumulative probability distribution function.

## L5a1c : Cauchy and chi-squared distribution and density functions

*DATAPAC Subprogram Library*

**CAUCDF** Computes the cumulative distribution function value for the Cauchy distribution with median 0 and 75% point 1.

**CAUPDF** Computes the probability density function value for the Cauchy distribution with median 0 and 75% point 1.

**CHSCDF** Computes the cumulative distribution function value for the chi-squared distribution with degrees of freedom parameter $\nu$.

*IMSL Subprogram Library*

**MDCH** Chi-squared probability distribution function.

**MDCHN** Non-central chi-squared probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**CHIDF** Chi-squared distribution function.

**CSNDF** Evaluate the noncentral chi-squared distribution function.

*NAG Subprogram Library*

**G01BCF** Returns the probability, P(x,n), associated with the upper tail of the chi-square distribution with n degrees of freedom, through the function name.

*Scientific Desk PC Subprogram Library*

**L5A1** Calculates the probability that a chi-squared distributed random variable (f degrees of freedom) is greater than a statistic x.

## L5a1d : Double exponential distribution and density functions

*DATAPAC Subprogram Library*

**DEXCDF** Computes the cumulative distribution function value for the double exponential (Laplace) distribution with mean 0.

**DEXPDF** Computes the probability density function value for the double exponential (Laplace) distribution with mean 0.

## L5a1e : Error function, exponential and extreme value distribution and density functions

*CMLIB Library (FNLIB Sublibrary)*

**ERF** Error function, $= (2/\sqrt{\pi}) \times$ the integral from 0 to $x$ of $\exp(-t^2)\,dt$.

**ERFC** Complementary error function, $= (2/\sqrt{\pi}) \times$ the integral from $x$ to $\infty$ of $\exp(-t^2)\,dt$.

*DATAPAC Subprogram Library*

**EV1CDF** Computes the cumulative distribution function value for the extreme value type 1 distribution.

**EV2CDF** Computes the cumulative distribution function value for the extreme value type 2 distribution with tail length parameter $\gamma$.

**EXPCDF** Computes the cumulative distribution function value for the exponential distribution with mean 1 and standard deviation 1.

**EXPPDF** Computes the probability density function value for the exponential distribution with mean 1 and standard deviation 1.

*IMSL, IMSL SFUN/LIBRARY, and NMS Subprogram Libraries*

**ERF** Error function, $= (2/\sqrt{\pi}) \times$ the integral from 0 to $x$ of $\exp(-t^2)\,dt$.

**ERFC** Complementary error function, $= (2/\sqrt{\pi}) \times$ the integral from $x$ to $\infty$ of $\exp(-t^2)\,dt$.

*NAG Subprogram Library*

**S15ADF** Complement of error function, erfc(x).

**S15AEF** Error function, erf(x).

## L5a1f : F distribution and density functions

*DATAPAC Subprogram Library*

**FCDF** Computes the cumulative distribution function value for the F-distribution with degrees of freedom parameters $\nu_1$ and $\nu_2$.

*IMSL Subprogram Library*

**MDFD** F probability distribution function.

**MDFDRE** F probability distribution function (integer or fractional degrees of freedom).

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**FDF** Evaluate the F distribution function.

*NAG Subprogram Library*

**G01BBF** Returns the probability associated with the upper tail of the F or variance-ratio distribution with m and n degrees of freedom, through the routine name.

*Scientific Desk PC Subprogram Library*

**L5A1F** Returns the probability that a random variable distributed as F with m and n degrees of freedom is greater than the statistic F.

## L5a1g : Gamma, general, and geometric distribution and density functions

*DATAPAC Subprogram Library*

**GAMCDF** Computes the cumulative distribution function value for the gamma distribution with tail length parameter $\gamma$.

**GEOCDF** Computes the geometric cumulative distribution function value at the value X with parameter P.

*IMSL Subprogram Library*

**MDGAM** Gamma probability distribution function.

**MDGC** General cumulative probability distribution function, given ordinates of the density.

**NDEST** Evaluate probability density function at specified points.

*IMSL STAT/LIBRARY and SFUN/LIBRARY Subprogram Libraries*

**GAMDF** Gamma distribution function.

**GCDF** Evaluate a general continuous cumulative distribution function given ordinates of the density.

| L5a1h : | Halfnormal and hypergeometric distribution and density functions |
|---|---|

*DATAPAC Subprogram Library*

**HFNCDF** Computes the cumulative distribution function value for the halfnormal distribution with mean $\sqrt{2/\pi}$ and standard deviation 1.

*IMSL Subprogram Library*

**MDHYP** Hypergeometric probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**HYPDF** Evaluate the hypergeometric distribution function.

**HYPPR** Evaluate the hypergeometric probability function.

*Scientific Desk PC Subprogram Library*

**L5A1H** Calculates cumulative hypergeometric probabilities (the probability of obtaining l marked items in a random sample of size k taken without replacement from a population containing m items, n of which are marked).

| L5a1k : | Kendall F statistic and Kolmogorov-Smirnov distribution and density functions |
|---|---|

*IMSL Subprogram Library*

**MDSMR** Kolmogorov-Smirnov statistics asymptotic probability distribution function.

**NMKSF** Frequency distribution of K and the probability of equalling or exceeding K, where K, the total score from the Kendall rank correlation coefficient calculations, and N, the sample size, are given.

*IMSL STAT/LIBRARY Subprogram Library*

**AKS1DF** Evaluate the distribution function of the one-sided Kolmogorov-Smirnov goodness of fit D+ or D- test statistic based on continuous data for one sample.

**AKS2DF** Evaluate the distribution function of the Kolmogorov-Smirnov goodness of fit D test statistic based on continuous data for two samples.

**KENDP** Compute the frequency distribution of the total score in Kendall's rank correlation coefficient.

*IMSL SFUN/LIBRARY Subprogram Library*

**AKS1DF** Evaluate the distribution function of the one-sided Kolmogorov-Smirnov goodness of fit D+ or D- test statistic based on continuous data for one sample.

**AKS2DF** Evaluate the distribution function of the Kolmogorov-Smirnov goodness of fit D test statistic based on continuous data for two samples.

## L5all : Lambda, logistic, and lognormal distribution and density functions

*DATAPAC Subprogram Library*

**LAMCDF** Computes the cumulative distribution function value for the (Tukey) lambda distribution with tail length parameter $\lambda$.

**LAMPDF** Computes the probability density function value for the (Tukey) lambda distribution with tail length parameter $\lambda$.

**LGNCDF** Computes the cumulative distribution function value for the lognormal distribution with mean $\sqrt{e}$.

**LOGCDF** Computes the cumulative distribution function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$.

**LOGPDF** Computes the probability density function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$.

*Scientific Desk PC Subprogram Library*

**L5A1L** Computes the natural logarithm of the normal distribution function (that is, the natural log of the integral from -infinity to x of the standard normal density).

## L5aln : Negative binomial and normal distribution and density functions

*DATAPAC Subprogram Library*

**NBCDF** Computes the cumulative distribution function value at X for the negative binomial distribution with parameters P and N.

**NORCDF** Computes the cumulative distribution function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1.

**NORPDF** Computes the probability density function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1.

*IMSL Subprogram Library*

**MDNOR** Normal or Gaussian probability distribution function.

**MSMRAT** Ratio of the ordinate to the upper tail area of the standardized normal (Gaussian) distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**AMILLR** Evaluate Mill's ratio (the ratio of the ordinate to the upper tail area of the standardized normal distribution).

**ANORDF** Normal (Gaussian) distribution function.

*IMSL SFUN/LIBRARY Subprogram Library*

**ANORDF** Normal (Gaussian) distribution function.

*NAG Subprogram Library*

**S15ABF** Cumulative normal distribution function.

**S15ACF** Complement of cumulative normal distribution function.

*Scientific Desk PC Subprogram Library*

**L5A1I** Calculates (1) the integral of the normal probability density function from -infinity to x, (2) the integral of the normal probability density function from x to infinity, and (3) the density of the normal probability function at x.

| **L5a1p :** | **Pareto and Poisson distribution and density functions** |
|---|---|

*DATAPAC Subprogram Library*

**PARCDF** Computes the cumulative distribution function value for the Pareto distribution with tail length parameter $\gamma$.

**POICDF** Computes the cumulative distribution function value at X for the Poisson distribution with tail length parameter $\lambda$.

*IMSL Subprogram Library*

**MDTPS** Cumulative probability and, optionally, individual terms of the Poisson probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**POIDF** Evaluate the Poisson distribution function.

**POIPR** Evaluate the Poisson probability function.

*MINITAB Interactive System*

**POISSON** Prints table of Poisson probabilities and cumulative distribution function.

| **L5a1t :** | **t distribution and density functions** |
|---|---|

*DATAPAC Subprogram Library*

**TCDF** Computes the cumulative distribution function value for Student's t distribution with degrees of freedom parameter $\nu$.

*IMSL Subprogram Library*

**MDTD** Student's t probability distribution function.

**MDTN** Non-central t probability distribution function.

**MDTNF** Integral related to calculation of non-central t and bivariate normal probability distribution functions.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**TDF** Evaluate the Student's t distribution function.

**TNDF** Evaluate the noncentral Student's t distribution function.

*NAG Subprogram Library*

**G01BAF** Returns the probability associated with the lower tail of the Student's t distribution with n degrees of freedom, through the routine name.

*Scientific Desk PC Subprogram Library*

**L5A1T** Computes the integral from t to positive infinity of Student's t distribution.

| **L5a1u :** | **Uniform distribution and density functions** |
|---|---|

*Collected Algorithms of the ACM*

**A519** RAKK, DURB, and EPST: Fortran subroutines for computing Kolmogorov-Smirnov probabilities with arbitrary boundaries. RAKK is a generalization of Massey's method. DURB is Durbin's method. EPST is the Epanechnikov, Steck method. (See R. Kallman, ACM TOMS 3 (1977) pp. 285-294.)

*DATAPAC Subprogram Library*

**UNICDF** Computes the cumulative distribution function value for the uniform (rectangular) distribution on the unit interval (0,1).

**UNIPDF** Computes the probability density function value for the uniform (rectangular) distribution on the unit interval (0,1).

## L5a1v : Von Mises distribution and density functions

*Collected Algorithms of the ACM*

**A518** VMISES: a Fortran function which computes the left tail area of the Von Mises distribution, which is equal to the incomplete modified Bessel function of the first kind and zero-th order (I0). (see G.W. Hill, ACM TOMS 3 (1977)pp. 279-284.)

**A571** BESRAT, VKAPPA, SPHERR, CAPPA3: Fortran functions providing statistics for von Mises's and Fisher's distributions of directions (the ratio of modified Bessel functions of the first kind). (See G.W. Hill, ACM TOMS 7 (1981) pp. 233-238.)

## L5a1w : Weibull distribution and density functions

*DATAPAC Subprogram Library*

**WEICDF** Computes the cumulative distribution function value for the Weibull distribution with tail length parameter $\gamma$.

## L5a2b : Inverse beta and binomial distribution and sparsity functions

*DATAPAC Subprogram Library*

**BINPPF** Computes the percent point function value at P for the binomial distribution with parameters PPAR and N.

*IMSL Subprogram Library*

**MDBETI** Inverse beta probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**BETIN** Inverse of the beta distribution function.

*NAG Subprogram Library*

**G01CDF** Returns the deviate, x(p), associated with the given lower tail probability p of the Beta distribution of the first kind with parameters a and b, through the function name.

## L5a2c : Inverse Cauchy and chi-squared distribution and sparsity functions

*DATAPAC Subprogram Library*

**CAUPPF** Computes the percent point function value for the Cauchy distribution with median 0 and 75% point 1.

**CAUSF** Computes the sparsity function value for the Cauchy distribution with median 0 and 75% point 1.

**CHSPPF** Computes the percent point function value for the chi-squared distribution with integer degrees of freedom parameter $\nu$.

*IMSL Subprogram Library*

**MDCHI** Inverse chi-squared probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**CHIIN** Inverse of the chi-squared distribution function.

*NAG Subprogram Library*

**G01CCF** Returns the deviate, x(p), associated with the given lower tail probability p of the chi-square distribution with n degrees of freedom, through the function name.

*Scientific Desk PC Subprogram Library*

**L5A2C** Computes the inverse of the chi-squared distribution function.

## L5a2d : Inverse double exponential distribution and sparsity functions

*DATAPAC Subprogram Library*

**DEXPPF** Computes the percent point function value for the double exponential (Laplace) distribution with mean 0.

**DEXSF** Computes the sparsity function value for the double exponential (Laplace) distribution with mean 0 and standard deviation $\sqrt{2}$.

## L5a2e : Inverse error function, inverse exponential and extreme value distribution and sparsity functions

*DATAPAC Subprogram Library*

**EV1PPF** Computes the percent point function value for the extreme value type 1 distribution with mean Euler's number 0.57721566.

**EV2PPF** Computes the percent point function value for the extreme value type 2 distribution with tail length parameter $\gamma$.

**EXPPPF** Computes the percent point function value for the exponential distribution with mean 1 and standard deviation 1.

**EXPSF** Computes the sparsity function value for the exponential distribution with mean 1 and standard deviation 1.

*IMSL Subprogram Library*

**MERFCI** Inverse complementary error function.

**MERFI** Inverse error function.

## L5a2f : Inverse F distribution and sparsity functions

*IMSL Subprogram Library*

**MDFI** Inverse F probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**FIN** Evaluate the inverse of the F distribution function.

*NAG Subprogram Library*

**G01CBF** Returns the deviate, x(p) associated with the lower tail probability p of the F variance-ratio distribution with m and n degrees of freedom, through the function name.

## L5a2g : Inverse gamma, general, and geometric distribution and sparsity functions

*DATAPAC Subprogram Library*

**GAMPPF** Computes the percent point function value for the gamma distribution with mean $\gamma$ and standard deviation $\sqrt{\gamma}$.

**GEOPPF** Computes the percent point function value for the geometric distribution with parameter PPAR.

*IMSL Subprogram Library*

**MDGCI** Inverse of a general cumulative probability distribution function, given ordinates of the density.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**GCIN** Evaluate the inverse of a general continuous cumulative distribution function given ordinates of the density.

## L5a2h : Inverse halfnormal distribution and sparsity functions

*DATAPAC Subprogram Library*

**HFNPPF** Computes the percent point function value for the halfnormal distribution with mean $\sqrt{2/\pi}$ and standard deviation 1.

## L5a2l : Inverse lambda, logistic, and lognormal distribution and sparsity functions

*DATAPAC Subprogram Library*

**LAMPPF** Computes the percent point function value for the (Tukey) lambda distribution with tail length parameter $\lambda$.

**LAMSF** Computes the sparsity function value for the (Tukey) lambda distribution with tail length parameter $\lambda$.

**LGNPPF** Computes the percent point function value for the lognormal distribution with mean $\sqrt{e}$.

**LOGPPF** Computes the percent point function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$.

**LOGSF** Computes the sparsity function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$.

## L5a2n : Inverse negative binomial and normal distribution and sparsity functions, normal order statistics

*DATAPAC Subprogram Library*

**NBPPF** Computes the percent point function value at P for the negative binomial distribution with parameters PPAR and N.

**NORPPF** Computes the percent point function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1.

**NORSF** Computes the sparsity function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1.

*IMSL Subprogram Library*

**MDNRIS** Inverse standard normal (Gaussian) probability distribution function.

**MSENO** Expected values of normal order statistics.

*IMSL STAT/LIBRARY Subprogram Library*

**ANORIN** Inverse of the normal (Gaussian) distribution function.

**ENOS** Evaluate the expected value of a normal order statistic.

*IMSL SFUN/LIBRARY Subprogram Library*

**ANORIN** Inverse of the normal (Gaussian) distribution function.

*NAG Subprogram Library*

**G01CEF** Returns the deviate, x(p), associated with the given lower tail probability p of the standardised Normal distribution, through the function name.

**G01DAF** Computes a set of Normal scores, i.e. the expected values of an ordered set of independent observations from a Normal distribution with mean 0.0 and standard deviation 1.0.

**G01DBF** Calculates an approximation to the set of Normal Scores, i.e. the expected values of an ordered set of independent observations from a Normal distribution with mean 0.0 and standard deviation 1.0.

*Scientific Desk PC Subprogram Library*

**L4ARD** Approximates the expected values of normal order statistics.

**L5A2N** Calculates the inverse of the cumulative normal distribution function.

## L5a2p : Inverse Pareto and Poisson distribution and sparsity functions

*DATAPAC Subprogram Library*

**PARPPF** Computes the percent point function value for the Pareto distribution with tail length parameter $\gamma$.

**POIPPF** Computes the percent point function value at P for the Poisson distribution with mean $\lambda$ and standard deviation $\sqrt{\lambda}$.

## L5a2t : Inverse t distribution and sparsity functions

*DATAPAC Subprogram Library*

**TPPF** Computes the percent point function value for the Student's t distribution with degrees of freedom parameter $\nu$.

*IMSL Subprogram Library*

**MDSTI** Inverse of a modification of Student's t probability distribution function.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**TIN** Evaluate the inverse of the Student's t distribution function.

*NAG Subprogram Library*

**G01CAF** Returns the deviate, x(p) associated with the given lower tail probability p of Student's t distribution with n degrees of freedom, through the function name.

**L5a2u : Inverse uniform distribution and sparsity functions**

*DATAPAC Subprogram Library*

**UNIPPF** Computes the percent point function value for the uniform (rectangular) distribution on the unit interval (0,1).

**UNISF** Computes the sparsity function value for the uniform (rectangular) distribution on the unit interval (0,1).

**L5a2w : Inverse Weibull distribution and sparsity functions**

*DATAPAC Subprogram Library*

**WEIPPF** Computes the percent point function value for the Weibull distribution with tail length parameter $\gamma$.

**L5b1n : Multivariate normal distribution and density functions**

*IMSL Subprogram Library*

**MDBNOR** Bivariate normal probability distribution function.

**MDTNF** Integral related to calculation of non-central t and bivariate normal probability distribution functions.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**BNRDF** Evaluate the bivariate normal distribution function.

**L5b2n : Inverse multivariate normal distribution function**

*NAG Subprogram Library*

**G01DCF** Computes an approximation to the variance-covariance matrix of an ordered set of independent observations from a Normal distribution with mean 0.0 and standard deviation 1.0.

## L6: Random Number Generation

The software documented in this class is first organized by the dimension of the distribution — univariate (i.e., one-dimensional) or multivariate (i.e., multi-dimensional), followed by service routines (for example, to set or retrieve the value of the seed).

Efficient methods have been developed for generating random numbers from the uniform and normal distributions. Random numbers from some other distributions can be obtained by transforming uniform numbers. Special methods have been developed for still other distributions.

In some applications, random numbers are paired or otherwise combined. For example, each sequential pair of uniform random numbers might be associated with a point in the unit square. Random number generators which produce correlated sequential pairs should not be used in such an application. New algorithms which don't exhibit such non-randomness have been developed.

Some random number generators produce acceptable — but different — results in more than one computer/compiler environment. Others produce exactly the same results in different environments. The slower execution time of these latter generators may be acceptable when different environments are necessary.

## References

[Dev86] L. Devroye. *Non-Uniform Random Variate Generation.* Springer-Verlag, New York, 1986.

[Knu81] D. E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming.* Addison-Wesley, Reading, MA, 2nd edition, 1981.

| **L6a2 :** | **Beta, binomial, and Boolean random numbers** |
|---|---|

*DATAPAC Subprogram Library*

**BETRAN** Generates a random sample of size N from the beta distribution with parameters $\alpha$ and $\beta$.

**BINRAN** Generates a random sample of size N from the binomial distribution with parameters P and NPAR.

*IMSL Subprogram Library*

**GGBN** Binomial random deviate generator.

**GGBTR** Beta random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNBET** Generate pseudorandom numbers from a beta distribution.

**RNBIN** Generate pseudorandom numbers from a binomial distribution.

*MINITAB Interactive System*

**BRANDOM** Generates K pseudo-random numbers from binomial distribution (number of successes in n Bernoulli trials with probability p of success).

**BTRIALS** Generates pseudo-random sequence of K 0's and 1's, with the probability p of a 1.

*NAG Subprogram Library*

**G05DLF** Returns a pseudo-random real number taken from a beta distribution of the first kind with parameters G and H.

**G05DMF** Returns a pseudo-random real number taken from a beta distribution of the second kind with parameters G and H.

**G05DZF** Returns a pseudo-random logical value – TRUE with probability P and FALSE with probability (1-P).

**G05EDF** Sets up the reference vector R for a binomial distribution of the number of successes in N trials, each with probability of success P.

*Scientific Desk PC Subprogram Library*

**L6A2** Generates one beta(p,q) deviate, using a rejection technique.

**L6A2B** Generates one binomial(n,p) deviate.

## L6a3 : Cauchy and chi-squared random numbers

*DATAPAC Subprogram Library*

**CAURAN** Generates a random sample of size N from the Cauchy distribution with median 0 and 75% point 1.

**CHSRAN** Generates a random sample of size N from the chi-squared distribution with integer degrees of freedom parameter $\nu$.

*IMSL Subprogram Library*

**GGCAY** Cauchy random deviate generator.

**GGCHS** Chi-squared random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNCHI** Generate pseudorandom numbers from a chi-squared distribution.

**RNCHY** Generate pseudorandom numbers from a Cauchy distribution.

*NAG Subprogram Library*

**G05DFF** Returns a pseudo-random real number taken from a Cauchy distribution with median A and semi-interquartile range B.

**G05DHF** Returns a pseudo-random real number taken from a chi-square distribution with N degrees of freedom.

*Scientific Desk PC Subprogram Library*

**L6A3** Generates one Cauchy random deviate.

**L6A3C** Returns one chi-squared deviate (n degrees of freedom).

## L6a4 : Double exponential random numbers

*DATAPAC Subprogram Library*

**DEXRAN** Generates a random sample of size N from the double exponential (Laplace) distribution with mean 0 and standard deviation $\sqrt{2}$.

## L6a5 : Exponential and extreme value random numbers

*Collected Algorithms of the ACM*

**A599** SEXPO, SGAMMA, SNORM, KPOISS, and SUNIF: Fortran functions which sample from the standard exponential, standard gamma, standard normal, Poisson, and uniform (0,1) distributions. (See J.H. Ahrens, K.D. Kohrt, and U. Dieter, ACM TOMS 9 (1983) pp. 255-257.)

*DATAPAC Subprogram Library*

**EV1RAN** Generates a random sample of size N from the extreme value type 1 distribution with mean Euler's number 0.57721566.

**EV2RAN** Generates a random sample of size N from the extreme value type 2 distribution with tail length parameter $\gamma$.

**EXPRAN** Generates a random sample of size N from the exponential distribution with mean 1 and standard deviation 1.

*IMSL Subprogram Library*

**GGEXN** Exponential random deviate generator.

**GGEXT** Random deviate generator for a mixture of two exponentials.

*IMSL STAT/LIBRARY Subprogram Library*

**RNEXP** Generate pseudorandom numbers from a standard exponential distribution.

**RNEXT** Generate pseudorandom numbers from a mixture of two exponential distributions.

*NAG Subprogram Library*

**G05DBF** Returns a pseudo-random real number taken from a (negative) exponential distribution with mean A.

*Scientific Desk PC Subprogram Library*

**L6A24** Generates two pseudo-random deviates from the error function density.

**L6A5** Generates one exponential deviate with mean and standard deviation u by inverting the distribution function.

## L6a6 : F distribution random numbers

*DATAPAC Subprogram Library*

**FRAN** Generates a random sample of size N from the F-distribution with degrees of freedom parameters $\nu_1$ and $\nu_2$.

*NAG Subprogram Library*

**G05DKF** Returns a pseudo-random real number taken from Snedecor's F (or Fisher's variance ratio) distribution with M and N degrees of freedom.

## L6a7 : Gamma, general (continuous and discrete), and geometric random numbers

*Collected Algorithms of the ACM*

**A599** SEXPO, SGAMMA, SNORM, KPOISS, and SUNIF: Fortran functions which sample from the standard exponential, standard gamma, standard normal, Poisson, and uniform (0,1) distributions. (See J.H. Ahrens, K.D. Kohrt, and U. Dieter, ACM TOMS 9 (1983) pp. 255-257.)

*DATAPAC Subprogram Library*

**GAMRAN** Generates a random sample of size N from the gamma distribution with tail length parameter $\gamma$, mean $\gamma$ and standard deviation $\sqrt{\gamma}$.

**GEORAN** Generates a random sample of size N from the geometric distribution with parameter P.

*IMSL Subprogram Library*

**GGAMR** One parameter gamma random deviate generator, and usable as basis for 2 parameter gamma, exponential, chi-squared, chi, beta, t and F deviate generator.

**GGDA** General discrete distribution random deviate generator using alias method.

**GGDT** General discrete distribution random deviate generator using table lookup method.

**GGEOT** Geometric random deviate generator.

**GGVCR** General continuous distribution random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNGAM** Generate pseudorandom numbers from a standard gamma distribution.

**RNGCS** Set up table to generate pseudorandom numbers from a general continuous distribution.

**RNGCT** Generate pseudorandom numbers from a general continuous distribution.

**RNGDA** Generate pseudorandom numbers from a general discrete distribution using an alias method.

**RNGDS** Set up table to generate pseudorandom numbers from a general discrete distribution.

**RNGDT** Generate pseudorandom numbers from a general discrete distribution using a table lookup method.

**RNGEO** Generate pseudorandom numbers from a geometric distribution.

*MINITAB Interactive System*

**DRANDOM** Generates K pseudo-random numbers from a user-specified discrete distribution.

*NAG Subprogram Library*

**G05DGF** Returns a pseudo-random real number taken from a gamma distribution with parameters G and H.

**G05EXF** Sets up the reference vector R for a discrete distribution with PDF (probability density function) or CDF (cumulative distribution function) P.

*Scientific Desk PC Subprogram Library*

**L6A7** Generates one gamma deviate.

**L6A76** Generates one geometric deviate.

## L6a8 : Halfnormal and hypergeometric random numbers

*DATAPAC Subprogram Library*

**HFNRAN** Generates a random sample of size N from the halfnormal distribution with mean $\sqrt{2/\pi}$ and standard deviation 1.

*IMSL Subprogram Library*

**GGHPR** Hypergeometric random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNHYP** Generate pseudorandom numbers from a hypergeometric distribution.

*NAG Subprogram Library*

**G05EFF** Sets up a reference vector R for a hypergeometric distribution of the number of specified items in a sample of size L, taken from a population of size N with M specified items in it.

## L6a12 : Lambda, logistic, and lognormal random numbers

*DATAPAC Subprogram Library*

**LAMRAN** Generates a random sample of size N from the (Tukey) lambda distribution with tail length parameter $\lambda$.

**LGNRAN** Generates a random sample of size N from the lognormal distribution with mean $\sqrt{e}$.

**LOGRAN** Generates a random sample of size N from the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$.

*IMSL Subprogram Library*

**GGNLG** Lognormal random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNLNL** Generate pseudorandom numbers from a lognormal distribution.

*NAG Subprogram Library*

**G05DCF** Returns a pseudo-random real number taken from a logistic distribution with mean A and spread B.

**G05DEF** Returns a pseudo-random real number taken from a lognormal distribution with parameters A and B.

*Scientific Desk PC Subprogram Library*

**L6A12** Generates one lognormal (e,s) deviate by transforming one normal(0,1) deviate x by using y=exp(e+sx).

**L6a14 : Negative binomial and normal random numbers, random normal order statistics**

*Collected Algorithms of the ACM*

**A599** SEXPO, SGAMMA, SNORM, KPOISS, and SUNIF: Fortran functions which sample from the standard exponential, standard gamma, standard normal, Poisson, and uniform (0,1) distributions. (See J.H. Ahrens, K.D. Kohrt, and U. Dieter, ACM TOMS 9 (1983) pp. 255-257.)

*CMLIB Library (FNLIB Sublibrary)*

**RGAUSS** Normal random number.

*CMLIB Library (RV Sublibrary)*

**RNOR** Generates quasi normal random numbers with zero mean and unit standard deviation.

*DATAPAC Subprogram Library*

**NBRAN** Generates a random sample of size N from the negative binomial distribution with parameters P and NPAR.

**NORRAN** Generates a random sample of size N from the normal (Gaussian) distribution with mean 0 and standard deviation 1.

*IMSL Subprogram Library*

**GGBNR** Negative binomial random deviate generator.

**GGNML** Normal or Gaussian random deviate generator.

**GGNO** Generate set of order statistics from normal distribution.

**GGNPM** Normal random deviate generator via the polar method.

**GGNQF** Normal random deviate generator. Function form of GGNML.

*IMSL STAT/LIBRARY Subprogram Library*

**RNNBN** Generate pseudorandom numbers from a negative binomial distribution.

**RNNOA** Generate pseudorandom numbers from a standard normal distribution using an acceptance/rejection method.

**RNNOF** Generate a pseudorandom number from a standard normal distribution.

**RNNOR** Generate pseudorandom numbers from a standard normal distribution using an inverse CDF method.

**RNNOS** Generate pseudorandom order statistics from a standard normal distribution.

*MINITAB Interactive System*

**NRANDOM** Generates K pseudo-random numbers from the normal distribution with specified mean and standard deviation.

*NAG Subprogram Library*

**G05DDF** Returns a pseudo-random real number taken from a normal (Gaussian) distribution with mean A and standard deviation B.

**G05EEF** Sets up the reference vector R for a negative binomial distribution of the number of successes before N failures, where each trial has probability of success P.

*NMS Subprogram Library*

**RNOR** Standard normal generator with zero mean and unit standard deviation. Uses ziggaraut algorithm. Fast, excellent statistical properties and portable.

*Scientific Desk PC Subprogram Library*

**L6A14F** Generates one normal(0,1) random deviate by inverting the distribution function.

**L6A14S** Generates n normal (0,1) single precision random numbers.

**L6A14T** Generates n normal(0,1) random deviates by inverting the distribution function.

*STARPAC Subprogram Library*

**NRAND** Generate a vector of normal pseudo-random numbers with zero mean and unit standard deviation.

**NRANDC** Generate a vector of normal pseudo-random numbers with mean YMEAN and standard deviation SIGMA.

## L6a16 : Pareto, Pascal, and Poisson random numbers, random permutations

*Collected Algorithms of the ACM*

**A599** SEXPO, SGAMMA, SNORM, KPOISS, and SUNIF: Fortran functions which sample from the standard exponential, standard gamma, standard normal, Poisson, and uniform (0,1) distributions. (See J.H. Ahrens, K.D. Kohrt, and U. Dieter, ACM TOMS 9 (1983) pp. 255-257.)

*DATAPAC Subprogram Library*

**PARRAN** Generates a random sample of size N from the Pareto distribution with tail length parameter $\gamma$.

**POIRAN** Generates a random sample of size N from the Poisson distribution with mean $\lambda$ and standard deviation $\sqrt{\lambda}$.

**RANPER** Generates a random permutation of size N of the values $1.0, 2.0, 3.0, \ldots, N-1, N$.

*IMSL Subprogram Library*

**GGNPP** Nonhomogeneous Poisson process generator with rate function $\lambda(t)$ – fixed interval, fixed number, or one at a time.

**GGPER** Generate a random permutation of the integers 1 to k.

**GGPON** Poisson random deviate generator where the Poisson parameter changes frequently.

**GGPOS** Poisson random deviate generator where the Poisson parameter does not change often.

*IMSL STAT/LIBRARY Subprogram Library*

**RNNPP** Generate pseudorandom numbers from a nonhomogeneous Poisson process.

**RNPER** Generate a pseudorandom permutation.

**RNPOI** Generate pseudorandom numbers from a Poisson distribution.

*MINITAB Interactive System*

**PRANDOM** Generates K pseudo-random numbers from the Poisson distribution with specified population mean K.

*NAG Subprogram Library*

**G05ECF** Sets up the reference vector R for a Poisson distribution with mean T.

**G05EHF** Performs a pseudo-random permutation of a vector of integers.

*Scientific Desk PC Subprogram Library*

**L6A16** Generates one Poisson deviate.

**L6A25** Generates one pseudo-random integer from the Pascal density function.

## L6a19 : Random samples, stable random numbers

*IMSL Subprogram Library*

**GGSRS** Generate a simple random sample from a finite population.

**GGSTA** Stable distribution random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNSRI** Generate a simple pseudorandom sample of indices.

**RNSRS** Generate a simple pseudorandom sample from a finite population.

**RNSTA** Generate pseudorandom numbers from a stable distribution.

*MINITAB Interactive System*

**SAMPLE** Randomly selects without replacement values from one or more vectors, optionally carrying along other vectors.

*NAG Subprogram Library*

**G05EJF** Selects a pseudo-random sample from an integer vector.

## L6a20 : t and triangular random numbers, random time series

*DATAPAC Subprogram Library*

**TRAN** Generates a random sample of size N from the Student's t distribution with degrees of freedom parameter $\nu$.

*IMSL Subprogram Library*

**FTGEN** Generation of a time series from a given ARIMA (Box-Jenkins) stochastic model.

**GGTRA** Triangular distribution random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNARM** Generate a time series from a specified ARMA model.

**RNTRI** Generate pseudorandom numbers from a triangular distribution on the interval (0,1).

*NAG Subprogram Library*

**G05DJF** Returns a pseudo-random real number taken from a Student's t-distribution with N degrees of freedom.

**G05EGF** Sets up a reference vector for an autoregressive moving-average (ARMA) time series model with Normally distributed errors, so that G05EWF may be used to generate successive terms. It also initializes the series to a stationary position.

**G05EWF** Generates the next term from an autoregressive moving-average time series using a reference vector set up by G05EGF.

*Scientific Desk PC Subprogram Library*

**L6A20** Generates one deviate from a symmetric triangular distribution in (0,1).

*TIMSAC Program Library*

**NADCON** Produces noise adaptive optimal controller gain and simulates the controlled process. The basic state space model is obtained from a fixed input-output model of the controlled system and the locally stationary autoregressive model of the vector noise of the controlled system, fitted by either the least squares method or a Bayesian procedure.

**L6a21 : Uniform (continuous and discrete) random numbers, random uniform order statistics**

*Collected Algorithms of the ACM*

**A599** SEXPO, SGAMMA, SNORM, KPOISS, and SUNIF: Fortran functions which sample from the standard exponential, standard gamma, standard normal, Poisson, and uniform (0,1) distributions. (See J.H. Ahrens, K.D. Kohrt, and U. Dieter, ACM TOMS 9 (1983) pp. 255-257.)

*CMLIB Library (FNLIB Sublibrary)*

**RAND** Uniform random number on [0,1].

**RUNIF** Sequence of uniform random numbers on [0,1].

*CMLIB Library (RV Sublibrary)*

**UNI** Uniform random-number generator on [0,1). Uses Fibbonacci algorithm. Fast, excellent statistical properties and highly portable.

*DATAPAC Subprogram Library*

**UNIRAN** Generates a random sample of size N from the uniform (rectangular) distribution on the unit interval (0,1).

*IMSL Subprogram Library*

**GGUBFS** Basic uniform (0,1) random number generator. Function form of GGUBS.

**GGUBS** Basic uniform (0,1) pseudo-random number generator.

**GGUBT** Uniform (0,1) pseudo-random number generator using alternate multiplier.

**GGUD** Discrete uniform random number generator.

**GGUO** Generate set of order statistics from uniform (0,1) distribution.

**GGUW** Uniform (0,1) random number generator with shuffling.

*IMSL MATH/LIBRARY Subprogram Library*

**RNUN** Generate pseudorandom numbers from a uniform (0,1) distribution.

**RNUNF** Generate a pseudorandom number from a uniform (0,1) distribution.

*IMSL STAT/LIBRARY Subprogram Library*

**RNUN** Generate pseudorandom numbers from a uniform (0,1) distribution.

**RNUND** Generate pseudorandom numbers from a discrete uniform distribution.

**RNUNF** Generate a pseudorandom number from a uniform (0,1) distribution.

**RNUNO** Generate pseudorandom order statistics from a uniform (0,1) distribution.

*MINITAB Interactive System*

**IRANDOM** Generates K pseudo-random integers in a specified interval.

**URANDOM** Generates K pseudo-random numbers from the uniform (0,1) distribution.

*NAG Subprogram Library*

**G05CAF** Returns a pseudo-random number taken from a uniform distribution between 0 and 1.

**G05DAF** Returns a pseudo-random real number taken from a uniform distribution between A and B.

**G05DYF** Returns a pseudo-random integer taken from a uniform distribution between M and N (inclusive).

**G05EBF** Sets up the reference vector R for a discrete uniform distribution between M and N inclusive.

**G05EYF** Returns a pseudo-random integer taken from a discrete distribution defined by a reference vector R.

*NMS Subprogram Library*

**UNI** Uniform random-number generator on [0,1). Uses Fibbonacci algorithm. Fast, excellent statistical properties and highly portable.

*PORT Subprogram Library*

**RANBYT** Returns the real random variate generated by UNI, together with its bit pattern presented in four 8-bit bytes.

**UNI** Returns a single real random variate from the uniform [0,1) distribution.

*Scientific Desk PC Subprogram Library*

**L6A21F** Generates one uniform single precision random number in the range (0,1).

**L6A21S** Generates n shuffled uniform single precision random numbers in the range (0,1).

**L6A21T** Generates n uniform single precision random numbers in the range (0,1).

**L6A9** Generates n uniform integer random numbers in the inclusive range (ia,ib).

## L6a22 : Von Mises random numbers

*IMSL Subprogram Library*

**GGVMS** Von Mises random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNVMS** Generate pseudorandom numbers from a von Mises distribution.

## L6a23 : Weibull random numbers

*DATAPAC Subprogram Library*

**WEIRAN** Generates a random sample of size N from the Weibull distribution with tail length parameter $\gamma$.

*IMSL Subprogram Library*

**GGWIB** Weibull random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNWIB** Generate pseudorandom numbers from a Weibull distribution.

*NAG Subprogram Library*

**G05DPF** Returns a pseudo-random number taken from a two parameter Weibull distribution with shape parameter A and scale parameter B.

*Scientific Desk PC Subprogram Library*

**L6A23** Generates one Weibull deviate.

## L6b3 : Random contingency tables and correlation matrices

*IMSL Subprogram Library*

**GGCOR** Generate a random orthogonal matrix and a random correlation matrix.

**GGTAB** Generate a random contingency table with given row and column totals.

*IMSL STAT/LIBRARY Subprogram Library*

**RNCOR** Generate a pseudorandom orthogonal matrix or a correlation matrix.

**RNTAB** Generate a pseudorandom two-way table.

## L6b5 : Random experimental designs

*SAS Program Library*

**PLAN** Generates random permutations of positive integers for experimental plans (e.g., completely random, split-plot, and hierarchical designs) given a specification of the randomized plan including number of levels of nesting. Option: seed of first permutation.

## L6b12 : Linear $L_1$ (least absolute value) approximation random numbers

*Collected Algorithms of the ACM*

**A564** L1GNR: a Fortran subroutine for generating test problems for discrete linear L-sub-1 approximation problems. (See K.L. Hoffman and D.R. Shier, ACM TOMS 6 (1980) pp. 615-617.)

## L6b13 : Multinomial random numbers

*IMSL Subprogram Library*

**GGMTN** Multinomial random deviate generator.

*IMSL STAT/LIBRARY Subprogram Library*

**RNMTN** Generate pseudorandom numbers from a multinomial distribution.

*Scientific Desk PC Subprogram Library*

**L6B13** Generates one multinomial (k-tuple) deviate.

## L6b14 : Multivariate normal random numbers

*IMSL Subprogram Library*

**GGNSM** Multivariate normal random deviate generator with given covariance matrix.

*IMSL STAT/LIBRARY Subprogram Library*

**RNMVN** Generate pseudorandom numbers from a multivariate normal distribution.

*NAG Subprogram Library*

**G05EZF** Returns a pseudo-random multivariate Normal vector taken from a distribution described by a reference vector set up by G05EAF.

*Scientific Desk PC Subprogram Library*

**L6B14** Generates k-plet multinormal deviates with mean 0 and covariance matrix cov.

## L6b15 : Random orthogonal matrices

*IMSL Subprogram Library*

**GGCOR** Generate a random orthogonal matrix and a random correlation matrix.

*IMSL STAT/LIBRARY Subprogram Library*

**RNCOR** Generate a pseudorandom orthogonal matrix or a correlation matrix.

## L6b21 : Multivariate uniform random numbers

*Collected Algorithms of the ACM*

**A647** Fortran subprograms for the generation of sequences of quasirandom vectors with low discrepancy. Such sequences may be used to reduce error bounds for multidimensional integration and global optimization. (See B.L. Fox, ACM TOMS 12 (1986) pp. 362-376.)

*IMSL Subprogram Library*

**GGSPH** Generation of uniform random deviates from the surface of the unit sphere in 3 or 4 space.

*IMSL STAT/LIBRARY Subprogram Library*

**RNSPH** Generate pseudorandom points on a unit circle or K-dimensional sphere.

## L6c : Service routines for random number generation (e.g., seed)

*IMSL MATH/LIBRARY Subprogram Library*

**RNGET** Retrieve the current value of the seed used in the IMSL random number generators.

**RNOPT** Select the uniform (0,1) multiplicative congruential pseudorandom number generator.

**RNSET** Initialize a random seed for use in the IMSL random number generators.

*IMSL STAT/LIBRARY Subprogram Library*

**RNGES** Retrieve the current value of the table in the IMSL random number generators that use shuffling.

**RNGET** Retrieve the current value of the seed used in the IMSL random number generators.

**RNOPG** Obtain the indicator of the uniform (0,1) multiplicative congruential pseudorandom number generator.

**RNOPT** Select the uniform (0,1) multiplicative congruential pseudorandom number generator.

**RNSES** Initialize the table in the IMSL random number generators that use shuffling.

**RNSET** Initialize a random seed for use in the IMSL random number generators.

*NAG Subprogram Library*

**G05CBF** Sets the basic generator routine G05CAF to a repeatable initial state.

**G05CCF** Sets the basic generator routine G05CAF to a non-repeatable initial state.

**G05CFF** Save state of random number generating routines.

**G05CGF** Restores the state of the basic generator routine G05CAF after a previous call to G05CFF.

*PORT Subprogram Library*

**RANSET** Initializes the uniform random number generator, UNI, to other than the default initial values.

## L7: Analysis of Variance

Software documented in this class is first organized by the dimension of the data, manifested as the type of analysis to be performed. In one-way analysis of variance (ANOVA) the data are one-dimensional observations from two or more populations (often called groups or treatments). Underlying two-way ANOVA is a two-dimensional matrix of cells representing different treatments; one-dimensional observations are associated with each cell. Underlying three-way ANOVA is either a third dimension of treatments superimposed on this two-dimensional matrix or a three-dimensional matrix. Multi-way ANOVA has an underlying multi-dimensional matrix. The observations are multi-dimensional in multivariate ANOVA.

The populations are fixed in fixed effects models; they are sampled from a distribution in random effects models.

Most analysis of variance procedures assume that the observations in each cell have independent, normally distributed errors with zero mean and constant variance. The appropriateness of any assumptions should be evaluated by statistical and/or graphical techniques. Transformation of the data may make such assumptions more reasonable (see class L2). Non-parametric (distribution-free) analyses, which use ranks rather than raw data, have less restrictive assumptions.

The observational data are *balanced* if an equal number of observations is associated with each cell. They are *complete* if there is at least one observation associated with each cell. The algorithms for one-way ANOVA and for higher-dimensional ANOVA for balanced complete data are straightforward and the structure of the output parallels that found in classic texts; special care must be taken in using and analyzing results from other ANOVA analyses.

### References

[BHH78] G. E. P. Box, W. G. Hunter, and J. S. Hunter. *Statistics for Experimenters.* John Wiley & Sons, New York, 1978.

[CC57] W. G. Cochran and G. M. Cox. *Experimental Design.* John Wiley & Sons, New York, 1957.

[Cox58] D. R. Cox. *Planning of Experiments.* John Wiley & Sons, New York, 1958.

[Deh89] K. Dehnad. *Quality Control, Robust Design and the Taguchi Method.* Wadsworth and Brooks/Cole, Pacific Grove, CA, 1989.

[Hic73] C. R. Hicks. *Fundamental Concepts in Design of Experiments.* Holt, Rinehart and Winston, New York, 1973.

[Mil86] R. Miller. *Beyond ANOVA, Basics of Applied Statistics.* John Wiley & Sons, New York, 1986.

[MJ84] G. A. Milliken and D. E. Johnson. *Analysis of Messy Data*, volume 1. Wadsworth, Inc., Belmont, CA, 1984.

[Nat63] M. G. Natrella. *Experimental Statistics, NBS Handbook 91.* U. S. Government Printing Office, Washington, D. C., 1963.

[NWK88] J. Neter, W. Wasserman, and M. Kutner. *Applied Linear Regression Models.* Richard D. Irwin, Inc., Homewood, Illinois, 2nd edition, 1988.

[Sea71] S. R. Searle. *Linear Models.* John Wiley & Sons, New York, 1971.

## L7a : One-way analysis of variance

*BMDP Program Library*

**P7D** Side-by-side histograms for each cell in one-way or two-way ANOVA, within-group summary statistics and ANOVA table (with equality of variance test and tests that do not assume equal variances). Options: trimmed mean analysis, ANOVA diagnostics, tests of pairwise mean comparisons, correlations, Winsorized means.

*STARPAC Subprogram Library*

• **AOV1** Compute and print a one-way analysis of variance of the input data.

**AOV1S** Compute and optionally print a one-way analysis of variance of the input data; return tag value of each group, number of observations in each group, group averages, and group standard deviations.

## L7a1 : Parametric one-way analysis of variance

*BMDP Program Library*

**P1V** Performs one-way ANOVA or ANCOVA with standard results. For ANCOVA, tests 1) equality of slopes, 2) zero slope, and 3) equality of adjusted cell means; plots the covariate for each group. Tests equality of pairs of means (or adjusted means). Options: linear contrasts, within-group correlations and statistics.

*DATAPLOT Interactive System*

**ANOVA** Performs analysis of variance on balanced data with up to 5 factors. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviation, residual degrees of freedom, F-tests of significance, as well as replication standard deviation, replication degrees of freedom, and lack of fit F statistic if there is replication.

*IMSL Subprogram Library*

**ACRDAN** Analysis of one-way classification design data.

**ACTRST** Contrast estimates and sums of squares.

**AGVACL** One or two-sided interval estimate of a variance component.

**ANCOV1** Covariance analysis for one-way classification design data.

**ASNKMC** Student-Newman-Keuls multiple comparison test.

*IMSL STAT/LIBRARY Subprogram Library*

**AONEC** Analyze a one-way classification model with covariates.

**AONEW** Analyze a one-way classification model.

**CTRST** Compute contrast estimates and sums of squares.

**SNKMC** Perform Student-Newman-Keuls multiple comparison test.

*MINITAB Interactive System*

**AOVONEWAY** Performs a one-way analysis of variance and prints standard results.

**ONEWAY** Performs one-way analysis of variance, prints standard results, and optionally saves results.

• **easy-to-use version**

*NAG Subprogram Library*

**G04AEF** Performs an analysis of variance for a one-way classification with treatment groups of possibly unequal size, and also computes the treatment group means. A fixed effects model is assumed.

*Scientific Desk PC Subprogram Library*

**L7A11** Computes averages, within treatment standard deviations, and the associated analysis of variance table for a one-way classification with an equal or unequal number of observations per treatment.

*SPSS Program Library*

**ANOVA** Performs one- to five-way analysis of variance and covariance for factorial designs. Options: decomposition of sum of squares by classical, regression, or hierarchical approach; control of order of entry of covariates and factor main effects.

**ONEWAY** Performs one-way ANOVA. Options: descriptive statistics, homogeneity of variance tests, fixed- and random-effects measures, test for trends, a priori contrasts, seven range tests for pairwise comparison, missing values.

## L7a2 : Nonparametric one-way analysis of variance

*BMDP Program Library*

**P3S** Computes and prints results from one or more of the following: sign test, Wilcoxon signed-rank test, Mann-Whitney rank-sum test, Kruskall-Wallis one-way ANOVA, Friedman two-way ANOVA, Kendall's coefficient of concordance, Kendall and Spearman rank-correlation coefficients.

*IMSL Subprogram Library*

**NAK1** Kruskal-Wallis test for identical populations.

**NAWRPE** Wilsons ANOVA (1, 2, or 3 way designs) with equal replication.

**NAWRPU** Wilsons ANOVA (1, 2, or 3 way designs) with unequal replication.

*MINITAB Interactive System*

**KRUSKAL-WALLIS** Perform Kruskal-Wallis test, based on ranks, of the null hypothesis that there is no difference among K population locations against the alternative of at least one difference. (This is a K-sample generalization of the Mann-Whitney-Wilcoxon test and is a nonparametric alternative to one-way ANOVA.)

*NAG Subprogram Library*

**G08AFF** Performs the Kruskal-Wallis one-way analysis of variance by ranks on k independent samples of possibly unequal sizes.

*SAS Program Library*

**NPAR1WAY** Performs nonparametric one-way analysis of variance on ranks and four rank scores (Wilcoxon, median, van der Waerden, and Savage).

*SPSS Program Library*

**NPAR TESTS** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values.

**L7b :** **Two-way analysis of variance** *(search also class L7d)*

*BMDP Program Library*

**P7D** Side-by-side histograms for each cell in one-way or two-way ANOVA, within-group summary statistics and ANOVA table (with equality of variance test and tests that do not assume equal variances). Options: trimmed mean analysis, ANOVA diagnostics, tests of pairwise mean comparisons, correlations, Winsorized means.

*DATAPLOT Interactive System*

**MEDIAN POLISH** Performs a 1-5 way median polish on balanced data. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviations, residual degrees of freedom, as well as replication standard deviation, replication degrees of freedom, and lack-of-fit F-statistic if there is replication.

*IMSL Subprogram Library*

**ARCBAN** Analysis of two-way classification design data.

**BEMDP** Median polish of a two-way table.

**NAFRE** Friedman's test for randomized complete block designs.

**NAWNRP** Wilson's ANOVA (2 or 3 way designs) without replicates.

**NAWRPE** Wilsons ANOVA (1, 2, or 3 way designs) with equal replication.

**NAWRPU** Wilsons ANOVA (1, 2, or 3 way designs) with unequal replication.

*IMSL STAT/LIBRARY Subprogram Library*

**ATWOB** Analyze a randomized block design or a two-way balanced design.

**FRDMN** Perform Friedman's test for a randomized complete block design.

**MEDPL** Compute a median polish of a two-way table.

*MINITAB Interactive System*

**MPOLISH** Uses median polish to fit an additive model to a two-way layout which may be unbalanced and may have empty cells. Options: fit columns first, number of iterations, save results.

**TWOWAYAOV** Performs two-way analysis of variance for balanced data (equal number of observations, one or more, in each cell) and prints standard results. Options: fit additive model, save results.

*NAG Subprogram Library*

**G04AFF** Performs an analysis of variance for a two-way cross-classification with equal cell frequencies. A fixed effects model is assumed.

**G04AGF** Performs an analysis of variance for a two-way hierarchical classification with subgroups of possibly unequal size, and also computes the treatment group and subgroup means.

**G08AEF** Performs the Friedman two-way analysis of variance by ranks on K related samples of size N.

**L7c :** **Three-way analysis of variance** *(search also class L7d)*

*IMSL Subprogram Library*

**ALSQAN** Analysis of Latin square design data.

**NAWNRP** Wilson's ANOVA (2 or 3 way designs) without replicates.

**NAWRPE** Wilsons ANOVA (1, 2, or 3 way designs) with equal replication.

**NAWRPU** Wilsons ANOVA (1, 2, or 3 way designs) with unequal replication.

*IMSL STAT/LIBRARY Subprogram Library*

**ALATN** Analyze a Latin square design.

*NAG Subprogram Library*

**G04ADF** Performs the analysis of variance for a Latin square design.

| **L7d :** | **Multi-way analysis of variance** |
|---|---|

*BMDP Program Library*

**P2V** ANOVA and ANCOVA for unbalanced fixed-effect models (including full and fractional factorial designs, Latin squares), and repeated measure models, or a combination of models, with Greenhouse-Geisser and Huynh-Feldt degree of freedom adjustment. Options: orthogonal decomposition of within-effects, save results.

**P3V** Uses maximum likelihood (ML) and restricted ML approaches to balanced and unbalanced fixed and random coefficient models of quite arbitrary form (including having covariates), with parameter estimation, hypothesis testing, and printing. Weights optional.

**P4V** Interactive or batch univariate and multivariate ANOVA and ANCOVA, including nested, repeated measures, split-plot, and changeover designs, and model building features. Options: cell weights for hypothesis testing, contrasts, tests of simple effects, save cell means.

*Collected Algorithms of the ACM*

**A591** A storage-efficient Fortran program for analysis of variance of balanced data, unbalanced data, and unbalanced data with missing cells. (See W.J. Hemmerle, ACM TOMS 8 (1982) pp. 383-401.)

*SPSS Program Library*

**MANOVA** Generalized multivariate analysis of variance for designs including randomized block, split-plot, crossed and/or nested, and repeated measures, with or without covariates. Options: multivariate statistics, contrasts, plots, missing values.

| **L7d1 :** | **Multi-way analysis of variance for balanced complete data** |
|---|---|

*BMDP Program Library*

**P8V** ANOVA for complete designs with equal cell sizes – nested, crossed, partially nested, partially crossed designs for fixed-effect models, mixed models (including repeated measures), and random-effect models, with parameter estimation and printing.

*DATAPLOT Interactive System*

**ANOVA** Performs analysis of variance on balanced data with up to 5 factors. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviation, residual degrees of freedom, F-tests of significance, as well as replication standard deviation, replication degrees of freedom, and lack of fit F statistic if there is replication.

**MEDIAN POLISH** Performs a 1-5 way median polish on balanced data. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviations, residual degrees of freedom, as well as replication standard deviation, replication degrees of freedom, and lack-of-fit F-statistic if there is replication.

**YATES ANALYSIS** Performs a Yates analysis of an experimental design. It estimates the factor effects for $2^k$ and $2^{k-p}$ factorial designs. The data must be entered in standard form.

*IMSL Subprogram Library*

**AFACN** Full factorial plan analysis.

• **AFACT** Full factorial plan analysis – easy to use version.

**AGBACP** Analysis of balanced complete experimental design structure data.

**AGXPM** Expected mean squares for balanced complete design models.

**ANESTE** Analysis of completely nested design data with equal numbers in the subclasses.

**AORDR** Reordering of the data obtained from any balanced complete experimental design.

*IMSL STAT/LIBRARY Subprogram Library*

**ABALD** Analyze a balanced complete experimental design for a fixed, random, or mixed model.

**ANEST** Analyze a completely nested random model with possibly unequal numbers in the subgroups.

**ANWAY** Analyze a balanced n-way classification model with fixed effects.

**CIDMS** Compute a confidence interval on a variance component estimated as proportional to the difference in two mean squares in a balanced complete experimental design.

**ROREX** Reorder the responses from a balanced complete experimental design.

*SAS Program Library*

**ANOVA** Performs univariate and multivariate analysis of variance for balanced data, including Latin-square, certain balanced incomplete block designs, completely nested (hierarchical) designs. Options: numerous means comparisons, missing values.

*SPSS Program Library*

**ANOVA** Performs one- to five-way analysis of variance and covariance for factorial designs. Options: decomposition of sum of squares by classical, regression, or hierarchical approach; control of order of entry of covariates and factor main effects.

## L7d2 : Multi-way analysis of variance for balanced incomplete data

*IMSL Subprogram Library*

**ABIBN** Analysis of balanced incomplete block and balanced lattice designs.

*IMSL STAT/LIBRARY Subprogram Library*

**ABIBD** Analyze a balanced incomplete block design or a balanced lattice design.

*SAS Program Library*

**ANOVA** Performs univariate and multivariate analysis of variance for balanced data, including Latin-square, certain balanced incomplete block designs, completely nested (hierarchical) designs. Options: numerous means comparisons, missing values.

## L7d3 : General linear models (unbalanced data)

*IMSL Subprogram Library*

**AGLMOD** General linear model analysis.

**AMEANS** Preparation of a set of unbalanced data for analysis by the method of unweighted means.

**ANESTU** Analysis of completely nested design data with unequal number in the subclasses.

*IMSL STAT/LIBRARY Subprogram Library*

**ANEST** Analyze a completely nested random model with possibly unequal numbers in the subgroups.

• **easy-to-use version**

*SAS Program Library*

**GLM** Performs simple and multiple least-squares regression, analysis of variance (especially for unbalanced data), analysis of covariance, response-surface regression, polynomial regression, partial correlation, multivariate analysis of variance, and repeated measures analysis of variance. Options: weights, missing values.

**NESTED** Performs analysis of variance and analysis of covariance for nested random designs. Especially good for designs involving large numbers of classification levels and observations. The data set must be sorted by the classification variables (assumed to form a nested set of effects).

**VARCOMP** Provides four methods (Type I, MIVQUEO, maximum-likelihood, and restricted maximum-likelihood) for estimating variance components in a general linear model containing random effects and optionally fixed effects. Option: missing values.

| L7e : | Multivariate analysis of variance |
|---|---|

*BMDP Program Library*

**P4V** Interactive or batch univariate and multivariate ANOVA and ANCOVA, including nested, repeated measures, split-plot, and changeover designs, and model building features. Options: cell weights for hypothesis testing, contrasts, tests of simple effects, save cell means.

*SAS Program Library*

**ANOVA** Performs univariate and multivariate analysis of variance for balanced data, including Latin-square, certain balanced incomplete block designs, completely nested (hierarchical) designs. Options: numerous means comparisons, missing values.

**GLM** Performs simple and multiple least-squares regression, analysis of variance (especially for unbalanced data), analysis of covariance, response-surface regression, polynomial regression, partial correlation, multivariate analysis of variance, and repeated measures analysis of variance. Options: weights, missing values.

| L7f : | Generate experimental designs |
|---|---|

*SAS Program Library*

**PLAN** Generates random permutations of positive integers for experimental plans (e.g., completely random, split-plot, and hierarchical designs) given a specification of the randomized plan including number of levels of nesting. Option: seed of first permutation.

| L7g : | Service routines for analysis of variance |
|---|---|

*BMDP Program Library*

**P9D** Provides descriptive statistics (means, standard deviations, frequencies, one-way ANOVA table) of groups (cells) for data classified into cells using one or more grouping variables. Options: miniplots of cell means (eight per page), plot frames are defined by combinations of levels of grouping variables.

## L8: Regression

The organization of class L8 is by the form of the regression model rather than by the dimension of the data; one input variable can be propogated into more than one variable in the model and the analyses in these cases can be quite different (simple linear regression and polynomial regression are illustrative).

In *simple linear regression* a straight line is fit to $(x_i, y_i), i = 1, \ldots, n$. Thus the model is:

$$y_i = b_0 + b_1 x_i.$$

The $i$-subscripts are dropped for simplicity:

$$y = b_0 + b_1 x.$$

This model is *linear* because it is linear in its parameters ($b_0$ and $b_1$). *Simple* refers to the straight line fit.

In *polynomial regression* the right hand side of the model is a polynomial in the independent variable $x$:

$$y = b_0 + b_1 x + \ldots + b_p x^p.$$

This is also a linear model.

In *multiple linear regression* there is more than one independent variable. Labeling those variables $x_1, \ldots, x_p$, the model is:

$$y = b_0 + b_1 x_1 + \ldots + b_p x_p.$$

To illustrate the model for polynomial regression in several variables, the quadratic model for two variables is:

$$y = b_0 + b_1 x_1 + b_2 x_1^2 + b_3 x_2 + b_4 x_2^2 + b_5 x_1 x_2.$$

In *nonlinear regression* the model is not linear in its parameters. A simple example with one nonlinear parameter (in the exponent) is:

$$y = b_0 + b_1 e^{b_2 x}.$$

In *simultaneous regression* the left hand side of the model is a matrix (with column dimension greater than 1) rather than a vector. In full matrix notation, this model is:

$$Y = Xb.$$

See classes E and K for a description of splines.

Those classes L8a through L8h for which there are large collections of software are further partitioned into parallel subclasses. The distinguishing features of this next level are the form of the error structure and the form of the objective function which is to be minimized. In *ordinary* regression the dependent variable $y$ is observed with error but independent $x$ variables are observed without error; in *errors in variables* regression and *measurement error models* the $y$-variable also is observed with error. In *least squares* regression the sum of the squares of the differences between the observed $y$-value and the fitted $y$-value ($\hat{y}$) is minimized:

$$\min \sum_{\text{all } i} (y_i - \hat{y}_i)^2.$$

The exponent 2 changes in $L_p$ regression for $p$ different from 2. In *robust* regression, weights may be automatically assigned to the data and iteratively updated to minimize the effects of outliers.

In ordinary least squares regression, single software modules often provide parameter estimation, analysis (e.g., graphics for analyzing whether the residuals are normally distributed and have constant variance), and inference (e.g., tests of hypothesis such as that a parameter is zero).

In *ridge regression* bias is introduced in the estimates of the regression parameters in return for smaller variances on those estimates.

If several regression analyses are to be performed on the same set (or subsets of the same set) of data, then more efficient software is available in class L8c2 than in L8c1.

## References

[DS81] N. R. Draper and H. Smith. *Applied Regression Analysis.* Wiley-Interscience, New York, 1981.

[DW80] C. Daniel and F. S. Wood. *Fitting Equations to Data.* Wiley-Interscience, New York, 1980.

[Ful87] W. Fuller. *Measurement Error Models.* John Wiley & Sons, New York, 1987.

[MT77] F. Mosteller and J. W. Tukey. *Data Analysis and Regression: A Second Course in Statistics.* Addison-Wesley, Reading, MA, 1977.

[NWK88] J. Neter, W. Wasserman, and M. Kutner. *Applied Linear Regression Models.* Richard D. Irwin, Inc., Homewood, Illinois, 2nd edition, 1988.

[Seb77] G. A. F. Seber. *Linear Regression Analysis.* John Wiley & Sons, New York, 1977.

[Seb89] G. A. F. Seber. *Nonlinear Regression.* John Wiley & Sons, New York, 1989.

**L8a1a : Parameter estimation in simple linear regression (i.e., $y = b_0 + b_1 x$)**

*BMDP Program Library*

**P1R** Performs multiple linear regression and prints standard results. Options: weights, form of input, regression on subsets or groups and test of equality of regression lines, intercept term present or absent, more printing, five plots, save predicted values and residuals.

*IMSL STAT/LIBRARY Subprogram Library*

**RONE** Analyze a simple linear regression model.

*MINITAB Interactive System*

**REGRESS** Performs simple or multiple linear regression, prints standard results. Options: amount of output, save results, weights, missing values, through the origin, compute and save regression diagnostics, lack of fit tests.

*SAS Program Library*

**REG** Fits least-squares estimates to linear regression models. Options: weights; parameter estimates, predicted values, residuals, Studentized residuals, confidence limits, hypothesis tests; collinearity diagnostics; influence diagnostics including partial regression leverage plots; Durbin-Watson statistic; hypothesis tests involving multiple dependent variables; parameter estimates subject to linear restriction.

**L8a1a1 : Parameter estimation in simple linear regression using unweighted data**

*DATAPLOT Interactive System*

**FIT** Fits a nonlinear model (including linear and polynomial models) to data by least squares. Output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication. Options: user-set starting values, number of iterations, accuracy, weighted and robust fits.

*IMSL Subprogram Library*

**BEMIRI** Estimates means, simple regression coefficients, their intercepts, standard errors of the regression coefficients, and standard deviations for arrays which contain missing values (in-core version).

**BEMIRO** Estimates means, simple regression coefficients, their intercepts, standard errors of the regression coefficients, and standard deviations for arrays which contain missing values (out-of-core version).

**RLONE** Analysis of a simple linear regression model.

*IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY Subprogram Libraries*

**RLINE** Fit a line to a set of data points using least squares.

*NAG Subprogram Library*

**G02CAF** Performs a simple linear regression with dependent variable Y and independent variable X.

**G02CBF** Performs a simple linear regression with no constant with dependent variable Y and independent variable X.

**G02CCF** Performs a simple linear regression with dependent variable Y and independent variable X, omitting cases involving missing values.

**G02CDF** Performs a simple linear regression with no constant, with dependent variable Y and independent variable X, omitting cases involving missing values.

*Scientific Desk PC Subprogram Library*

**L8A1A** Performs simple linear regression for the model $Y = a + bX$.

*SPSS Program Library*

**REGRESSION** Calculates multiple regression equation and associated statistics and plots. Options: backward, forward, stepwise, or subset of variables selection, residual diagnostics and analysis, partial residual plots, missing values.

*STARPAC Subprogram Library*

• **LLS** Compute and print a four-part unweighted linear least squares analysis with user-specified model (design matrix); return residuals.

**LLSS** Compute and optionally print a four-part unweighted linear least squares analysis with user-specified model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters.

## L8a1a2 : Parameter estimation in simple linear regression using weighted data

*STARPAC Subprogram Library*

• **LLSW** Compute and print a four-part weighted linear least squares analysis with user-specified model (design matrix); return residuals.

**LLSWS** Compute and optionally print a four-part weighted linear least squares analysis with user-specified model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters.

• **easy-to-use version**

| **L8a1d :** | **Inference in simple linear regression (e.g., calibration)** *(search also class L8a1a)* |
|---|---|

*IMSL Subprogram Library*

**RLINCF** Response control using a fitted simple linear regression model.

**RLINPF** Inverse prediction using a fitted simple linear regression model.

*IMSL STAT/LIBRARY Subprogram Library*

**RINCF** Perform response control given a fitted simple linear regression model.

**RINPF** Perform inverse prediction given a fitted simple linear regression model.

| **L8a2 :** | **Simple linear $L_p$ regression for $p$ different from 2 (e.g., least absolute value, minimax)** |
|---|---|

*Collected Algorithms of the ACM*

**A564** L1GNR: a Fortran subroutine for generating test problems for discrete linear L-sub-1 approximation problems. (See K.L. Hoffman and D.R. Shier, ACM TOMS 6 (1980) pp. 615-617.)

*CMLIB Library (SLRPACK Sublibrary)*

**LINFS** Solves the simple model, $Y = \beta_1 + \beta_2 X$, under the Chebychev norm criterion.

*IMSL Subprogram Library*

**RLLAV** Perform linear regression using the least absolute values criterion.

**RLLMV** Perform linear regression using the minimax criterion.

*IMSL STAT/LIBRARY Subprogram Library*

**RLAV** Fit a multiple linear regression model using the least absolute values criterion.

**RLMV** Fit a multiple linear regression model using the minimax criterion.

*Scientific Desk PC Subprogram Library*

**L8CS** Calculates simple linear regression least absolute value estimates. The least absolute curve fitting problem is to find $a$ and $b$ such that the sum of $|Y_i - a - X_i b|$ is minimized, $(i = 1, \ldots, n)$.

| **L8a3 :** | **Robust simple linear regression** |
|---|---|

*DATAPLOT Interactive System*

**LOWESS SMOOTH** Perform a locally weighted least squares analysis of a two-dimensional data set. Output includes predicted and residual values. The user can specify the smoothing interval.

*ROSEPAK Interactive Program Library*

**IRLS** Robust regression by iteratively reweighted least squares. Weights are determined from the data and are computed as functions of the scaled residuals. Options: $L_1$ and $L_2$ starts, eight weight functions, rank determination, and convergence criterion.

**L8a4 : Errors in variables regression**

*CMLIB Library (ODRPACK Sublibrary)*

**SODR** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated.

**SODRC** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated.

*CMLIB Library (SLRPACK Sublibrary)*

**RGM** Computes estimates of simple linear regression parameters for a geometric mean regression.

**RWILL** Estimates simple linear regression coefficients when both variables are subject to errors which are not necessarily homogeneous in variance (method due to Williamson).

**RYORK** Estimates simple linear regression coefficients when both variables are subject to errors which are not necessarily homogeneous in variance.

**L8b1a : Degree determination in polynomial regression (i.e., $y = b_0+b_1x+b_2x^2+\ldots b_px^p$)**

*IMSL STAT/LIBRARY Subprogram Library*

**RPOLY** Analyze a polynomial regression model.

**L8b1b1 : Parameter estimation in polynomial regression not using orthogonal polynomials**

*DATAPAC Subprogram Library*

**POLY** Computes a least squares polynomial fit (of degree IDEG) of the response variable data in the vector Y as a function of vector X and with optional weights.

*DATAPLOT Interactive System*

**FIT** Fits a nonlinear model (including linear and polynomial models) to data by least squares. Output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication. Options: user-set starting values, number of iterations, accuracy, weighted and robust fits.

*STARPAC Subprogram Library*

• **LLSP** Compute and print a four-part unweighted linear least squares analysis with polynomial model (design matrix); return residuals.

**LLSPS** Compute and optionally print a four-part unweighted linear least squares analysis with polynomial model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters.

**LLSPW** Compute and print a four-part weighted linear least squares analysis with polynomial model (design matrix); return residuals.

**LLSPWS** Compute and optionally print a four-part weighted linear least squares analysis with polynomial model (design matrix); return residuals, parameter estimates, residual standard devia-

• **easy-to-use version**

tion, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters.

**L8b1b2 : Parameter estimation in polynomial regression using orthogonal polynomials**

*BMDP Program Library*

**P5R** Least squares fit of a polynomial in one independent variable to the dependent variable. Prints standard results and goodness-of-fit statistics for each polynomial degree. Computations use orthogonal polynomials. Options: weights, additional printing, and three plots.

*IMSL Subprogram Library*

**RLDCQM** Decoding of a quadratic regression model.

**RLDCVA** Variance estimates for decoded orthogonal polynomial regression coefficients.

**RLDCW** Variances of coded orthogonal polynomial regression coefficients for usage in conjunction with IMSL routines RLFOTH and RLFOTW, and provided to prepare input for IMSL routine RLDCVA.

**RLDOPM** Coefficient decoder for an orthogonal polynomial regression model.

• **RLFOR** Fit a univariate curvilinear regression model using orthogonal polynomials with optional weighting and prediction analysis - easy-to-use version.

**RLFOTH** Fit a univariate curvilinear regression model using orthogonal polynomials.

**RLFOTW** Fit a univariate curvilinear regression model using orthogonal polynomials with weighting.

**RLPOL** Generate orthogonal polynomials with associated constants AA and BB.

*IMSL STAT/LIBRARY Subprogram Library*

**OPOLY** Generate orthogonal polynomials with respect to a specified interval and weights.

**RCURV** Fit a polynomial curve using least squares.

**RFORP** Fit an orthogonal polynomial regression model.

**TCSCP** Transform coefficients from a quadratic regression model generated from squares and crossproducts of centered variables to a model using uncentered variables.

*Scientific Desk PC Subprogram Library*

**L8A2BD** Performs weighted polynomial regression using orthogonal polynomials.

**L8b1c : Analysis of polynomial regression results** *(search also class L8b1b)*

*IMSL STAT/LIBRARY Subprogram Library*

**RCASP** Compute case statistics for a polynomial regression model given the fit based on orthogonal polynomials.

**RSTAP** Compute summary statistics for a polynomial regression model given the fit based on orthogonal polynomials.

**L8b1d : Inference in polynomial regression** *(search also class L8b1b)*

*IMSL Subprogram Library*

**RLOPDC** Response prediction using an orthogonal polynomial regression model.

*IMSL STAT/LIBRARY Subprogram Library*

**RCASP** Compute case statistics for a polynomial regression model given the fit based on orthogonal polynomials.

**RSTAP** Compute summary statistics for a polynomial regression model given the fit based on orthogonal polynomials.

**L8c1a1 : Variable selection in multiple linear regression using raw data (i.e., $y = b_0 + b_1x_1 + \ldots + b_px_p$)**

*BMDP Program Library*

**P2R** Multiple linear regression, with standard results. Options: weights, form of input, forward or backward stepping, interactive stepping, stepping sets of variables (e.g. design variables), forcing variables into the model, eleven diagnostics (including Cook and AP statistics) available for printing, plotting, and saving.

**P9R** Estimates regression equations for "best" (by R-squared, adjusted R-squared, or Mallows' C(p) criterion) subset of predictor variables by Furnival-Wilson algorithm. Options: weights, form of input, Durbin-Watson statistic. Cook's distance and several types of residuals may be printed, plotted, or saved.

*IMSL Subprogram Library*

**RLSEP** Selection of a regression model using a forward stepwise algorithm, and computation of the usual analysis of variance table entries – easy-to-use version.

*MINITAB Interactive System*

**STEPWISE** Performs stepwise linear regression using forward selection, backward elimination, conventional stepwise, or user intervention. Options available through subcommands: F-to-enter and F-to-remove, force and remove sets of variables, print next "best" (by the F-statistic) K alternatives.

*ROSEPAK Interactive Program Library*

**VNULL** A program for detecting rank degeneracy. It can select a set of independent columns of a matrix when numerical rank can be determined.

*SAS Program Library*

**RSQUARE** Uses the R-squared statistic to select optimal subsets of independent variables for multiple regression. Can specify largest and smallest number of independent variables for a subset and number of subsets of each size. Options: weights; statistics for each model selected including Akaike's information criterion, Mallows' C-p, and others.

**STEPWISE** Provides five methods (forward selection, backward elimination, stepwise, maximum and minimum R-squared improvements) for stepwise regression. Options: weights, significance levels, Mallows' C-p statistic.

*SPSS Program Library*

**REGRESSION** Calculates multiple regression equation and associated statistics and plots. Options: backward, forward, stepwise, or subset of variables selection, residual diagnostics and analysis, partial residual plots, missing values.

## L8c1a2 : Variable selection in multiple linear regression using correlation or covariance data

*BMDP Program Library*

**P2R** Multiple linear regression, with standard results. Options: weights, form of input, forward or backward stepping, interactive stepping, stepping sets of variables (e.g. design variables), forcing variables into the model, eleven diagnostics (including Cook and AP statistics) available for printing, plotting, and saving.

**P9R** Estimates regression equations for "best" (by R-squared, adjusted R-squared, or Mallows' C(p) criterion) subset of predictor variables by Furnival-Wilson algorithm. Options: weights, form of input, Durbin-Watson statistic. Cook's distance and several types of residuals may be printed, plotted, or saved.

*IMSL Subprogram Library*

**RLEAP** Leaps and bounds algorithm for determining a number of best regression subsets from a full regression model. USLEAP is a special purpose output routine designed to be used only in conjunction with RLEAP.

**RLSTP** Regression model selection using a forward stepwise algorithm with results available after each step.

*IMSL STAT/LIBRARY Subprogram Library*

**RBEST** Select the best multiple linear regression models.

**RSTEP** Build multiple linear regression models using forward selection, backward selection, or stepwise selection.

*SAS Program Library*

**RSQUARE** Uses the R-squared statistic to select optimal subsets of independent variables for multiple regression. Can specify largest and smallest number of independent variables for a subset and number of subsets of each size. Options: weights; statistics for each model selected including Akaike's information criterion, Mallows' C-p, and others.

*SPSS Program Library*

**REGRESSION** Calculates multiple regression equation and associated statistics and plots. Options: backward, forward, stepwise, or subset of variables selection, residual diagnostics and analysis, partial residual plots, missing values.

## L8c1a3 : Variable selection in multiple linear regression using other data (e.g., principal components, preference pairs)

*BMDP Program Library*

**P4R** Regression analysis for a dependent variable on a set of principal components computed from the independent variables in a stepwise manner determined either by magnitude of eigenvalue or correlations between dependent variable and components, with printing. Options: form of input, more printing, four plots.

**P9M** Scoring based on preference pairs – for each observation construct score as linear combination of variables with coefficients based on expert preference, in stepwise manner. Options: printing, plots, compare results when analysis is repeated for different judges.

| **L8c1b :** | **Parameter estimation in multiple linear regression** *(search also L8c1a)* |
|---|---|

*BMDP Program Library*

**P1R** Performs multiple linear regression and prints standard results. Options: weights, form of input, regression on subsets or groups and test of equality of regression lines, intercept term present or absent, more printing, five plots, save predicted values and residuals.

*SAS Program Library*

**REG** Fits least-squares estimates to linear regression models. Options: weights; parameter estimates, predicted values, residuals, Studentized residuals, confidence limits, hypothesis tests; collinearity diagnostics; influence diagnostics including partial regression leverage plots; Durbin-Watson statistic; hypothesis tests involving multiple dependent variables; parameter estimates subject to linear restriction.

| **L8c1b1 :** | **Parameter estimation in multiple linear regression using raw data** |
|---|---|

*DATAPLOT Interactive System*

**FIT** Fits a nonlinear model (including linear and polynomial models) to data by least squares. Output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication. Options: user-set starting values, number of iterations, accuracy, weighted and robust fits.

*IMSL STAT/LIBRARY Subprogram Library*

**RGIVN** Fit a multivariate linear regression model via fast Givens transformations.

*MINITAB Interactive System*

**REGRESS** Performs simple or multiple linear regression, prints standard results. Options: amount of output, save results, weights, missing values, through the origin, compute and save regression diagnostics, lack of fit tests.

*STARPAC Subprogram Library*

• **LLS** Compute and print a four-part unweighted linear least squares analysis with user-specified model (design matrix); return residuals.

**LLSS** Compute and optionally print a four-part unweighted linear least squares analysis with user-specified model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters.

**LLSW** Compute and print a four-part weighted linear least squares analysis with user-specified model (design matrix); return residuals.

**LLSWS** Compute and optionally print a four-part weighted linear least squares analysis with user-specified model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters.

| **L8c1b2 :** | **Parameter estimation in multiple linear regression using correlation data** |
|---|---|

*IMSL Subprogram Library*

**RLMUL** Multiple linear regression analysis.

• **easy-to-use version**

*IMSL STAT/LIBRARY Subprogram Library*

**RCOV** Fit a multiple linear regression model given the variance-covariance matrix.

*NAG Subprogram Library*

**G02CGF** Performs a multiple linear regression on a set of variables whose means, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients are given.

**G02CHF** Performs a multiple linear regression with no constant on a set of variables whose sums of squares and cross-products about zero and correlation-like coefficients are given.

**L8c1c : Analysis of multiple linear regression results** *(search also classes L8c1a and L8c1b)*

*IMSL Subprogram Library*

**RLFITI** Pure replication error degrees of freedom and sum of squares - in-core version.

**RLFITO** Pure replication error degrees of freedom and sum of squares - out-of-core version.

**RLRES** Perform a residual analysis for a fitted regression model.

*IMSL STAT/LIBRARY Subprogram Library*

**RCASE** Compute case statistics and diagnostics given data points, coefficient estimates, and the R matrix for a fitted general linear model.

**RCOVB** Compute the estimated variance-covariance matrix of the estimated regression coefficients given the R matrix.

**ROTIN** Compute diagnostics for detection of outliers and influential data points given residuals and the R matrix for a fitted general linear model.

**L8c1d : Inference in multiple linear regression** *(search also classes L8c1a and L8c1b)*

*IMSL Subprogram Library*

**RLPRDI** Confidence intervals for the true response and for the average of a set of future observations on the response – in-core version.

**RLPRDO** Confidence intervals for the true response and for the average of a set of future observations on the response – out-of-core version.

*IMSL STAT/LIBRARY Subprogram Library*

**RCASE** Compute case statistics and diagnostics given data points, coefficient estimates, and the R matrix for a fitted general linear model.

**RSTAT** Compute statistics related to a regression fit given the coefficient estimates and the R matrix.

**L8c2 : Several multiple linear regressions**

*BMDP Program Library*

**P1R** Performs multiple linear regression and prints standard results. Options: weights, form of input, regression on subsets or groups and test of equality of regression lines, intercept term present or absent, more printing, five plots, save predicted values and residuals.

*NAG Subprogram Library*

**G02CJF** Performs one or more multiple linear regressions, regressing each of a set of dependent variables separately on the same set of independent variables. Input to the routine is in the form of raw data. Output includes, for each dependent variable, estimates of the regression coefficients, and an estimate of the variance of residuals.

## L8c3 : Multiple linear $L_p$ regression for $p$ different from 2

*Collected Algorithms of the ACM*

**A564** L1GNR: a Fortran subroutine for generating test problems for discrete linear L-sub-1 approximation problems. (See K.L. Hoffman and D.R. Shier, ACM TOMS 6 (1980) pp. 615-617.)

**A615** KBEST: a Fortran subprogram to determine the best subset of parameters to fit a linear regression under a least absolute value criterion. It utilizes the simplex method of linear programming within a branch-and-bound algorithm to solve the best subset problem. (See R.D. Armstrong, P.O. Beck, and M.T. Kung, ACM TOMS 10 (1984) pp. 202-206.)

*DATAPLOT Interactive System*

**PRE-FIT** Fits a nonlinear model (including linear and polynomial models) to data over a specified discrete lattice of parameter values by least squares, least absolute value, and, optionally, other $L_p$ criteria. For the best and worst set parameter values for each criterion, output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication.

*IMSL Subprogram Library*

**RLLAV** Perform linear regression using the least absolute values criterion.

**RLLMV** Perform linear regression using the minimax criterion.

*IMSL STAT/LIBRARY Subprogram Library*

**RLAV** Fit a multiple linear regression model using the least absolute values criterion.

**RLMV** Fit a multiple linear regression model using the minimax criterion.

*Scientific Desk PC Subprogram Library*

**L8CP** Estimates (and estimates the best subset of) the parameters which fit a linear regression under a least absolute value criterion. This program utilizes the simplex method of linear programming within a branch-and-bound algorithm to solve the subset problem.

## L8c4 : Robust multiple linear regression

*ROSEPAK Interactive Program Library*

**IRLS** Robust regression by iteratively reweighted least squares. Weights are determined from the data and are computed as functions of the scaled residuals. Options: $L_1$ and $L_2$ starts, eight weight functions, rank determination, and convergence criterion.

## L8c5 : Measurement error models

*CMLIB Library (ODRPACK Sublibrary)*

- **SODR** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated.

**SODRC** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated.

## L8c6 : Multiple linear regression models based on ranks

*NAG Subprogram Library*

**G08RAF** Calculates the parameter estimates, score statistics and their variance-covariance matrices for the linear model using a likelihood based on the ranks of the observations.

**G08RBF** Calculates the parameter estimates, score statistics and their variance-covariance matrices for the linear model using a likelihood based on the ranks of the observations when some of the observations may be right censored.

## L8d : Polynomial regression in several variables

*IMSL Subprogram Library*

**RLCOMP** Generation of an orthogonal central composite design.

*IMSL STAT/LIBRARY Subprogram Library*

**RCOMP** Generate an orthogonal central composite design.

*SAS Program Library*

**RSREG** Estimates a quadratic response surface using least-squares regression and determines critical values to optimize the response. Options: weights, lack of fit test, surface plotting, eigenvalues of the associated quadratic form.

## L8e1 : Nonlinear least squares regression (i.e., $y = F(X, b)$)

*IMSL STAT/LIBRARY Subprogram Library*

**RNLIN** Fit a nonlinear regression model.

*INVAR Interactive Program Library*

**INVAR1** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results. Line printer graphics only.

**INVAR2** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results and DISSPLA graphics.

*SAS Program Library*

**NLIN** Performs nonlinear least-squares regression using one of four iterative methods (modified Gauss-Newton, Marquardt, gradient, or steepest-descent, and multivariate secant or false position (DUD). User provides starting values for the parameters, and derivatives of the model for all but the DUD method. Options: weights, bounds on the parameter estimates, objective function to be minimized, grid search for starting values.

| L8e1a : | Variable selection in nonlinear regression |
|---|---|

*BMDP Program Library*

**PLR** Stepwise logistic regression for binary dependent variable and categorical (design variables are formed) and continuous independent variables, using either maximum likelihood or approximate asymptotic estimates for stepping. Three options for generating design variables, plots, interactive stepping.

*SAS Program Library*

**PROBIT** Calculates maximum-likelihood estimates of the intercept, slope and natural (threshold) response rate for biological assay data using a modified Gauss-Newton algorithm.

*SPSS Program Library*

**PROBIT** Estimates the effects of one or more independent variables on a dichotomous dependent variable by maximum likelihood. Options: form of response model (probit and/or logit), logarithmic transformation, missing values.

| L8e1b1 : | Parameter estimation in nonlinear least squares regression using unweighted data, user provides no derivatives |
|---|---|

*BMDP Program Library*

**PAR** Performs nonlinear regression using pseudo-Gauss-Newton algorithm. Derivatives are NOT specified. Options: weights, linear inequality constraints, maximum likelihood, functions of parameters, ridging, four plots, fitting models defined by differential equations.

*CMLIB Library (NL2SN Sublibrary)*

**NL2SN** Minimizes a nonlinear sum of squares using residual values only.

*DATAPLOT Interactive System*

**FIT** Fits a nonlinear model (including linear and polynomial models) to data by least squares. Output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication. Options: user-set starting values, number of iterations, accuracy, weighted and robust fits.

**PRE-FIT** Fits a nonlinear model (including linear and polynomial models) to data over a specified discrete lattice of parameter values by least squares, least absolute value, and, optionally, other $L_p$ criteria. For the best and worst set parameter values for each criterion, output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication.

*IMSL Subprogram Library*

**RSMITZ** Least squares fit of the non-linear regression model $y_i = \alpha + \beta\gamma^{x_i} + e_i$.

**ZXSSQ** Minimum of the sum of squares of m functions in n variables using a finite difference Levenberg-Marquardt algorithm.

*STARPAC Subprogram Library*

• **NLS** Unweighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values.

**NLSC** Unweighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user.

• **easy-to-use version**

**NLSS** Unweighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user.

**L8e1b2 : Parameter estimation in nonlinear least squares regression using unweighted data, user provides derivatives**

*BMDP Program Library*

**P3R** Performs nonlinear least squares regression with standard results. Six functions are built in; others can be specified. Options: weights, evaluates functions of parameters (with standard errors), upper and lower limits on parameters, ridging, exact linear constraints, maximum likelihood estimates, and five plots.

*CMLIB Library (NL2SN Sublibrary)*

**NL2S1** Minimizes a nonlinear sum of squares using both residual and gradient values supplied by the user.

*STARPAC Subprogram Library*

• **NLSD** Unweighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values.

**NLSDC** Unweighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user.

**NLSDS** Unweighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user.

**L8e1b3 : Parameter estimation in nonlinear least squares regression using weighted data, user provides no derivatives**

*BMDP Program Library*

**PAR** Performs nonlinear regression using pseudo-Gauss-Newton algorithm. Derivatives are NOT specified. Options: weights, linear inequality constraints, maximum likelihood, functions of parameters, ridging, four plots, fitting models defined by differential equations.

*CMLIB Library (NL2SN Sublibrary)*

**NL2SN** Minimizes a nonlinear sum of squares using residual values only.

*IMSL Subprogram Library*

**ZXSSQ** Minimum of the sum of squares of m functions in n variables using a finite difference Levenberg-Marquardt algorithm.

*STARPAC Subprogram Library*

• **NLSW** Weighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values.

**NLSWC** Weighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user.

**NLSWS** Weighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user.

| L8e1b4 : | Parameter estimation in nonlinear least squares regression using weighted data, user provides derivatives |
|---|---|

*BMDP Program Library*

**P3R** Performs nonlinear least squares regression with standard results. Six functions are built in; others can be specified. Options: weights, evaluates functions of parameters (with standard errors), upper and lower limits on parameters, ridging, exact linear constraints, maximum likelihood estimates, and five plots.

*CMLIB Library (NL2SN Sublibrary)*

**NL2S1** Minimizes a nonlinear sum of squares using both residual and gradient values supplied by the user.

*STARPAC Subprogram Library*

• **NLSWD** Weighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values.

**NLSWDC** Weighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user.

**NLSWDS** Weighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user.

| L8e2 : | Ridge regression |
|---|---|

*BMDP Program Library*

**P3R** Performs nonlinear least squares regression with standard results. Six functions are built in; others can be specified. Options: weights, evaluates functions of parameters (with standard errors), upper and lower limits on parameters, ridging, exact linear constraints, maximum likelihood estimates, and five plots.

**PAR** Performs nonlinear regression using pseudo-Gauss-Newton algorithm. Derivatives are NOT specified. Options: weights, linear inequality constraints, maximum likelihood, functions of parameters, ridging, four plots, fitting models defined by differential equations.

| L8e5 : | Nonlinear measurement error models |
|---|---|

*CMLIB Library (ODRPACK Sublibrary)*

• **SODR** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated.

**SODRC** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated.

• **easy-to-use version**

## L8f : Simultaneous regression

*BMDP Program Library*

**P6R** Computes the partial correlations of a set of variables after removing the linear effects of a second set of variables. Can be used for regression, especially if multiple dependent variables are present. Prints standard results. Options: weights, form of input, additional printing and plots.

*Collected Algorithms of the ACM*

**A633** LDA: a Fortran subroutine for linear dependency analysis of multivariate data. (See R.C. Ward, G.J. Davis, and V.E. Kane, ACM TOMS 11 (1985) pp. 170-182.)

*IMSL Subprogram Library*

**OFIMA3** Least squares solution to the matrix equation AT = B.

*IMSL STAT/LIBRARY Subprogram Library*

**RCOV** Fit a multiple linear regression model given the variance-covariance matrix.

**RGIVN** Fit a multivariate linear regression model via fast Givens transformations.

**RGLM** Fit a multivariate general linear model.

**RLEQU** Fit a multivariate linear regression model with linear equality restrictions $H\beta = G$ imposed on the regression parameters given results from IMSL routine RGIVN after IDO = 1 and IDO = 2 and prior to IDO = 3.

## L8g : Spline (i.e., piecewise polynomial) regression

*Collected Algorithms of the ACM*

**A642** CUBGCV: a Fortran subprogram for the fast O(n) computation of a cubic smoothing spline fitted to n noisy data points, with the degree of smoothing chosen to minimize the expected mean square error at the data points when the variance of the error in the data is known, or the generalized cross validation when it is unknown. The data may be unequally spaced and nonuniformly weighted. Bayesian point error estimates are also calculated. (See M.F. Hutchinson, ACM TOMS 12 (1986) pp. 150-153.)

*CMLIB Library (FC Sublibrary)*

**FC** Fits piecewise polynomial to discrete data with equality and inequality constraints.

*DATAPLOT Interactive System*

**SPLINE FIT** Performs a least squares spline fit. The user can specify both the degree of the piecewise polynomial (linear through tenth degree, default is cubic) and the knot points (i.e., the regions where distinct polynomials are fit). Both the predicted and the residual values are available for further analysis and plotting.

## L8h : EDA regression

*DATAPLOT Interactive System*

**LOWESS SMOOTH** Perform a locally weighted least squares analysis of a two-dimensional data set. Output includes predicted and residual values. The user can specify the smoothing interval.

**SMOOTH** Perform either a least squares smooth or a robust smooth (Tukey's 3RSR smooth). Both the degree (zero to ten) and the width (the odd number of equally spaced points to use at each step) of the smooth can be specified. A moving average smooth is a degree one smooth. Output includes the predicted and residual values.

*MINITAB Interactive System*

**RLINE** Fits straight line to x-y data by resistant line procedure – partitions data by x-value into three groups and uses an iterative procedure to find the line that makes the median residual in the left and the right partitions equal.

**RSMOOTH** Computes resistant smoother by 4253H, twice (or 3RSSH, twice), i.e. successive application of running medians and Hanning (running weighted averages), and save results.

*Scientific Desk PC Subprogram Library*

**L8G1A** Determines an $r$ for the equation $Y_i = b_0 + b_1 X_i^R$ so that $Y_i = b_0 + b_1 Z_i$, where $Z_i = X_i^r$ plots as a straight line. The iteration scheme used for R is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $r$.

**L8G1E** Determines an $a$ for the equation $Y_i = b_0 + b_1 a^{X_i}$ so that $Y_i = b_0 + b_1 Z_i$, where $Z_i = a^{X_i}$ plots as a straight line. The iteration scheme used for R is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $a$.

**L8i : Service routines for regression (e.g., matrix manipulation for variable selection)**

*IMSL Subprogram Library*

**RLSUBM** Retrieval of a symmetric submatrix from a matrix stored in symmetric storage mode by RLSTP.

**RLSUM** Reordering of the rows and corresponding columns of a symmetric matrix stored in symmetric storage mode.

**USLEAP** Print results of the best-regressions analysis performed by IMSL routine RLEAP.

*IMSL STAT/LIBRARY Subprogram Library*

**RORDM** Reorder rows and columns of a symmetric matrix.

*NAG Subprogram Library*

**G02CEF** Takes selected elements from two vectors (typically vectors of means and standard deviations) to form two smaller vectors, and selected rows and columns from two matrices (typically either matrices of sums of squares and cross-products of deviations from means and Pearson product-moment correlation coefficients, or matrices of sums of squares and cross-products about zero and correlation-like coefficients) to form two smaller matrices, allowing re-ordering of elements in the process.

**G02CFF** Re-orders the elements in two vectors (typically vectors of means and standard deviations), and the rows and columns in two matrices (typically either matrices of sums of squares and cross-products of deviations from means and Pearson product-moment correlation coefficients, or matrices of sums of squares and cross-products about zero and correlation-like coefficients).

*STARPAC Subprogram Library*

**DCKLS** Check user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values.

**DCKLSC** Check user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user.

**STPLS** Select optimum step size for computing finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values.

**STPLSC** Select optimum step size for computing finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user.

## L9: Categorical Data Analysis

The software for the analysis of categorical data, also known as frequency tables or contingency tables, is organized by the size of the data set being analyzed. In a two-by-two table or matrix, the two rows in the matrix represent two categories of a variable, the two columns represent two categories of a different variable, and the entries are counts of the numbers of observations in those categories. (In analysis of variance, by contrast, the entries are observations themselves.) In two-way tables a two-dimensional matrix is of arbitrary size. A natural question in both two-by-two and two-way analyses is whether the two variables are independent.

Use of the log-linear model allows analysis of data with more than two categorical variables, including tables with "structural zeros" (i.e., categories in which the count is necessarily zero). Log-linear model software can be used to estimate the form of the model through a stepwise analysis of a hierarchy of models. The underlying distribution of the observations is most commonly assumed to be multinomial; if distributional assumptions are not appropriate then the median polish, which uses the median instead of the mean as a measure of central tendency, may be more appropriate. Graphical displays of parameter estimates and of residuals are useful in analyzing the appropriateness of the fitted model.

## References

[BFh75] Y. M. M. Bishop, S. E. Fienberg, and P. W. Holland. *The Analysis of Cross-Classified Categorical Data.* MIT Press, Cambridge, Massachusetts, 1975.

[Fie77] Fienberg. S. E. *The Analysis of Cross-Classified Categorical Data.* MIT Press, Cambridge, Massachusetts, 1977.

[Pla81] R. L. Plackett. *The Analysis of Categorical Data.* Macmillan, New York, 2nd edition, 1981.

[Tuk77] J. W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, Reading, MA, 1977.

**L9 : Categorical data analysis**

*BMDP Program Library*

**P4F** Forms two- or multi-way frequency tables. Options: percents; 25 tests and measures for two-way tables; fits and tests log-linear models, tests of marginal and partial association, stepwise models, three forms of input, structural zeros, cell and strata deletion, residuals.

*MINITAB Interactive System*

**TABLE** Produces and prints one-way, two-way, and multi-way tables of counts with 20 optional subcommands for summarizing (e.g., cell mean, standard deviation), marginals, performing chi-square tests for each 2-way table, handling missing values, and selecting forms of input and output.

*SAS Program Library*

**FREQ** Builds frequency or crosstabulation tables for one-way to n-way categorical data. Can compute tests and measures of association for two-way tables and can do stratified analysis and compute statistics within as well as across strata for n-way tables. Options: missing values, weights, additional analysis.

*SPSS Program Library*

**CROSSTABS** Produces tables that are the joint distribution of two or more variables that have a limited number of distinct values. Handles integer, continuous, and character data. Options: summary statistics, several measures of association, missing values.

## L9a : Analysis of 2-by-2 tables of categorical data

*IMSL Subprogram Library*

**NHEXT** Fishers exact method for 2-by-2 tables.

*IMSL STAT/LIBRARY Subprogram Library*

**CTTWO** Perform a chi-squared analysis of a 2-by-2 contingency table.

## L9b : Analysis of two-way tables of categorical data *(search also class L9d)*

*Collected Algorithms of the ACM*

**A643** FEXACT: a Fortran subprogram for Fisher's exact test on unordered r-by-c contingency tables. (See C.R. Mehta and N.R. Patel, ACM TOMS 12 (1986) pp. 154-161.)

*IMSL Subprogram Library*

**BDTWT** Computations of a two-way frequency table.
**CBNRHO** Estimation of the bivariate normal correlation coefficient using a contingency table.
**CTPR** Compute exact probabilities for contingency tables.
**CTRBYC** Analysis of a contingency table.

*IMSL STAT/LIBRARY Subprogram Library*

**CTCHI** Perform a chi-squared analysis of a two-way contingency table.
**CTPRB** Compute exact probabilities in a two-way contingency table.

*MINITAB Interactive System*

**CHISQUARE** Performs chi-square test for association (non-independence) on a two-way table and prints standard results.

*NAG Subprogram Library*

**G01AFF** Performs the analysis of a two-way rc contingency table or classification. If r=c=2, and the total number of objects classified is 40 or fewer, then the probabilities for Fisher's exact test are computed. Otherwise, a test statistic is computed (with Yates' correction when r=c=2), which under the assumption of no association between the classifications has approximately a chi-square distribution with (r-1)(c-1) degrees of freedom.

*SAS Program Library*

**CATMOD** Analyzes two-dimensional contingency tables by fitting linear models to functions of response frequencies using a maximum-likelihood estimation of parameters for log-linear models and the analysis of generalized logits, or using a weighted-least-squares estimation of parameters for general linear models. Options: weights, parameter testing.

| L9c : | Categorical data analysis using the log-linear model |
|---|---|

*IMSL Subprogram Library*

**CTLLF** Log-linear fit of contingency table.

*IMSL STAT/LIBRARY Subprogram Library*

**PRPFT** Perform iterative proportional fitting of a contingency table using a loglinear model.

*SAS Program Library*

**CATMOD** Analyzes two-dimensional contingency tables by fitting linear models to functions of response frequencies using a maximum-likelihood estimation of parameters for log-linear models and the analysis of generalized logits, or using a weighted-least-squares estimation of parameters for general linear models. Options: weights, parameter testing.

*SPSS Program Library*

**HILOGLINEAR** Estimates parameters of hierarchical log-linear models for frequency tables by iterative proportional fitting. Options: backward elimination, cell weights, plots, algorithm tuning parameters, plots, missing values.

**LOGLINEAR** Models cell frequencies using multinomial response model and produces maximum likelihood estimates of parameters by means of the Newton-Raphson algorithm. Options: cell weights, plots, linear combinations, contrasts, algorithm tuning parameters, missing values.

| L9d : | EDA analysis of categorical data |
|---|---|

*DATAPLOT Interactive System*

**MEDIAN POLISH** Performs a 1-5 way median polish on balanced data. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviations, residual degrees of freedom, as well as replication standard deviation, replication degrees of freedom, and lack-of-fit F-statistic if there is replication.

*IMSL Subprogram Library*

**BEMDP** Median polish of a two-way table.

*IMSL STAT/LIBRARY Subprogram Library*

**MEDPL** Compute a median polish of a two-way table.

*MINITAB Interactive System*

**MPOLISH** Uses median polish to fit an additive model to a two-way layout which may be unbalanced and may have empty cells. Options: fit columns first, number of iterations, save results.

## L10: Time Series Analysis

Software for time series analysis is organized by the number and dimension of the series, beginning with the univariate series, i.e., one series of one-dimensional observations, then bivariate time series (two time series of one-dimensional observations), followed by series with multi-dimensional observations. In all cases, however, some preprocessing of what are effectively single one-dimensional time series may be needed (also see class L1a). Most analyses assume that the series are stationary; in first-order stationarity the expected mean value of the errors is constant for all terms in the series, and second-order stationarity also assumes constant variance.

Two widely used approaches to times series analysis are Box-Jenkins methods and spectral analysis. The Box-Jenkins approach seeks to build models with autoregressive terms (to model the "memory" of the underlying system) and moving average terms (to model the random effects). The method seems most appropriate when the source of the time series data is a structured system subject to random forcings.

Spectral analysis, which had its origin in the work of Fourier, seeks to transform a given time series to its representation in terms of frequencies. Assuming that the series can be modelled as a superposition of oscillations with the appropriate frequencies and amplitudes, spectral analysis seeks to isolate those characteristics. The method is most appropriate for series with strong deterministic components as well as noise.

The two methods are not mutually exclusive, and in fact share a number of techniques (e.g., autocorrelation analysis). Spectral analysis is more commonly used than Box-Jenkins methods in physical science and engineering applications.

## References

[And70] T. W. Anderson. *The Statistical Analysis of Time Series.* John Wiley & Sons, New York, 1970.

[BD87] P. Brockwell and R. Davis. *Time Series: Theory and Methods.* Springer-Verlag, New York, 1987.

[BJ76] G. E. Box and G. M. Jenkins. *Time Series Analysis: Forecasting and Control.* Holden-Day, San Francisco, 1976.

[BK83] D. R. Brillinger and P. R. Krishnaiah, editors. *Time Series in the Frequency Domain*, volume 3 of *Handbook of Statistics.* North-Holland, Amsterdam, 1983.

[Blo76] P. Bloomfield. *Fourier Analysis of Time Series: An Introduction.* John Wiley & Sons, New York, 1976.

[BP80] J. S. Bendat and A. G. Piersol. *Engineering Applications of Correlation and Spectral Analysis.* John Wiley & Sons, New York, 1980.

[Bra78] R. N. Bracewell. *The Fourier Transform and Its Applications.* McGraw-Hill, New York, 2nd edition, 1978.

[Bri75] D. R. Brillinger. *Time Series: Data Analysis and Theory.* Holt, Rinehart and Winston, New York, 1975.

[BT58] R. B. Blackman and J. W. Tukey. *The Measurement of Power Spectra.* Dover, New York, 1958.

[Cha75] C. Chatfield. *The Analysis of Time Series: Theory and Practice.* Chapman and Hall, London, 1975.

[Gra77] C. W. J. Granger. *Forecasting Economic Time Series.* Academic Press, New York, 1977.

[Han70] E. J. Hannan. *Multiple Time Series.* John Wiley & Sons, New York, 1970.

[HKR85] E. J. Hannan, P. R. Krishnaiah, and M. M. Rao, editors. *Time Series in the Time Domain*, volume 5 of *Handbook of Statistics.* North-Holland, Amsterdam, 1985.

[JW68] G. M. Jenkins and D. G. Watts. *Spectral Analysis and its Applications.* Holden-Day Inc., San Francisco, 1968.

[Pan83] A. Pankratz. *Forecasting with Univariate Box-Jenkins Models: Concepts and Cases.* John Wiley & Sons, New York, 1983.

[Pri81a] M. B. Priestley. *Univariate Series*, volume 1 of *Spectral Analysis and Time Series.* Academic Press, London, 1981.

[Pri81b] M. B. Priestley. *Multivariate Series, Predicition and Control*, volume 2 of *Spectral Analysis and Time Series.* Academic Press, London, 1981.

[PW83] S. M. Pandit and S. M. Wu. *Time Series and System Analysis with Applications.* John Wiley & Sons, New York, 1983.

| **L10a1a :** | **Elementary time series transformations** *(search also class L2a)* |
|---|---|

*MINITAB Interactive System*

**LAG** Computes lagged observations in a time series.

*STARPAC Subprogram Library*

**SAMPLE** Sample (extract) every NSth observation from a series; return sampled series (no printed output).

| **L10a1b :** | **Stationarity transformations** *(search also class L8a1)* |
|---|---|

*IMSL Subprogram Library*

**FTRDIF** Transformations, differences and seasonal differences of a time series for model identification.

*IMSL STAT/LIBRARY Subprogram Library*

**BCTR** Perform a forward or an inverse Box-Cox (power) transformation.

| **L10a1c1 :** | **Difference filters** |
|---|---|

*IMSL Subprogram Library*

**FTRDIF** Transformations, differences and seasonal differences of a time series for model identification.

*IMSL STAT/LIBRARY Subprogram Library*

**DIFF** Difference a time series.

*MINITAB Interactive System*

**DIFFERENCES** Computes differences between observations at a specified lag in a time series.

*NAG Subprogram Library*

**G13AAF** Carries out non-seasonal and seasonal differencing on a time series. Information which allows the original series to be reconstituted from the differenced series is also produced. This information is required in time series forecasting.

*STARPAC Subprogram Library*

• **DIF** Perform first-difference filter operation; return differenced series (no printed output).

**DIFM** Perform first-difference filter operation on series with missing data; return differenced series (no printed output).

## L10a1c2 : Symmetric linear filters (e.g., moving averages)

*DATAPLOT Interactive System*

**SMOOTH** Perform either a least squares smooth or a robust smooth (Tukey's 3RSR smooth). Both the degree (zero to ten) and the width (the odd number of equally spaced points to use at each step) of the smooth can be specified. A moving average smooth is a degree one smooth. Output includes the predicted and residual values.

*STARPAC Subprogram Library*

• **GFSLF** Compute and plot gain function of symmetric linear filter.

**GFSLFS** Compute and optionally plot gain function of symmetric linear filter with user-supplied control values; return gain function values and corresponding frequency values.

**HIPASS** Filter series with symmetric linear high-pass filter; return filtered series (no printed output).

**HPCOEF** Compute symmetric linear high-pass filter coefficients; return coefficients (no printed output).

**LOPASS** Filter series with symmetric linear low-pass filter; return filtered series (no printed output).

**LPCOEF** Compute symmetric linear low-pass filter coefficients; return filter coefficients (no printed output).

**MAFLT** Perform simple moving average; return filtered series (no printed output).

**SLFLT** Perform symmetric linear filter operation with user-supplied filter coefficients; return filtered series (no printed output).

## L10a1c3 : Autoregressive linear filters

*STARPAC Subprogram Library*

**ARFLT** Perform autoregressive filter operation with user-supplied filter coefficients; return filtered series (no printed output).

• **DIFC** Perform user-specified difference filter operation; return differenced series (no printed output).

**DIFMC** Perform user-specified difference filter operation on series with missing data; return differenced series (no printed output).

• **GFARF** Compute and plot gain and phase functions of autoregressive or difference filter.

**GFARFS** Compute and optionally plot gain and phase functions of autoregressive or difference filter; with user-supplied control values; return gain and phase function values and corresponding frequency values.

## L10a1c4 : Other time series filters

*NAG Subprogram Library*

**G13BAF** Filters a time series by an ARIMA model.

**G13BBF** Filters a time series by a transfer function model.

• **easy-to-use version**

**L10a1d : Taper**

*STARPAC Subprogram Library*

**TAPER** Center a series about its mean and apply a split-cosine-bell taper; return the tapered series (no printed output).

**L10a2 : Time domain time series analysis**

*BMDP Program Library*

**P2T** Interactive or batch Box-Jenkins time series analysis for univariate time domain models (including ARIMA, regression, intervention, and transfer function models) – model identification, parameter estimation, testing, forecasting. Options: print, plot, differencing and filtering, save results.

*IMSL Subprogram Library*

**FTFREQ** Single or multichannel time series analysis in the time and frequency domains.

**L10a2a : Time series summary statistics**

*IMSL Subprogram Library*

**FTAUTO** Mean, variance, autocovariances, autocorrelations, and partial autocorrelations for a stationary time series.

*STARPAC Subprogram Library*

• **ACF** Compute and print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model.

**ACFD** Compute and print a two-part auto- and partial correlation analysis of a sequence of differenced series, select the order of an autoregressive process which models each series, and estimate the parameters of these models.

**ACFF** Compute and print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use FFT for computations.

**ACFFS** Compute and optionally print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use FFT for computations; use user-supplied control values; return autocovariance function, and order and parameter estimates of selected autoregressive model.

**ACFM** Compute and print a two-part auto- and partial correlation analysis of a series with missing observations.

**ACFMS** Compute and optionally print a two-part auto- and partial correlation analysis of a series with missing observations; use user-supplied control values; return autocovariance function.

**ACFS** Compute and optionally print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use user-supplied control values; return autocovariance function, and order and parameter estimates of selected autoregressive model.

**L10a2a1 : Autocovariances and autocorrelations**

*DATAPAC Subprogram Library*

**AUTOCO** Computes the sample autocorrelation coefficient of the data in the input vector X.

*DATAPLOT Interactive System*

**CORRELATION PLOT** Plots autocorrelations vs. lag for equi-spaced univariate time series data or plots the cross-correlation coefficient vs. lag for equi-spaced bivariate time series data.

*IMSL STAT/LIBRARY Subprogram Library*

**ACF** Compute the sample autocorrelation function of a stationary time series.

*MINITAB Interactive System*

**ACF** Computes and graphs the autocorrelations of a time series, and optionally saves results.

*NAG Subprogram Library*

**G13ABF** Computes the sample autocorrelation function of a time series. It also computes the sample mean, the sample variance and a statistic which may be used to test the hypothesis that the true autocorrelation function is zero.

**L10a2a2 : Partial autocorrelations**

*IMSL STAT/LIBRARY Subprogram Library*

**PACF** Compute the sample partial autocorrelation function of a stationary time series.

*MINITAB Interactive System*

**PACF** Computes and graphs partial autocorrelations of a time series and optionally saves results.

*NAG Subprogram Library*

**G13ACF** Calculates partial autocorrelation coefficients given a set of autocorrelation coefficients. It also calculates the predictor error variance ratios for increasing order of finite lag autoregressive predictor, and the autoregressive parameters associated with the predictor of maximum order.

**L10a2b : Stationarity analysis** *(search also class L10a2a)*

*DATAPLOT Interactive System*

**CONTROL CHART** Plots a specified statistic (mean (default), standard deviation, or range) for replicated data as a function of time or replication group number. Output contains computed statistics, a typical value line and upper and lower control lines.

**L10a2c : Autoregressive models**

*IMSL Subprogram Library*

**FTWEIN** Wiener forecast for a stationary stochastic process.

*IMSL STAT/LIBRARY Subprogram Library*

**SPWF** Compute the Wiener forecast operator for a stationary stochastic process.

*TIMSAC Program Library*

**PERARS** Fits periodic autoregressive models by the method of least squares realized through the Householder transformation. The outputs are the estimates of the regression coefficients and innovation variance of the periodic AR-model for each instant.

## L10a2c1 : Autoregressive model identification

*NAG Subprogram Library*

**G13ACF** Calculates partial autocorrelation coefficients given a set of autocorrelation coefficients. It also calculates the predictor error variance ratios for increasing order of finite lag autoregressive predictor, and the autoregressive parameters associated with the predictor of maximum order.

*STARPAC Subprogram Library*

• **ACF** Compute and print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model.

**ACFD** Compute and print a two-part auto- and partial correlation analysis of a sequence of differenced series, select the order of an autoregressive process which models each series, and estimate the parameters of these models.

**ACFF** Compute and print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use FFT for computations.

**ACFFS** Compute and optionally print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use FFT for computations; use user-supplied control values; return autocovariance function, and order and parameter estimates of selected autoregressive model.

**ACFM** Compute and print a two-part auto- and partial correlation analysis of a series with missing observations.

**ACFMS** Compute and optionally print a two-part auto- and partial correlation analysis of a series with missing observations; use user-supplied control values; return autocovariance function.

**ACFS** Compute and optionally print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use user-supplied control values; return autocovariance function, and order and parameter estimates of selected autoregressive model.

*TIMSAC Program Library*

**BSUBST** Produces Bayesian estimates of time series models such as pure AR models, AR models with non-linear terms, AR models with polynomial type mean value functions, etc. The goodness of fit of a model is checked by the analysis of several steps ahead prediction errors. By preparing an external subroutine SETX, any time series model which is linear in its parameters can be analysed.

**UNIMAR** A basic program for fitting of autoregressive models of successively higher orders by the method of least squares realized through Householder transformation. The outputs are the estimates of the coefficients, the innovation variances and the corresponding AIC statistics.

| L10a2c2 : | Autoregressive model parameter estimation |
|---|---|

*TIMSAC Program Library*

**EXSAR** Produces exact maximum likelihood estimates of the parameters of a scalar AR-model.

**UNIBAR** Fits an autoregressive model by a Bayesian procedure. The least squares estimates of the parameters are obtained by the Householder transformation.

| L10a2d : | ARMA and ARIMA models (Box-Jenkins methods) |
|---|---|

*SPSS Program Library*

**BOX-JENKINS** Box-Jenkins modeling and forecasting of time series data. Options: log or power transformation, seasonal or nonseasonal differencing, parameter selection (autoregressive and/or moving average, seasonal or nonseasonal), control of estimation.

| L10a2d1 : | ARMA and ARIMA model identification |
|---|---|

*IMSL Subprogram Library*

**FTARPS** Preliminary estimation of the autoregressive parameters in an ARIMA stochastic model.

**FTMA** Preliminary estimation of the moving average parameters in an ARIMA stochastic model.

*IMSL STAT/LIBRARY Subprogram Library*

**NSPE** Compute preliminary estimates of the autoregressive and moving average parameters of an ARMA model.

*NAG Subprogram Library*

**G13ADF** Calculates preliminary estimates of the parameters of an autoregressive integrated moving average (ARIMA) model from the autocorrelation function of the appropriately differenced time series.

| L10a2d2 : | ARMA and ARIMA model parameter estimation |
|---|---|

*IMSL Subprogram Library*

**FTCP** Non-seasonal ARIMA (Box-Jenkins) stochastic model analysis for a single time series with full parameter iteration and maximum likelihood estimation.

**FTML** Maximum likelihood estimation of autoregressive and moving average parameters in an ARIMA (Box-Jenkins) stochastic model.

*IMSL STAT/LIBRARY Subprogram Library*

**ARMME** Compute method of moments estimates of the autoregressive parameters of an ARMA model.

**MAMME** Compute method of moments estimates of the moving average parameters of an ARMA model.

**NSLSE** Compute least squares estimates of parameters for a nonseasonal ARMA model.

*MINITAB Interactive System*

**ARIMA** Fits non-seasonal and seasonal models to a time series with p the order of the AR part, d the number of differences, q the order of the MA part and with optional seasonality with period S, AR order P, number of differences D, and MA order Q. Options: starting values, forecasting, save results.

*NAG Subprogram Library*

**G13AEF** Fits a seasonal autoregressive-integrated moving average (ARIMA) model to an observed time series, using a non-linear least squares procedure incorporating backforecasting. Parameter estimates are obtained, together with appropriate standard errors. The residual series is returned, and information for use in forecasting the time series is produced for use by the routines G13AGF and G13AHF.

• **G13AFF** Is an easy-to-use version of G13AEF. It fits a seasonal autoregressive integrated moving average (ARIMA) model to an observed time series, using a non-linear least squares procedure incorporating backforecasting. Parameter estimates are obtained, together with appropriate standard errors. The residual series is returned, and information for use in forecasting the time series is produced for use in the routines G13AGF and G13AHF.

**G13BEF** Estimates the parameters in a multi-input model relating one output series to one or more input series, using a choice of non-linear least squares, exact likelihood, or marginal likelihood for estimation criteria. If no input series are present, estimates a univariate ARIMA model.

*STARPAC Subprogram Library*

• **AIME** Compute and print a five-part least squares analysis of the parameter estimates of an ARIMA model; return parameter estimates and residuals.

**AIMEC** Compute and optionally print a five-part least squares analysis of the parameter estimates of an ARIMA model using user-supplied control values; return parameter estimates and residuals.

**AIMES** Compute and optionally print a five-part least squares analysis of the parameter estimates of an ARIMA model using user-supplied control values; return parameter estimates, residuals, number of parameters estimated, residual standard deviation, predicted values, standard deviations of the predicted values and variance-covariance matrix of the estimated parameters.

*TIMSAC Program Library*

**XSARMA** Produces exact maximum likelihood estimates of the parameters of a scalar ARMA model.

## L10a2d3 : ARMA and ARIMA model forecasting

*IMSL Subprogram Library*

**FTCAST** Time series forecasts and probability limits using an ARIMA (Box-Jenkins) model.

*IMSL STAT/LIBRARY Subprogram Library*

**NSBJF** Compute Box-Jenkins forecasts and their associated probability limits for a nonseasonal ARMA model.

*MINITAB Interactive System*

**ARIMA** Fits non-seasonal and seasonal models to a time series with p the order of the AR part, d the number of differences, q the order of the MA part and with optional seasonality with period S, AR order P, number of differences D, and MA order Q. Options: starting values, forecasting, save results.

*NAG Subprogram Library*

**G13AGF** Accepts a series of new observations of a time series, the model of which is already fully specified, and updates the state set information for use in constructing further forecasts. The previous specifications of the time series model should have been obtained by using G13AEF or G13AFF to estimate the relevant parameters. The supplied state set will originally have been produced by G13AEF or G13AFF, but may since have been updated by earlier calls to G13AGF.

**G13AHF** Produces forecasts of a time series, given a time series model which has already been fitted to the time series using routine G13AEF or G13AFF. The original observations are not required, since G13AHF uses as input either the original state set produced by G13AEF or G13AFF or the state set updated by a series of new observations using G13AGF. Standard errors of the forecasts are also provided.

**G13AJF** Applies a fully specified seasonal ARIMA model to an observed time series, generates the state set for forecasting and (optionally) derives a specified number of forecasts together with their standard deviations.

*STARPAC Subprogram Library*

• **AIMF** Compute and print the minimum mean square error forecasts obtained using an ARIMA model.

**AIMFS** Compute and optionally print the minimum mean square error forecasts obtained using an ARIMA model; return forecasts and their standard errors.

## L10a2e : State-space time series analysis

*IMSL Subprogram Library*

**FTKALM** Kalman filtering.

*IMSL STAT/LIBRARY Subprogram Library*

**KALMN** Perform Kalman filtering and evaluate the likelihood function for the state-space model.

## L10a2f : Analysis of a locally stationary series

*TIMSAC Program Library*

**BLOCAR** Locally fits autoregressive models to non-stationary time series by a Bayesian procedure. Power spectra for stationary spans are graphically printed out. (This program is tentative and further testing and improvement through practical applications will be necessary.)

**MLOCAR** Locally fits autoregressive models to non-stationary time series by minimum AIC procedure.

## L10a3 : Frequency domain time series analysis *(search also class J1)*

*BMDP Program Library*

**P1T** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), missing values, remove seasonal means and linear trend, filtering, save results.

*DATAPLOT Interactive System*

**ALLAN STANDARD DEVIATION PLOT** Generate an Allan standard deviation plot with horizontal axis the subsample size and vertical axis the computed Allan standard deviation for the subsample size.

**ALLAN VARIANCE PLOT** Generate an Allan variance plot with horizontal axis the subsample size and vertical axis the computed Allan variance for the subsample size.

*IMSL Subprogram Library*

**FTFREQ** Single or multichannel time series analysis in the time and frequency domains.

**L10a3a1 : Pilot spectral analysis**

*DATAPAC Subprogram Library*

**TIME** Performs a time series analysis on the data in the input vector X.

**L10a3a2 : Periodogram analysis**

*DATAPAC Subprogram Library*

**FOURIE** Performs a Fourier analysis of the data in the input vector X.

*DATAPLOT Interactive System*

**PERIODOGRAM** Generates a frequency vs. power plot for equi-spaced univariate time series data. Alternate names: HARMONIC PLOT, FOURIER PLOT.

*IMSL STAT/LIBRARY Subprogram Library*

**PFFT** Compute the periodogram of a stationary time series using a fast Fourier transform.

*STARPAC Subprogram Library*

**IPGM** Compute and print an integrated periodogram analysis of a series; use FFT for computations.

**IPGMP** Compute and print an integrated periodogram analysis of a series; input periodogram rather than original series.

**IPGMPS** Compute and optionally print an integrated periodogram analysis of a series; input periodogram rather than original series; return integrated periodogram and corresponding frequencies.

**IPGMS** Compute and optionally print an integrated periodogram analysis of a series; use FFT for computations; return integrated periodogram and corresponding frequencies.

**MDFLT** Smooth a periodogram by applying a sequence of modified Daniell filters; return the smoothed periodogram (no printed output).

**PGM** Compute and print a periodogram analysis of a series; use FFT for computations.

**PGMS** Compute and optionally print a periodogram analysis of a series; use FFT for computations; return periodogram and corresponding frequencies.

**L10a3a3 : Spectrum estimation using the periodogram**

*IMSL Subprogram Library*

**FTFPS** Fast Fourier transform estimates of power spectra and cross spectra of time series.

*IMSL STAT/LIBRARY Subprogram Library*

**SSWD** Estimate the nonnormalized spectral density of a stationary time series using a spectral window given the time series data.

**SSWP** Estimate the nonnormalized spectral density of a stationary time series using a spectral window given the periodogram.

*NAG Subprogram Library*

**G13CBF** Calculates the smoothed sample spectrum of a univariate time series using spectral smoothing by the trapezium frequency (Daniell) window.

**L10a3a4 : Spectrum estimation using the Fourier transform of the autocorrelation function**

*DATAPLOT Interactive System*

**SPECTRUM** Plots the spectrum (smoothed Fourier transform of the autocorrelation function) for univariate time series data which are equi-spaced (in time, for example), or, for bivariate time series, plots the cross-, co-, quadrature, coherency, phase, or gain spectrum plots.

*NAG Subprogram Library*

**G13CAF** Calculates the smoothed sample spectrum of a univariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window.

*STARPAC Subprogram Library*

• **UFS** Compute and print a univariate Fourier spectrum analysis of a series.

**UFSF** Compute and print a univariate Fourier spectrum analysis of a series; use FFT for computations.

**UFSFS** Compute and optionally print a univariate Fourier spectrum analysis of a series using user-supplied control values; use FFT for computations; return Fourier spectrum and corresponding frequencies.

**UFSM** Compute and print a univariate Fourier spectrum analysis of a series with missing observations.

**UFSMS** Compute and optionally print a univariate Fourier spectrum analysis of a series with missing observations using user-supplied control values; return Fourier spectrum and corresponding frequencies.

**UFSMV** Compute and print a univariate Fourier spectrum analysis of a series with missing observations; input covariances rather than original series.

**UFSMVS** Compute and optionally print a univariate Fourier spectrum analysis of a series with missing observations using user-supplied control values; input covariances rather than original series; return Fourier spectrum and corresponding frequencies.

**UFSS** Compute and optionally print a univariate Fourier spectrum analysis of a series using user-supplied control values; return Fourier spectrum and corresponding frequencies.

**UFSV** Compute and print a univariate Fourier spectrum analysis of a series; input covariances rather than original series.

**UFSVS** Compute and optionally print a univariate Fourier spectrum analysis of a series using user-supplied control values; input covariances rather than original series; return Fourier spectrum and corresponding frequencies.

**L10a3a5 : Spectrum estimation using autoregressive models**

*CMLIB Library (MXENTRP Sublibrary)*

**BURG** Computes the coefficients of a finite length causal forward or backward prediction filter and uses both the forward and backward predictions in a symmetric manner to generate the maximum entropy spectrum by means of a Toeplitz recursion.

• **easy-to-use version**

*STARPAC Subprogram Library*

- • **UAS** Compute and print a univariate autoregressive spectrum analysis of a series.
- **UASF** Compute and print a univariate autoregressive spectrum analysis of a series; use FFT for computations.
- **UASFS** Compute and optionally print a univariate autoregressive spectrum analysis of a series using user-supplied control values; use FFT for computations; return autoregressive and Fourier spectrum and corresponding frequencies.
- **UASS** Compute and optionally print a univariate autoregressive spectrum analysis of a series using user-supplied control values; return autoregressive and Fourier spectrum and corresponding frequencies.
- **UASV** Compute and print a univariate autoregressive spectrum analysis of a series; input covariances rather than original series.
- **UASVS** Compute and optionally print a univariate autoregressive spectrum analysis of a series using user-supplied control values; input covariances rather than original series; return autoregressive and Fourier spectrum and corresponding frequencies.

## L10a3a6 : Spectral windows

*IMSL STAT/LIBRARY Subprogram Library*

- **DIRIC** Compute the Dirichlet kernel.
- **FEJER** Compute the Fejer kernel.

## L10a3b : Complex demodulation

*DATAPAC Subprogram Library*

- **DEMOD** Performs a complex demodulation on the data in the input vector X at the input demodulation frequency F.

*DATAPLOT Interactive System*

- **COMPLEX DEMODULATION PLOT** Performs complex demodulation and plots either an amplitude or a phase plot for equi-spaced data with a user-specified demodulation frequency. Option: recover estimate of demodulation frequency.

*STARPAC Subprogram Library*

- • **DEMOD** Compute and plot the results of a complex demodulation of the input series.
- **DEMODS** Compute and optionally plot the results of a complex demodulation of the input series; return amplitude and phase functions of demodulated series.

## L10b2 : Time domain analysis of two time series

*BMDP Program Library*

- **P2T** Interactive or batch Box-Jenkins time series analysis for univariate time domain models (including ARIMA, regression, intervention, and transfer function models) – model identification, parameter estimation, testing, forecasting. Options: print, plot, differencing and filtering, save results.

## L10b2a : Summary statistics for two time series

*DATAPLOT Interactive System*

**CORRELATION PLOT** Plots autocorrelations vs. lag for equi-spaced univariate time series data or plots the cross-correlation coefficient vs. lag for equi-spaced bivariate time series data.

*IMSL Subprogram Library*

**FTCRXY** Cross-covariance of two mutually stationary time series.

*IMSL STAT/LIBRARY Subprogram Library*

**CCF** Compute the sample cross-correlation function of two stationary time series.

*MINITAB Interactive System*

**CCF** Computes and graphs cross-correlations between two time series.

*NAG Subprogram Library*

**G13BCF** Calculates cross correlations between two time series.

*STARPAC Subprogram Library*

• **CCF** Compute and print a two-part cross correlation analysis of a pair of series.

**CCFF** Compute and print a two-part cross correlation analysis of a pair of series; use FFT for computations.

**CCFFS** Compute and optionally print a two-part cross correlation analysis of a multivariate series using user-supplied control values; use FFT for computations; return cross covariance function.

**CCFM** Compute and print a two-part cross correlation analysis of a pair of series with missing observations.

**CCFMS** Compute and optionally print a two-part cross correlation analysis of a multivariate series with missing observations using user- supplied control values; return cross covariance function.

**CCFS** Compute and optionally print a two-part cross correlation analysis of a multivariate series using user-supplied control values; return cross covariance function.

## L10b2b : Transfer function models

*IMSL Subprogram Library*

**FTTR** Parameter estimates for a univariate transfer function model.

*IMSL STAT/LIBRARY Subprogram Library*

**IRNSE** Compute estimates of the impulse response weights and noise series of a univariate transfer function model.

**TFPE** Compute preliminary estimates of parameters for a univariate transfer function model.

*NAG Subprogram Library*

**G13BDF** Calculates preliminary estimates of the parameters of a transfer function model.

• **easy-to-use version**

## L10b3 : Frequency domain analysis of two time series *(search also class J1)*

*BMDP Program Library*

**P1T** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), missing values, remove seasonal means and linear trend, filtering, save results.

## L10b3a2 : Cross-periodogram analysis

*IMSL STAT/LIBRARY Subprogram Library*

**CPFFT** Compute the cross periodogram of two stationary time series using a fast Fourier transform.

## L10b3a3 : Cross-spectrum estimation using the cross-periodogram

*IMSL Subprogram Library*

**FTFPS** Fast Fourier transform estimates of power spectra and cross spectra of time series.

*IMSL STAT/LIBRARY Subprogram Library*

**CSSWD** Estimate the nonnormalized cross-spectral density of two stationary time series using a spectral window given the time series data.

**CSSWP** Estimate the nonnormalized cross-spectral density of two stationary time series using a spectral window given the spectral densities and cross periodogram.

*NAG Subprogram Library*

**G13CDF** Calculates the smoothed sample cross spectrum of a bivariate time series using spectral smoothing by the trapezium frequency (Daniell) window.

## L10b3a4 : Cross-spectrum estimation using the Fourier transform of the cross-correlation or cross-covariance function

*DATAPLOT Interactive System*

**SPECTRUM** Plots the spectrum (smoothed Fourier transform of the autocorrelation function) for univariate time series data which are equi-spaced (in time, for example), or, for bivariate time series, plots the cross-, co-, quadrature, coherency, phase, or gain spectrum plots.

*NAG Subprogram Library*

**G13CCF** Calculates the smoothed sample cross spectrum of a bivariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window.

*STARPAC Subprogram Library*

• **BFS** Compute and print a bivariate Fourier spectrum analysis of a pair of series.

**BFSF** Compute and print a bivariate Fourier spectrum analysis of a pair of series; use FFT for computations.

**BFSFS** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series using user-supplied control values; use FFT for computations; return squared coherency and phase components of the cross spectrum and the corresponding frequencies.

**BFSM** Compute and print a bivariate Fourier spectrum analysis of a pair of series with missing observations.

**BFSMS** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series with missing observations using user-supplied control values; return squared coherency and phase components of the cross spectrum and the corresponding frequencies.

**BFSMV** Compute and print a bivariate Fourier spectrum analysis of a pair of series with missing observations; input covariances rather than original series.

**BFSMVS** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series with missing observations using user-supplied control values; input covariances rather than original series; return squared coherency and phase components of the cross spectrum and the corresponding frequencies.

**BFSS** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series using user-supplied control values; return squared coherency and phase components of the cross spectrum and the corresponding frequencies.

**BFSV** Compute and print a bivariate Fourier spectrum analysis of a pair of series; input covariances rather than original series.

**BFSVS** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series using user-supplied control values; input covariances rather than original series; return squared coherency and phase components of the cross spectrum and the corresponding frequencies.

## L10b3a6 : Spectral functions

*DATAPLOT Interactive System*

**SPECTRUM** Plots the spectrum (smoothed Fourier transform of the autocorrelation function) for univariate time series data which are equi-spaced (in time, for example), or, for bivariate time series, plots the cross-, co-, quadrature, coherency, phase, or gain spectrum plots.

*NAG Subprogram Library*

**G13CEF** Calculates the cross amplitude spectrum and squared coherency, together with lower and upper bounds from the univariate and bivariate (cross) spectra.

**G13CFF** Calculates the gain and phase together with lower and upper bounds from the univariate and bivariate spectra.

**G13CGF** Calculates the noise spectrum together with multiplying factors for the bounds and the impulse response function and its standard error, from the univariate and bivariate spectra.

## L10c : Analysis of a multivariate time series *(search also classes J1, L3e3, and L10b)*

*Collected Algorithms of the ACM*

**A640** SFRMG: A Fortran subprogram which takes real matrices A (n-by-n), B (n-by-m), and C (l-by-n) and forms the complex frequency response matrix CEB, where E is the inverse of (FREQ×I – A), I is the n-by-n identity matrix and FREQ is a complex scalar parameter taking values along the imaginary axis for continuous-time systems and on the unit circle for discrete-time systems. (See A.J. Laub, ACM TOMS 12 (1986) pp. 26-33.)

*IMSL Subprogram Library*

**FTFREQ** Single or multichannel time series analysis in the time and frequency domains.

**FTKALM** Kalman filtering.

*IMSL STAT/LIBRARY Subprogram Library*

**KALMN** Perform Kalman filtering and evaluate the likelihood function for the state-space model.

*NAG Subprogram Library*

**G13DAF** Calculates the cross covariance or cross correlation function of a multivariate time series.

**G13DBF** Calculates the multivariate partial autocorrelation function of a multivariate time series.

**G13DCF** Fits a vector autoregressive moving average (VARMA) model to an observed vector of time series using the method of maximum likelihood. Standard errors of parameter estimates are computed along with their appropriate correlation matrix. The routine also calculates estimates of the residual series.

*TIMSAC Program Library*

**BLOMAR** Locally fits multivariate autoregressive models to non-stationary time series by a Bayesian procedure. (This program is tentative and further testing and improvement through practical applications will be necessary.)

**MLOMAR** Locally fits multivariate autoregressive models to non-stationary time series by the minimum AIC procedure using the Householder transformation.

**MULBAR** Determines multivariate autoregressive models by a Bayesian procedure. The basic least squares estimates of the parameters are obtained by the Householder transformation.

**MULMAR** Fits a multivariate autoregressive model by the minimum AIC procedure. Only the possibilities of zero coefficients at the beginning and end of the model are considered. The least squares estimate of the parameters are obtained by the Householder transformation.

## L10d : Analysis of two multi-channel time series

*IMSL Subprogram Library*

**FTCROS** Means, variances, cross-covariances, and cross-correlations for two mutually stationary n channel time series.

**FTWENM** Multichannel Wiener forecast.

**FTWENX** Maximum likelihood parameter estimates for a multichannel, single output time series model.

*IMSL STAT/LIBRARY Subprogram Library*

**MCCF** Compute the multichannel cross-correlation function of two mutually stationary multichannel time series.

**MLSE** Compute least squares estimates of a linear regression model for a multichannel time series with a specified base channel.

**MWFE** Compute least squares estimates of the multichannel Wiener filter coefficients for two mutually stationary multichannel time series.

*NAG Subprogram Library*

**G13BEF** Estimates the parameters in a multi-input model relating one output series to one or more input series, using a choice of non-linear least squares, exact likelihood, or marginal likelihood for estimation criteria. If no input series are present, estimates a univariate ARIMA model.

**G13BGF** Accepts a series of new observations of an output time series and any associated input time series, for which a multi-input model is already fully specified, and updates the state set information for use in constructing further forecasts.

**G13BHF** Produces forecasts of a time series (the output series) which depends on one or more other (input) series via a multi-input model which will usually have been fitted using G13BEF.

**G13BJF** Produces forecasts of a time series (the output series) which depends on one or more other (input) series via a previously estimated multi-input model for which the state set information is not available.

## L11: Correlation Analysis

Software in this class provides more sophisticated correlation analyses than that provided in class L4.

### References

[Tho78] R. M. Thorndike. *Correlation Procedures for Research.* Gardner Press, Inc., New York, 1978.

[Whe84] R. J. Wherry. *Contributions to Correlation Analysis.* Academic Press, Orlando, FL, 1984.

| **L11 :** | **Correlation analysis** *(search also classes L4 and L13c)* |
|---|---|

*BMDP Program Library*

**P6R** Computes the partial correlations of a set of variables after removing the linear effects of a second set of variables. Can be used for regression, especially if multiple dependent variables are present. Prints standard results. Options: weights, form of input, additional printing and plots.

*SPSS Program Library*

**PARTIAL CORR** Produces a matrix of partial correlation coefficients describing the relationship between two variables, adjusted for effects of additional variables. Options: one- or two-tailed significance levels, univariate statistics, missing values.

## L12: Discriminant Analysis

Discriminant analysis is concerned with distinguishing between two or more populations by developing a discriminant function in which the dependent variable, an indicator variable identifying the population, is a function of one or more independent random variables. A "training" data set of observations of the independent variables and known population memberships may be used to develop the discriminant function, often by stepwise techniques. Classical discriminant analysis assumes that each population is multivariate normal with known variance-covariance structure, and substantial statistical inference is possible in this case. The quality of the fitted function and the appropriateness of statistical assumptions can be evaluated through statistics including "percent correctly classified," residual analysis, and graphical techniques.

### References

[And58] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis.* John Wiley & Sons, New York, 1958.

[BFO84] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth International Group, Belmont, CA, 1984.

[Cac73] T. Cacoullos, editor. *Discriminant Analysis and Applications.* Academic Press, New York, 1973.

[DH73] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley-Interscience, New York, 1973.

[Gol78] M. Goldstein. *Discrete Discriminant Analysis.* John Wiley & Sons, New York, 1978.

[Jam85] M. James. *Classification Algorithms.* John Wiley & Sons, New York, 1985.

[Kle80] W. R. Klecka. *Discriminant Analysis.* Sage Publications, Beverly Hills, CA, 1980.

**L12 : Discriminant analysis**

*BMDP Program Library*

**P7M** Stepwise forward or backward discriminant analysis (including jackknifed classifications, percent correctly classified, Mahalanobis' distances, canonical variable coefficients, eigenvalues, scores, and plot of first two canonical variables). Options: interactive stepping, save results.

*IMSL Subprogram Library*

**ODFISH** Linear discriminant analysis method of Fisher for reducing the number of variables.

**ODNORM** Multivariate normal linear discriminant analysis among several known groups.

*IMSL STAT/LIBRARY Subprogram Library*

**DMSCR** Use Fisher's linear discriminant analysis method to reduce the number of variables.

**DSCRM** Perform a linear or a quadratic discriminant function analysis among several known groups.

**NNBRD** K nearest neighbor discrimination.

*SAS Program Library*

**CANDISC** Performs a canonical discriminant analysis, computes Mahalanobis distances, and does both univariate and multivariate one-way analyses of variance. Tests zero correlations using an F approximation. Options: weights, missing values.

**DISCRIM** Computes linear or quadratic discriminant functions for classifying observations into two or more groups. The distribution within each group should be approximately multivariate normal. The classification criterion can be based on either the individual within-group covariance matrices or the pooled covariance matrix. Options: homogeneity of the within-group covariance test, missing values.

**NEIGHBOR** Performs a nearest neighbor discriminant analysis, classifying observations into groups according to either the nearest neighbor rule or the k-nearest-neighbor when the classes do not have multivariate normal distributions. Proximity is determined by either Mahalanobis or Euclidean distances. Options: use of prior probabilities in the classification, missing values.

**STEPDISC** Performs a stepwise discriminant analysis by forward selection, backward elimination, or stepwise selection of variables. The classes are assumed to be multivariate normal with a common covariance matrix. Options: weights, missing values.

*SPSS Program Library*

**DISCRIMINANT** Performs multiple discriminant analysis. Options: statistical analyses, classifications, five stepwise selection methods (including minimizing Wilks' lambda, maximizing Mahalanobis' distance, maximizing Rao's V), varimax rotation, missing values.

## L13: Covariance Structure Models

A major application of two covariance structure model methods, principal components analysis and factor analysis, is the reduction of the dimensionality of a data set through the identification of a small collection of new independent variables, each a linear combination of the independent variables in the original data set, which contain the bulk of the variation in the original data. These new variables often provide better understanding of the data and, because they are fewer, can often be more effectively used in subsequent analyses, e.g., regression.

## References

[Bar87] D. J. Bartholomew. *Latent Variable Models and Factor Analysis.* Oxford University Press, London, 1987.

[Har76] H. H. Harman. *Modern Factor Analysis.* The University of Chicago Press, Chicago, 1976.

[Jol86] I. T. Jolliff. *Principal Component Analysis.* Springer-Verlag, New York, 1986.

[Mal80] E. Malinowski. *Factor Analysis in Chemistry.* John Wiley & Sons, New York, 1980.

[Mor76] D. F. Morrison. *Multivariate Statistical Methods.* McGraw-Hill, New York, 1976.

[Tim75] N. H. Timm. *Multivariate Analysis with Applications in Education and Psychology.* BrooksCole, Monterey, California, 1975.

**L13a : Factor analysis**

*BMDP Program Library*

**P4M** Provides four methods of initial factor extraction from a correlation or covariance matrix, and several methods of rotation, prints shaded correlations, factor loadings, factor score coefficients, factor scores, Mahalanobis distances, and plots. Options: weights, form of input, save results.

**P8M** Boolean factor analysis of binary (dichotomous) data. Options: initial estimates of the loading matrix, printing, save results.

*CMLIB Library (CLUSTER Sublibrary)*

**SPARSE** Approximates the covariance matrix by BB(trans) where B contains many zeros and corresponds to the loading matrix for factor analysis where clusters of variables are the factors.

*IMSL Subprogram Library*

**OFCOEF** Compute a matrix of factor score coefficients for input to IMSL routine OFSCOR.

**OFCOMM** Compute an unrotated factor loading matrix according to a common factor model by unweighted or generalized least squares, or by maximum likelihood procedures.

**OFHARR** Transformation of unrotated factor loading matrix to oblique axes by Harris-Kaiser method.

**OFIMAG** Compute an unrotated factor loading matrix according to an image model.

**OFPROT** Oblique transformation of the factor loading matrix using a target matrix, including pivot and power vector options.

**OFRESI** Communalities and normalized factor residual correlation matrix calculation.

**OFROTA** Orthogonal rotation of a factor loading matrix using a generalized orthomax criterion, including quartimax, varimax, and equamax.

**OFSCHN** Orthogonal transformation of the factor loading matrix using a target matrix.

**OFSCOR** Compute a set of factor scores given the factor score coefficient matrix.

*IMSL STAT/LIBRARY Subprogram Library*

**FACTR** Extract initial factor loading estimates in factor analysis.

**FCOEF** Compute a matrix of factor score coefficients for input to IMSL routine FSCOR.

**FDOBL** Compute a direct oblimin rotation of a factor loading matrix.

**FHARR** Compute an oblique rotation of an unrotated factor loading matrix using the Harris-Kaiser method.

**FIMAG** Compute the image transformation matrix.

**FOPCS** Compute an orthogonal Procrustes rotation of a factor loading matrix using a target matrix.

**FPRMX** Compute an oblique Promax or Procrustes rotation of a factor loading matrix using a target matrix, including pivot and power vector options.

**FRESI** Compute communalities and the standardized factor residual correlation matrix.

**FROTA** Compute an orthogonal rotation of a factor loading matrix using a generalized orthomax criterion, including quartimax, varimax, and equamax rotations.

**FRVAR** Compute the factor structures and the variance explained by each factor.

**FSCOR** Compute a set of factor scores given the factor score coefficient matrix.

*NAG Subprogram Library*

**G11SAF** Fits a latent variable model (with a single factor) to data consisting of a set of measurements on individuals in the form of binary-valued sequences (generally referred to as score patterns). Various measures of goodness of fit are calculated along with the factor (theta) scores.

*SAS Program Library*

**FACTOR** Performs several types of common factor and component analysis for multivariate data, a correlation matrix, a covariance matrix, a factor pattern, or a matrix of scoring coefficients. A variety of methods are available for extracting factors, for prior communality estimation, and for rotation. Options: weights, factor scores.

*SPSS Program Library*

**FACTOR** Performs principal components analysis and factor analysis. Options: seven factor extraction techniques (principal axis factoring, least squares, maximum likelihood, etc.), rotation (varimax, equamax, quartimax, etc.), factor scores (regression, Bartlett, Anderson-Rubin), plot, missing values.

## L13b : Principal components analysis

*IMSL Subprogram Library*

**OFPRI** Compute an unrotated factor loading matrix according to a principal component model.

**OPRINC** Principal components of a multivariate sample of observations.

*IMSL STAT/LIBRARY Subprogram Library*

**PRINC** Compute principal components from a variance-covariance matrix or a correlation matrix.

*SAS Program Library*

**PRINCOMP** Performs principle component analysis on raw data, a correlation matrix, or a covariance matrix. Options: weights, missing values.

*SPSS Program Library*

**FACTOR** Performs principal components analysis and factor analysis. Options: seven factor extraction techniques (principal axis factoring, least squares, maximum likelihood, etc.), rotation (varimax, equamax, quartimax, etc.), factor scores (regression, Bartlett, Anderson-Rubin), plot, missing values.

| **L13c :** | **Canonical correlation analysis** |
|---|---|

*BMDP Program Library*

**P6M** Computes canonical correlation analysis for two sets of variables and Bartlett's test for the significance of the remaining eigenvalues, with printing. Options: weights, form of input, additional printing and plotting, save results.

*SAS Program Library*

**CANCORR** Performs canonical correlation and tests correlation hypotheses using an F approximation. Both standardized and unstandardized canonical coefficients and correlations between canonical variable and the original variables are produced. Options: canonical redundancy analysis, partial canonical correlation, weights, output data sets of scores on each canonical variable and canonical coefficients.

## L14: Cluster Analysis

Cluster analysis is a tool for discovering structure in large multi-dimensional data sets. The first step in most cluster analysis is constructing a matrix whose $(i,j)$ entry is the similarity (or dissimilarity, e.g., Euclidean distance) between observations $i$ and $j$ (see class L14d). Single link is one of a large collection of joining methods. In it, each observation initially is a "cluster" and in the iterative step those two clusters are merged which contain the two observations which are the most similar of all pairs of observations in different clusters. Alternatives such as complete link produce tighter clusters than the long narrows ones often produced by single link.

Single link is an unconstrained nested method. It is nested because once a cluster is formed, it remains intact through subsequent iterative steps. An example of constrained one-way clustering is Fisher's algorithm, in which observations also have a time component, and in the iterative step only those clusters which are consecutive in time are candidates for merging. K-means clustering is a widely used non-nested method. The number of clusters, $k$, is specified, and in the iterative step observations can be moved from one cluster to another to minimize distance to cluster means.

## References

[BFO84] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth International Group, Belmont, CA, 1984.

[Har75] J. A. Hartigan. *Clustering Algorithms.* John Wiley & Sons, New York, 1975.

[JS71] N. Jardine and R. Sibson. *Mathematical Taxonomy.* John Wiley & Sons, New York, 1971.

[Spa80] H. Spath. *Cluster Analysis Algorithms.* Ellis Horwood Limited, Chichester, England, 1980.

| **L14a1a1 :** | **Joining clustering algorithms (e.g., single link)** |
|---|---|

*BMDP Program Library*

**P1M** Stepwise cluster analysis of variables using one of four measures of similarity, three criteria for combining clusters, with printing of a summary table of clusters, shaded distance measure display, and a tree showing cluster formation. Options: form of input, additional printing and display.

**P2M** Stepwise cluster analysis of cases (observations) using one of four distance measures (including Euclidean and one for data that are frequency counts) and three linkage algorithms (single, centroid, k nearest neighbors), with a summary table of clusters and a cluster tree. Options: weights, standardized data.

*CMLIB Library (CLUSTER Sublibrary)*

**JOIN** Uses a general joining algorithm to form and output a tree of clusters of cases.

**LETREE** Uses a leader clustering algorithm to construct a tree whose levels are determined by user-defined thresholds.

**LINK** Constructs and prints tree of clusters obtained by adding cases in succession to minimize the sum of the linking distances.

**SLINK** Utilizes the single-linkage clustering algorithm to construct a tree from a user-specified distance matrix.

*IMSL Subprogram Library*

**OCLINK** Perform a single-linkage or complete-linkage hierarchical cluster analysis given a similarity matrix.

*IMSL STAT/LIBRARY Subprogram Library*

**CLINK** Perform a hierarchical cluster analysis given a distance matrix.

*SAS Program Library*

**CLUSTER** Hierarchically clusters observations by one of eleven procedures (standard linkage methods, density linkage (including kth-nearest-neighbor and two-stage), and maximum-likelihood for mixtures of spherical multivariate normal distributions). Input data can be either coordinates or distances. Options: trimming input data, missing values.

*SPSS Program Library*

**CLUSTER** Performs agglomerative hierarchical clustering. Options: clustering methods (single link, complete link, between- and within-groups average link, median, centroid, and Ward's methods), six proximity measures, missing values.

## L14a1a2 : Divisive clustering algorithms

*CMLIB Library (CLUSTER Sublibrary)*

**SPARSE** Approximates the covariance matrix by BB(trans) where B contains many zeros and corresponds to the loading matrix for factor analysis where clusters of variables are the factors.

**SPLIT1** Splits cases in each variable until all within-cluster variances are smaller than a user-specified threshold.

*SAS Program Library*

**VARCLUS** Performs either disjoint or hierarchical clustering of variables by maximizing the variation accounted for by either the first principal component or the centroid component of each cluster. Options: weights, missing values.

## L14a1a3 : Switching clustering algorithms

*CMLIB Library (CLUSTER Sublibrary)*

**DOT** Creates a tree of clusters of cases for categorical data by minimum-mutation fits (changes of variable values between a cluster and its direct ancestor in the tree are minimized).

## L14a1a4 : Predict missing values using cluster analysis

*CMLIB Library (CLUSTER Sublibrary)*

**AID** Forms a tree of clusters by splitting cases on values of individual variables to minimize the sum of the squared deviations from the cluster means. The tree can be used to predict the value of a different variable for a new case.

**VARCO** Sets up a tree of clusters of cases and uses the variance components algorithm to predict the value of the mean and variance for a predictor variable for each node in the tree, which can be used as approximations for a new case.

## L14a1b : Non-nested clustering algorithms

*BMDP Program Library*

**PKM** By k-means procedure, partitions a set of cases (observations) into k clusters – beginning with user-specified initial clusters or one cluster, proceeding in divisive stepwise manner, then doing iterative reallocation – prints cluster profile and plot. Options: weights, standardize data (four ways), save results.

*CMLIB Library (CLUSTER Sublibrary)*

**ASSIGN** Assigns each case to the cluster whose center is a minimum Euclidean distance from the case. Can be used with CLUSTER subroutine RELOC to form a user-defined K-means package.

**BUILD** Builds clusters by the K-means algorithm, printing the results for all intermediate iterations.

**DITTO** Partitions categorical data into clusters by maximizing the matches between cases in a cluster and the cluster mode.

**MIX** Fits the mixture model by a maximum log-likelihood criterion.

**MIXIND** Fits the mixture model from k multivariate normals where k is the desired number of clusters. The variables are assumed to have variance constant over different clusters.

**QUICK** Finds a quick partition of the cases by comparing, to a user-defined threshold, the Euclidean distances to the existing cluster leaders.

**RELOC** Sets each cluster center equal to the cluster mean. Can be used with CLUSTER subroutine ASSIGN to form a user-defined K-means package.

**SEARCH** Produces and outputs tree which maximizes $\alpha \times$TRUE - $(1.0 - \alpha)\times$FALSE where TRUE is the number of triads correctly predicted by the tree and FALSE is the number of triads incorrectly predicted by the tree and $\alpha$ is a user-defined probability.

*IMSL STAT/LIBRARY Subprogram Library*

**KMEAN** Perform a K-means (centroid) cluster analysis.

*SAS Program Library*

**FASTCLUS** Performs a disjoint cluster analysis by minimizing the sum of squared distances from the cluster means. User specifies the maximum number of clusters and optionally, the minimum radius of the clusters. Designed for use with large data sets. Options: weights, missing values.

**NEIGHBOR** Performs a nearest neighbor discriminant analysis, classifying observations into groups according to either the nearest neighbor rule or the k-nearest-neighbor when the classes do not have multivariate normal distributions. Proximity is determined by either Mahalanobis or Euclidean distances. Options: use of prior probabilities in the classification, missing values.

**VARCLUS** Performs either disjoint or hierarchical clustering of variables by maximizing the variation accounted for by either the first principal component or the centroid component of each cluster. Options: weights, missing values.

*SPSS Program Library*

**QUICK CLUSTER** Produces clusters by finding cluster centers based on values of cluster variables and by assigning cases to centers that are nearest. Default is McQueen's k-means clustering method. Options: specify the number of clusters, select initial cluster centers, update centers, missing values.

## L14a2 : Constrained clustering algorithms

*CMLIB Library (CLUSTER Sublibrary)*

**FISH** Clusters a sequence of cases into subsequences by Fisher's method of exact optimization.

## L14b : Two-way cluster analysis

*BMDP Program Library*

**P3M** Forms blocks (submatrices of the data matrix) where a subset of the cases (for a subset of the variables) cluster together, with printing of the blocks and tree diagrams for cases and for variables – appropriate for categorical data with few levels.

*CMLIB Library (CLUSTER Sublibrary)*

**JOIN2** Joins data values in a case-by-variable matrix into blocks until all within-block variances are greater than a user-specified threshold.

**SPLIT2** Splits matrix of case-by-variable data values into blocks until all within-block variances are less than a given threshold. Includes user-controlled constraints.

## L14c : Display cluster analysis output

*IMSL Subprogram Library*

**USTREE** Print a binary tree.

*IMSL STAT/LIBRARY Subprogram Library*

**TREEP** Print a binary tree.

*SAS Program Library*

**TREE** Prints a tree diagram from the output generated by SAS procedures CLUSTER or VARCLUS. Can also create an output data set identifying disjoint clusters at a specified level in the tree. Optional user control of plot features.

## L14d : Service routines for cluster analysis (e.g., compute distance matrix)

*CMLIB Library (CLUSTER Sublibrary)*

**DIST** Computes the Euclidean distance between two cases or two variables.

**IN** Reads in raw data matrix and labels for the cases and variables and sets up the data structures necessary for most of the subroutines in the CLUSTER sublibrary.

**INVERT** Computes the inverse and determinant of a symmetric matrix (e.g. a covariance matrix).

**MISS** Replaces a missing data value by the cluster mean.

**OUT** Prints a data matrix with the appropriate labels for the cases and variables which is output from many of the subroutines from the CLUSTER sublibrary.

**SCALE** Discretizes the data into classes.

**STAND** Standardizes a data matrix such that each variable has mean zero and unit variance. Missing values are ignored.

**TWO** Computes overall mean and covariance matrices of a data set with appropriate labels for the cases and variables.

**WCOV** Computes the within-cluster covariance matrix.

**WDIST** Computes a weighted Euclidean distance between two cases.

*IMSL Subprogram Library*

**OCDIS** Pairwise Euclidean distances between the columns of a matrix.

*IMSL STAT/LIBRARY Subprogram Library*

**CDIST** Compute a matrix of dissimilarities (or similarities) between the columns (or rows) of a matrix.

**CNUMB** Compute cluster membership for a hierarchical cluster tree.

*SAS Program Library*

**ACECLUS** Obtains approximate estimates of the pooled within-cluster covariance matrix when the clusters can be assumed multivariate normal with equal covariance matrices. Neither cluster membership nor the number of clusters need to be known. Options: weights, missing values.

## L15 : Life testing and survival analysis

*BMDP Program Library*

**P1L** Estimates survival (time-to-response) distribution of patients who have been observed over varying periods of time by product-limit (Kaplan-Meier) or actuarial life table (Cutler-Ederer) method. Options: three forms of input, Mantel-Cox and Breslow test of equality of survival curves, five plots.

**P2L** Analyzes survival data with covariates using Cox proportional hazard regression model. Options: two forms of input, stepwise selection of covariates, time-dependent covariates, stratification, significance tests, three plots, print survival functions and residuals.

*IMSL Subprogram Library*

**CLIFE** Life table analysis.

*IMSL STAT/LIBRARY Subprogram Library*

**ACTBL** Produce population and cohort life tables.

• **HAZEZ** Perform nonparametric hazard rate estimation using kernel functions. Easy-to-use version of HAZRD.

**HAZRD** Perform nonparametric hazard rate estimation using kernel functions and quasi-likelihoods.

**HAZST** Perform hazard rate estimation over a grid of points using a kernel function.

**KAPMR** Compute Kaplan-Meier estimates of survival probabilities in stratified samples.

• **easy-to-use version**

*SAS Program Library*

LIFEREG Fits parametric models to failure-time data that may be right censored. Models include exponential, Weibull, log normal, and log logistic. Parameters are estimated by maximum likelihood using a Newton-Raphson algorithm. Independent variable may be continuous or discrete.

LIFETEST Computes nonparametric estimates of the survival distribution (by the product limit method or the life table method) and computes rank tests for association of the response variable with covariates for stratified data that may be right censored. Options: tests homogeneity between strata, missing values, printer plots.

*SPSS Program Library*

SURVIVAL Performs survival analysis. Options: subgroup comparison (using Lee and Desu algorithm calculated from statistic D), plots of the survival functions (cumulative distribution, probability density, hazard rate), missing values.

## L16 : Multidimensional scaling

*IMSL STAT/LIBRARY Subprogram Library*

MSDST Compute distances in a multidimensional scaling model.

MSIDV Perform individual-differences multidimensional scaling for metric data using alternating least squares.

## L17 : Statistical data sets

*IMSL STAT/LIBRARY Subprogram Library*

GDATA Retrieves one of several data sets frequently cited in the statistical literature, including the Longley data, Fisher's iris data, and data from books by Anderson, Draper and Smith, Box and Jenkins, and Robinson.

# N: Data Handling

Software for data handling provides the basic tools needed to manipulate data structures. A data structure is a method for organizing data, together with a set of possible operations on the data. While the most familiar data structures include real numbers, character strings, arrays, and sequential files, many more exotic ones have proven useful. Examples of these are lists, stacks, queues, heaps, trees, hash tables, and special file types. Lists are ordered sequences of items. Stacks and queues are special types of lists. In a stack items are retrieved on a last-in first-out basis, while in a queue, first-in first-out is the rule. In a priority queue each item is given a priority and the first-in item of the highest priority is retrieved first. Heaps are priority queues which are implemented in a certain very efficient manner. Trees are structures which represent hierarchies. Software for manipulating each of these can be found in this chapter.

Software for input and output of data is catalogued in class N1. This software provides capabilities for reading and writing not usually available as standard options in programming languages, such as free-format input and easy-to-read output of vectors and matrices.

Operations on two specialized data types, bit strings and character strings, can be found in classes N2 and N3. Typical operations on character strings include conversion between upper and lower case, conversion between characters and their integer ASCII values, and locating first and last non-blank characters. There is little software currently available in GAMS for performing bit manipulation.

Class N4 contains software for the management of more complex data types. The ACM Collected Algorithms contains software for managing stacks, heaps, decision trees, and text files. The IMSL and PORT library routines cataloged here manage the internal working storage stack utilized by other subprograms in these libraries; these routines are not usually called directly by library users.

Software designed to find the position of a particular piece of data within a data structure, usually some form of list, is catalogued in class N5. Software for rearranging lists into sorted order is found in N6. The process of combining two sorted lists into a third list containing the data of both is called merging; software for this operation is cataloged in class N7. Software for permuting lists, i.e. rearranging them into the order specified by a given rank vector, is found in class N8.

## References

[HS82] E. Horowitz and S. Sahni. *Fundamentals of Data Structures.* Computer Sciences Press, Rockville, Maryland, 1982.

[K173] D. E. Knuth. *The Art of Computer Programming Volume 1: Fundamental Algorithms.* Addison-Wesley, Reading, Massachusetts, 1973.

## N1 : Input, output of data

*DATAPAC Subprogram Library*

**READ** Performs a format-free read.

**READG** Performs a format-free read of data from input unit IRD.

**SKIPR** Reads through (skips over) NLHEAD lines from input unit 5.

**WRITE** Writes out the contents of the vector X.

*IMSL Subprogram Library*

**USCWV** Print a complex vector.

**USTREE** Print a binary tree.

**USWBM** Print a matrix stored in band storage mode.

**USWBS** Print a matrix stored in band symmetric storage mode.

**USWCH** Print a complex matrix stored in Hermitian storage mode.

**USWCM** Print a complex matrix stored in full storage mode.

**USWFM** Print a matrix stored in full storage mode.

**USWFV** Print a vector.

**USWSM** Print a matrix stored in symmetric storage mode.

*IMSL MATH/LIBRARY Subprogram Library*

**WRCRL** Print a complex rectangular matrix with a given format and labels.

• **WRCRN** Print a complex rectangular matrix with integer row and column labels.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**PAGE** Set or retrieve page width and length for printing.

**WRIRL** Print an integer rectangular matrix with a given format and labels.

• **WRIRN** Print an integer rectangular matrix with integer row and column labels.

**WROPT** Set or retrieve an option for printing a matrix.

**WRRRL** Print a real rectangular matrix with a given format and labels.

• **WRRRN** Print a real rectangular matrix with integer row and column labels.

*MAGEV Cyber 205 Subprogram Library*

**SLICE4** To provide the user with high-speed I/O, taking advantage of the efficiency of the system-provided subroutines Q7BUFIN and Q7BUFOUT, and of the concept of parallelism, achievable by distributing the data transfer over 4 different I/O-paths, or channels.

*NAG Subprogram Library*

**X04BAF** Writes a single formatted record to an external file.

**X04BBF** Reads a single formatted record from an external file.

*Scientific Desk PC Subprogram Library*

**N1ECC** Allows entry or correction of a rectangular matrix or vector from the keyboard, with input checking.

**N1EEC** Allows entry or correction of a symmetric matrix stored in packed form from the keyboard, with input checking.

**N1LM** Allows screen display, entry, and correction, of a large vector or matrix. Elements, rows, or columns can be entered, changed or set to zero.

**N1M** Prints a matrix on the video display screen or on a printer.

**N1MSD** Prints a matrix on the video display screen or on a printer, allowing specification of the number of output digits desired. Accepts either single or double precision input array

**N1RG** Writes the Mth row of a matrix, or N elements of a vector, using G15.7 format.

**N1RS** Writes the Mth row of a symmetric matrix stored in packed form, using G15.7 format.

**N1TC** Identifies array output by printing page number, title, and column numbers for an array that is subsequently to be printed.

**N1V** Prints a vector (array) on the video display screen or a line printer.

**N1VSD** Prints a vector (array) on the video display screen or a line printer, allowing specification of the number of output digits desired. Accepts either single or double precision input array.

**N3** Reads a datum as a character string of length 20 from the keyboard and converts it to the desired type if it is possible. Otherwise it returns an error code. Also converts all lower case letters to upper case.

| N2 : | Bit manipulation |
|---|---|

*IMSL Subprogram Library*

**GTPBC** Count of the number of zero bits in a given subset of a real word.

| N3 : | Character manipulation |
|---|---|

*BMDP Program Library*

**P4D** Counts frequency of each number, letter, or symbol in single-column fields (A1 format). Options: input case label variables in A4 format, diagnostic printing useful in preliminary data screening. Specified characters may be replaced by blanks or symbols.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**ACHAR** Return the character whose ASCII value is the input integer argument.

**CVTSI** Convert a character string containing an integer number into the corresponding integer form.

**IACHAR** Return the integer ASCII value of a character argument.

**ICASE** Convert from character to the integer ASCII value without regard to case.

**IICSR** Compare two character strings using the ASCII collating sequence but without regard to case.

**IIDEX** Determine the string position indicating the starting position at which a key character sequence begins without regard to case.

*Scientific Desk PC Subprogram Library*

**N3FNB** Returns the index of the first non-blank character in an input string.

**N3LNB** Returns the index of the last non-blank character in an input string.

**N3LOW** Converts an input string to lower case.

**N3UP** Converts an input string to upper case.

| N4 : | Storage management (e.g., stacks, heaps, trees) |
|---|---|

*Collected Algorithms of the ACM*

**A528** Three program packages which provide a framework for a portable Fortran subroutine library: machine-dependent constants, automatic error handling, and dynamic storage allocation using a stack. Developed for use with the PORT library. (See P.A. Fox, A.D. Hall, and N.L. Schryer, ACM TOMS 4 (1978) pp. 176-188.)

**A561** Fortran subroutines for efficient table maintenance using heaps. (See D.K. Kahaner, ACM TOMS 6 (1980) pp. 444-449.)

**A606** NITPACK: Fortran subprograms for implementing decision trees. NITREE converts a paper tree into a format that may be transmitted to a user at a computer terminal. NIT performs the transmission. (see P.W. Gaffney et al., ACM TOMS 9 (1983) pp. 418-426.)

**A607** TES (Text Exchange System): transportable Fortran programs for management and exchange of programs and other text. TES defines a format for information storage and includes two programs, the first creates, reads, and maintains TES files, and the second permits TES to be distributed on a tape in TES format. (See W.V. Snyder and R.J. Hanson, ACM TOMS 9 (1983) pp. 427-440.)

*IMSL MATH/LIBRARY, STAT/LIBRARY and SFUN/LIBRARY Subprogram Libraries*

**IWKCIN** Initialize bookkeeping locations describing the character workspace stack.
**IWKIN** Initialize bookkeeping locations describing the workspace stack.

*PORT Subprogram Library*

**ISTKGT** Allocates (gets) an array from the storage stack for PORT library programs.
**ISTKIN** Initialize the length of the dynamic storage stack for PORT library programs.
**ISTKMD** Changes size of last stack allocation for PORT library programs.
**ISTKQU** Returns the number of available items that remain in the stack for PORT library programs.
**ISTKRL** Releases the last storage allocations requested for PORT library programs.
**ISTKST** Returns information on the status of the stack for PORT library programs.

## N5a : Search for extreme value in an array

*CMLIB Library (BLAS Sublibrary)*

**ICAMAX** Find smallest index of maximum magnitude component of a complex vector.
**ISAMAX** Find smallest index of maximum magnitude component of a single precision vector.

*CMLIB Library (XBLAS Sublibrary)*

**ISAMIN** Find the smallest index of the minimum magnitude component of a real vector.
**ISMIN** Find the smallest index of the minimum component of a real vector.

*DATAPAC Subprogram Library*

**MAX** Computes the sample maximum of the data in the input vector X.
**MIN** Computes the sample minimum of the data in the input vector X.

*IMSL Subprogram Library*

**USMNMX** Determination of the minimum and maximum values of a vector.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**ICAMAX** Find the smallest index of the component of a complex vector having maximum magnitude.
**ICAMIN** Find the smallest index of the component of a complex vector having minimum magnitude.
**IIMAX** Find the smallest index of the maximum component of a integer vector.
**IIMIN** Find the smallest index of the minimum of an integer vector.
**ISAMAX** Find the smallest index of the component of a single-precision vector having maximum absolute value.
**ISAMIN** Find the smallest index of the component of a single-precision vector having minimum absolute value.
**ISMAX** Find the smallest index of the component of a single-precision vector having maximum value.
**ISMIN** Find the smallest index of the component of a single-precision vector having minimum value.

*NAG Subprogram Library*

**F06FLF** Elements of real vector with largest and smallest absolute value.
**F06JLF** Index, real vector element with largest absolute value.
**F06JMF** Index, complex vector element with largest absolute value.
**F06KLF** Last non-negligible element of real vector.

*PORT Subprogram Library*

**EXTRMI** Finds extremal points of an integer function defined on a mesh.
**EXTRMR** Finds extremal points of a real function defined on a mesh.

| N5b : | Search for insertion position in an array |
|---|---|

*PORT Subprogram Library*

**INTRVI** Finds the interval in an integer array to which an integer element belongs.

**INTRVR** Finds the interval in a real array to which a real element belongs.

| N5c : | Search on a key |
|---|---|

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**IIDEX** Determine the string position indicating the starting position at which a key character sequence begins without regard to case.

**ISRCH** Search a sorted integer vector for a given integer and return its index.

**SRCH** Search a sorted vector for a given scalar and return its index.

**SSRCH** Search a character vector, sorted in ascending ASCII order, for a given string and return its index.

## N6: Sorting

Sorting is the rearrangement of units of data (called *records*) so that a particular data field (the *key*) is in ascending or descending order. Although more common in business processing, this problem has also seen many scientific applications since Von Neumann coded sorting algorithms on the EDVAC in 1945. Techniques for sorting can be generally classified into two types — internal or external — depending upon whether all the data is stored in memory or not.

Internal sorting algorithms usually accept a one-dimensional array of keys and one or more "parallel" data arrays. The output of the program is either a set of rearranged data arrays (*active* sorting) or a permutation array $P$ (*passive* sorting). In the latter case the data are not reordered; instead one may find the $i^{th}$ element in sorted order in the $P(i)^{th}$ position. Thus, the values in the permutation array $P$ represent the ranks of the corresponding keys. In the classification scheme we also partition the software by the type of key, i.e., integer, real, or character. Good general purpose algorithms are those that minimize data comparisons and exchanges and do not have large workspace requirements. Such programs usually run in the time proportional to $n \log n$, where $n$ is the number of records. C.A.R. Hoare's QUICKSORT is generally considered the best. These considerations may change for vector processors, however.

External sorting methods are usually combinations of internal sorting and external merges. Here one is more interested in minimizing the amount of time spent on the input or output of data. Most computer systems provide external sorting utilities as part of the operating system, although not all will interface with Fortran.

### References

[K373] D. E. Knuth. *The Art of Computer Programming Volume 3: Sorting and Searching.* Addison-Wesley, Reading, Massachusetts, 1973.

[Wir76] N. L. Wirth. *Algorithms + Data Structures = Programs.* Prentice-Hall, Englewood Cliffs, NJ, 1976.

## N6a1 : Passive internal sorting (i.e. construct pointer array, rank)

*NAG Subprogram Library*

**M01DZF** Ranks arbitrary data according to a user-supplied comparison routine.

## N6a1a : Passive internal sorting of integers

*IMSL MATH/LIBRARY Subprogram Library*

**SVIBP** Sort an integer array by absolute values and return a pointer array.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**SVIGP** Sort an integer array by algebraic values and return a pointer array.

*NAG Subprogram Library*

**M01ACF** Sorts an array of pointers via a detached key sort to give the ascending order of a given integer array.

**M01ADF** Sorts an array of pointers via a detached key sort to give the descending order of a given integer array.

**M01DBF** Ranks a vector of integer numbers in ascending or descending order.

**M01DFF** Ranks the rows of a matrix of integer numbers in ascending or descending order.

**M01DKF** Ranks the columns of a matrix of integer numbers in ascending or descending order.

*PORT Subprogram Library*

**SRTPAI** Passively sorts integer data into ascending order.

**SRTPDI** Passively sorts integer data into descending order.

## N6a1b : Passive internal sorting of real data

*DATAPAC Subprogram Library*

**CODE** Codes the elements of the input vector X; 1.0 for minimum, 2.0 for next larger, etc.

**RANK** Ranks (in ascending order) the N elements of the single precision vector X, and puts the resulting N ranks into the vector XR.

*IMSL Subprogram Library*

**NMRANK** Numerical ranking.

**VSAR** Sorting of matrices (with options).

*IMSL MATH/LIBRARY Subprogram Library*

**SVRBP** Sort a real array by absolute values and return a pointer array.

**SVRGP** Sort a real array by algebraic values and return a pointer array.

*IMSL STAT/LIBRARY Subprogram Library*

**SCOLR** Sort columns of a real rectangular matrix using keys in rows.

**SVRGP** Sort a real array by algebraic values and return a pointer array.

*NAG Subprogram Library*

**M01AAF** Sorts an array of pointers via a detached key sort to give the ascending order of a given real array.

**M01ABF** Sorts an array of pointers via a detached key sort to give the descending order of a given real array.

**M01DAF** Ranks a vector of real numbers in ascending or descending order.

**M01DJF** Ranks the columns of a matrix of real numbers in ascending or descending order.

*PORT Subprogram Library*

**SRTPAR** Passively sorts real data into ascending order.

**SRTPDR** Passively sorts real data into descending order.

*Scientific Desk PC Subprogram Library*

**N6A2P** Sorts a real array R into ascending order. A pointer array is also returned.

**N6A2PD** Sorts a double precision array D into ascending order. A pointer array is also returned.

## N6a1c : Passive internal sorting of character data

*NAG Subprogram Library*

**M01DCF** Ranks a vector of character data in ASCII or reverse ASCII order of a specified substring.

*PORT Subprogram Library*

**SRTPAH** Passively sorts Hollerith data into ascending order.

**SRTPDH** Passively sorts Hollerith data into descending order.

## N6a2 : Active internal sorting

*SPSS Program Library*

**SORT CASES** Sorts numeric or alphanumeric data in either ascending or descending order according to one or more variables.

## N6a2a : Active internal sorting of integers

*Collected Algorithms of the ACM*

**A505** SPN: a Fortran subprogram implementing an insertion sort for linked lists. The insertion method is insensitive to the key distribution and is comparable in running time to Shellsort. (See W. Janko, ACM TOMS 2 (1976) pp. 204-206.)

*CMLIB Library (SSORT Sublibrary)*

**ISORT** Sorts an integer array in either increasing or decreasing order. Optionally another integer array can be carried along.

*IMSL MATH/LIBRARY Subprogram Library*

**SVIBN** Sort an integer array by absolute values.

**SVIBP** Sort an integer array by absolute values and return a pointer array.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**SVIGN** Sort an integer array by algebraic values.

**SVIGP** Sort an integer array by algebraic values and return a pointer array.

*MAGEV Cyber 205 Subprogram Library*

**ISORTDA** To sort the elements of a real or integer array in a non-increasing, or descending, order with respect to absolute value.

**QSORT** To sort the elements of a real or integer array in a non-decreasing, or ascending, order with respect to algebraic value.

**SORTR** To sort a linearly stored set of fixed length records, or equivalently, the columns of a matrix, into a non-decreasing (ascending) order with respect to the algebraic values of the elements of a separate real or integer array known as the key.

*NAG Subprogram Library*

**M01AGF** Sorts the rows of an integer matrix into ascending order of an index column.

**M01AHF** Sorts the rows of an integer matrix into descending order of an index column.

**M01ALF** Sorts a vector of integer numbers into ascending order and provides an index indicating the position of the sorted numbers in the original array.

**M01AMF** Sorts a vector of integer numbers into descending order and provides an index indicating the position of the sorted numbers in the original array.

**M01AQF** Sorts a vector of integer numbers into ascending order.

**M01ARF** Sorts a vector of integer numbers into descending order.

**M01CBF** Rearranges a vector of integer numbers into ascending or descending order.

*PORT Subprogram Library*

**SRTAI** Actively sorts integer data into ascending order.

**SRTDI** Actively sorts integer data into descending order.

*SAS Program Library*

**SORT** Sorts observations by one or more variables in ascending or descending order. Options: handle duplicate records, maintain order of observations with identical values of the sorting variables, several collating sequences.

## N6a2b : Active internal sorting of real data

*CMLIB Library (SSORT Sublibrary)*

**SSORT** Sorts an array X (of N real numbers) into increasing or decreasing order. An optional array Y is carried along with X.

*DATAPAC Subprogram Library*

**SORT** Sorts (in ascending order) the N elements of the vector X and puts the resulting N sorted values into the vector Y.

**SORTC** Sorts (in ascending order) the N elements of the vector X and rearranges the elements of the vector Y.

**SORTP** Sorts (in ascending order) the N elements of the vector X, puts the resulting N sorted values into the vector Y, and puts the position (in the original vector X) of each of the sorted values into the single precision vector XPOS.

*IMSL Subprogram Library*

**VSAR** Sorting of matrices (with options).

**VSODA** Sorting of columns of a double precision matrix in ascending order of keys in rows.

**VSORA** Sorting of columns of a real matrix into ascending order of keys in rows.

**VSRTA** Sorting of arrays by algebraic value.

**VSRTM** Sorting of arrays by absolute value.

**VSRTP** Sorting of arrays by absolute value, pointer array returned.

**VSRTR** Sorting of arrays by algebraic value, pointer array returned.

*IMSL MATH/LIBRARY Subprogram Library*

**SVRBN** Sort a real array by absolute values.

**SVRBP** Sort a real array by absolute values and return a pointer array.

**SVRGN** Sort a real array by algebraic values.

**SVRGP** Sort a real array by algebraic values and return a pointer array.

*IMSL STAT/LIBRARY Subprogram Library*

**SCOLR** Sort columns of a real rectangular matrix using keys in rows.

**SVRGN** Sort a real array by algebraic values.

*MAGEV Cyber 205 Subprogram Library*

**ISORTDA** To sort the elements of a real or integer array in a non-increasing, or descending, order with respect to absolute value.

**QSORT** To sort the elements of a real or integer array in a non-decreasing, or ascending, order with respect to algebraic value.

**SORTR** To sort a linearly stored set of fixed length records, or equivalently, the columns of a matrix, into a non-decreasing (ascending) order with respect to the algebraic values of the elements of a separate real or integer array known as the key.

*MINITAB Interactive System*

**ORDER** Sorts in ascending order the values in each of one or more vectors.

**SORT** Sorts a vector in ascending order and optionally carries along other vectors.

*NAG Subprogram Library*

**M01AEF** Sorts the rows of a real matrix into ascending order of an index column.

**M01AFF** Sorts the rows of a real matrix into descending order of an index column.

**M01AJF** Sorts a vector of real numbers into ascending order and provides an index indicating the position of the sorted numbers in the original array.

**M01AKF** Sorts a vector of real numbers into descending order and provides an index indicating the position of the sorted numbers in the original array.

**M01ANF** Sorts a vector of real numbers into ascending order.

**M01APF** Sorts a vector of real numbers into descending order.

**M01CAF** Rearranges a vector of real numbers into ascending or descending order.

*PORT Subprogram Library*

**SRTAR** Actively sorts real data into ascending order.

**SRTDR** Actively sorts real data into descending order.

*SAS Program Library*

**SORT** Sorts observations by one or more variables in ascending or descending order. Options: handle duplicate records, maintain order of observations with identical values of the sorting variables, several collating sequences, saved output.

*Scientific Desk PC Subprogram Library*

**N6A2** Sorts a real array R into ascending order.

**N6A2A** Sorts a real array R into ascending order by absolute value.

**N6A2AD** Sorts a double precision array D into ascending order by absolute value.

**N6A2D** Sorts a double precision array D into ascending order.

**N6A2P** Sorts a real array R into ascending order. A pointer array is also returned.

**N6A2PD** Sorts a double precision array D into ascending order. A pointer array is also returned.

## N6a2c : Active internal sorting of character data

*CMLIB Library (SSORT Sublibrary)*

**CSORT** Sorts a character array in either increasing or decreasing order. Optionally another character array can be carried along.

*NAG Subprogram Library*

**M01BAF** Sorts an integer vector containing character data into reverse alphabetic order.

**M01BBF** Sorts a vector containing character data into alphabetic order.

**M01BCF** Sorts selected columns (records) of an integer array containing character data, so that the elements of index rows (keys) are in reverse alphabetic order.

**M01BDF** Sorts selected columns (records) of an integer array containing character data, so that the elements of index rows (keys) are in alphabetic order.

**M01CCF** Rearranges a vector of character data so that a specified substring is in ASCII or reverse ASCII order.

*PORT Subprogram Library*

**SRTAH** Actively sorts Hollerith data into ascending order.

**SRTDH** Actively sorts Hollerith data into descending order.

*SAS Program Library*

**SORT** Sorts observations by one or more variables in ascending or descending order. Options: handle duplicate records, maintain order of observations with identical values of the sorting variables, several collating sequences, saved output.

## N7 : Merging

*MAGEV Cyber 205 Subprogram Library*

**HMERGE** To merge elements of two half-word vectors, or of a scalar and a vector, into a half-word result vector. The Cyber 205 hardware instruction MRGV is utilized.

**MRGSORT** To merge up to 4 already sorted subfiles residing in arrays into one sorted file. With subfile or file denoting an array containing a real or integer key (one 64-bit word per key element) of length N words, plus a two-dimensional array A(LR,N) containing the N records of length LR. The sorting order is non-decreasing, or ascending, with respect to the algebraic values of the key-elements.

## N8 : Permuting

*IMSL Subprogram Library*

**VSRTU** Interchange the rows or columns of a matrix using a permutation vector such as the one obtained from IMSL routines VSRTP or VSRTR.

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**PERMA** Permute the rows or columns of a matrix.

**PERMU** Rearrange the elements of an array as specified by a permutation.

*MAGEV Cyber 205 Subprogram Library*

**HREV** To reverse the elements of a half-word vector.

*NAG Subprogram Library*

**M01EAF** Rearranges a vector of real numbers into the order specified by a vector of ranks.

**M01EBF** Rearranges a vector of integer numbers into the order specified by a vector of ranks.

**M01ECF** Rearranges a vector of character data into the order specified by a vector of ranks.

**M01ZAF** Inverts a permutation, and hence converts a rank vector to an index vector, or vice versa.

**M01ZBF** Checks the validity of a permutation.

**M01ZCF** Decomposes a permutation into cycles, as an aid to re-ordering ranked data.

*PORT Subprogram Library*

**SRTRH** Rearranges Hollerith data according to permutation stored in IP.

**SRTRI** Rearranges integer data according to permutation stored in IP.

**SRTRR** Rearranges real data according to permutation stored in IP.

# P: Computational Geometry

Classified here is a small collection of computational geometry software providing both some fundamental geometric calculations (e.g., areas and volumes) and implementation of selected algorithms, including the convex hull (the smallest convex set containing a given set of points) and the Voronoi diagram and its dual (also known as Thiessen triangulation).

## References

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry.* Springer-Verlag, New York, 1985.

| **P :** | **Computational geometry** *(search also classes G and Q)* |
|---|---|

*Collected Algorithms of the ACM*

**A499** CONOPT: a subprogram which determines the contour scanning path for a two-dimensional region. The path is designed to help accelerate the propagation of edge effects when solving two-dimensional partial differential equations using iterative methods. (See W. Kinsner and E.D. Torre, ACM TOMS 2 (1976) pp. 82-86.)

**A523** CONVEX: a Fortran subroutine for determining which points of a planar set are vertices of the convex hull of the set. (See W. F. Eddy, ACM TOMS 3 (1977) pp. 411-412.)

**A550** PROPS and SRFINT: Fortran subroutines for computing surface area, centroid, volume, weight, moments, and products of inertia of solid polyhedra. The routines operate directly on geometric definitions consisting of vertex coordinates and face lists, in a format similar to that employed in graphics programs (See A.M. Messner and G.Q. Taylor, ACM TOMS 6 (1980) pp. 121-130.)

**A595** HC: a Fortran subprogram for finding one or more Hamiltonian circuits in a directed graph. (See S. Martello, ACM TOMS 9 (1983) pp. 131-138.)

**A624** A set of Fortran programs for triangulation and interpolation at arbitrarily distributed points in the plane. (See R.J. Renka, ACM TOMS 10 (1984) pp. 440-442.)

**A625** A Fortran subprogram which relates a general two-dimensional domain to a rectangular grid laid over it. (See J.R. Rice, ACM TOMS 10 (1984) pp. 443-452 and 453-462.)

*CGLIB Subprogram Library*

**ADNODE** Adds a node to a triangulation of a set of points in the plane producing a new triangulation.

**AREA** Given a sequence of points in the plane, this function computes the area bounded by the closed polygonal curve which passes through the points in the specified order.

**BNODES** Given a triangulation of a set of points in the plane, returns the nodes on the boundary of the convex hull of the set of points.

**DECOMP** Decomposes a not-necessarily convex polygon into a sequence of convex polygons.

**DELETE** Deletes a boundary edge from a triangulation of a set of points in the plane.

**EDGE** Given a triangulation of a set of points in the plane, forces an edge between two given points in the triangulation.

**TRMESH** Creates a Thiessen triangulation of n arbitrarily spaced points in the plane.

**VOR** Constructs a 2-d Voronoi diagram for a set of sites in a simply connected bounded planar polygonal region.

**VORU** Constructs a 2-d Voronoi diagram for a set of sites in the Euclidean plane.

*NAG Subprogram Library*

**D03MAF** Places a triangular mesh over a given 2-dimensional region. The region may have any shape, including one with holes.

*Scientific Desk PC Subprogram Library*

**P4MED** Computes the geometric median of a finite set S of points X belonging to n dimensional space. The geometric median is that point G such that the sum of $(X\text{-}G)^2$ over all points is less than the sum of $(X\text{-}Y)^2$ for all Y in the space.

# Q: Graphics

Computer graphics involves the generation, representation, manipulation, processing, or evaluation of graphics objects by a computer as well as the association of graphic objects with related nongraphic information residing in computer files. Graphic objects may be photographic images, or they may be created with the aid of a computer in the form of alphanumeric characters, special symbols, line drawings, or gray-shaded areas. Such artificially created objects may be rendered in black and white or in color.

Computer graphics is often classified into three subareas.

1. *Generative graphics.* This involves artificially created objects, usually in the form of line drawings. The main tasks to be performed here are picture construction and generation on a display surface, often a plotter or a display screen.

2. *Image analysis.* This involves processing of digitized representations via grey levels, of photographic images. The main tasks to be performed here are image enhancement, image evaluation, and image recognition.

3. *Cognitive graphics.* This involves the abstract models of graphic objects and relationships between them. An abstract model is the idea of an object regardless of its instantaneous appearance, e.g. a triangle is a triangle regardless of the length of its sides, location, or orientation. The main tasks to be performed here are recognition of abstract objects and extraction of these from an image. Cognitive graphics merges (1) and (2).

Software for generative graphics is the only type currently listed in this class. Output devices can be a line printer, a display screen, or a laser printer. Often graphics software can produce a file that can be shipped to a particular type printer without any additional processing. More often the file is in a device-independent form that must be post processed before it can be sent to a printer or a plotter. This is sometimes called a graphics metafile. Some software that supports display screen output is suitable for interactive graphics. That is, the programmer can develop an application that allows the user to input information while examining the screen. Software that permits a user to "zoom" into a plot by moving a box on the screen is a good example.

Users may have software control of graphics devices at a number of different levels. For example, one Fortran subroutine call might simply draw a line between two specified points. Another might plot a smooth curve through a given set of points. Yet another might draw a two-dimensional perspective plot on a surface in three dimensions represented by a table of function values, complete with title and axis labels.

Because graphical display of data is so important to statistical applications, several statistics packages have integrated graphics capabilities. These have not been classified here but in class L3.

## References

[Gil78] W. K. Giloi. *Interactive Computer Graphics.* Prentice-Hall, Englewood Cliffs, New Jersey, 1978.

| **Q :** | **Graphics** *(search also class L3)* |
|---|---|

*Collected Algorithms of the ACM*

**A531** GCONTR: a Fortran subroutine for determining sequences of points in the plane which may be used to draw contours through equal values of a surface. (See W.V. Snyder, ACM TOMS 4 (1978) pp. 290-294.)

**A626** TRICP: a Fortran subprogram for computing contours of a function defined by a set of irregularly distributed data points in the plane. (See A. Preusser, ACM TOMS 10 (1984) pp. 178-189.)

**A657** CON3D: Fortran software for plotting contour surfaces of a function of three variables. (See G. Sewell, ACM TOMS 14 (1988) pp. 42-44.)

*DATAPLOT Interactive System*

**CONTOUR PLOT** Generate a contour plot with user specified contour levels.

**FRACTAL PLOT** Generate a two-dimensional self-similar fractal figure via the random iteration algorithm. The user specifies the initial rotation angle, the x and y axis scale factors and translations, the secondary rotation angle, and probability weights for each map.

*DISSPLA Subprogram Library*

**DISSPLA** A large Fortran-callable library for producing publication-quality plots of two- and three-dimensional data. Capabilities include drawing of axes, grids, and labels, curve drawing, area shading, interpolation and smoothing, multiple plots per page, surface view plotting, contouring, scaling, projection and rotation, and elaborate character fonts. Many output devices are supported; device-independent output is also supported.

*IMSL Subprogram Library*

**USPLO** Printer plot of up to ten functions.

*IMSL MATH/LIBRARY Subprogram Library*

**PLOTP** Print a plot of up to ten sets of points.

*NCAR Subprogram Library*

**NCAR** A collection of Fortran 77 programs and subroutines for displaying scientific data. Capabilities include x-y coordinate plots, contour plots, world maps, solid-colored maps, 2D vector fields, drawing lines in 3D space, halftone (gray scale), background grids, surface views, iso-surfaces of 3D data, text in various fonts, and dashed lines with user-defined patterns. Many output devices are supported; device-independent output is also supported.

*PLOT10 Subprogram Library*

**PLOT10** A set of Fortran subprograms for two-dimensional general purpose computer graphics. Output is supported only on Tektronix terminals. Capabilities include: 1) low-level plotting calls such as move and draw (TCS), 2) higher level calls such as drawing bar plots (AGII), and 3) Calcomp preview routines. The latter are standard Calcomp library calls which allow users to preview pen plotter output on Tektronix terminals.

*Scientific Desk PC Subprogram Library*

**L3C3** Plots tabled values of up to five functions.

*TEMPLATE Subprogram Library*

**TEMPLATE** A device and computer-independent general-purpose, two and/or three dimensional Fortran-callable graphics system for static, interactive, and dynamic applications. Included are line drawing, text generation, coordinate system generation, windows, blanking regions, charting and graphing (e.g., contour and surface plots, business graphics), interactive input, graphics structures, display list operations, and color redefinition and selection.

*VOLKSGRAPHER Subprogram Library*

**VG** A collection of Fortran callable subroutines for two dimensional graphics and data plotting. Its main features are ease of use, portability, and interactive capabilities, such as automatic zooming, adding and moving text on the screen, etc. Output to a laser printer or pen plotter.

*ZETA Subprogram Library*

**ZETA** A set of Fortran-callable subroutines which provide access to the Zeta plotter, an off-line, 4 color, pen plotter. The library contains the standard Calcomp library routines with modest extensions.

# R: Service Routines

This chapter contains subprograms which perform fairly low-level utility functions such as error checking, error handling, and retrieval of information about machine characteristics. These programs are generally not of immediate interest to routine users of software libraries. However, they are quite important for those writing software intended to be portable and to be reliable on a wide variety of machines.

| **R :** | **Service routines** |
|---|---|

*IMSL MATH/LIBRARY and IMSL/STAT Subprogram Libraries*

**DTIME** Get time of day.

**IDYWK** Compute the day of the week for a given date.

**NDAYS** Compute the number of days from January 1, 1900, to the given date.

**NDYIN** Give the date corresponding to the number of days since January 1, 1900.

**TDATE** Get today's date.

*MAGEV Cyber 205 Subprogram Library*

**LOCF** To obtain the address of an array element.

**MAPFL** To map (part of) a common block into a local file, rather than letting the system map it to the drop-file.

**QUADIT** To provide a printed report about the utilization of system resources.

## R1: Machine-dependent constants

Modules classified in this class are useful for increasing the portability of programs. In Fortran, some degree of portability can be obtained by using a subset of Fortran acceptable to all compilers and by isolating all machine-dependent parameters so they are easily changed when moving to a new environment. The latter can be done in two distinct ways. The most direct is to place machine specific information in a visible place in your program and document clearly which machine parameters you are using and how they should be changed to move to another computer, or perhaps to another precision. The advantage of this is that your program becomes fully self-contained. An alternate approach is to use widely available subroutines which return information about the current machine environment. These allow you to write very portable programs. When machine information is required it is obtained "automatically" by evaluation of these functions within your program. Programs which supply this information are listed in this class.

The decision as to which of these approaches to use depends mostly on the volume of software that might have to be transported. The first seems quite simple for only one, or at most a few, routines. The second is very useful if a substantial amount of software is being moved, since all the changes are localized in the machine-constant routines. Even when moving a small block of code the latter is often better since it is easy to forget to make small but necessary changes in lower level routines.

### References

[BF80] W. S. Brown and S. I. Feldman. Environment parameters and basic functions for floating-point computation. *ACM Transactions on Mathematical Software*, 6(4):510–523, 1980.

[FHS78] P. A. Fox, A. D. Hall, and N. L. Schryer. The PORT mathematical subroutine library. *ACM Transactions on Mathematical Software*, 4(2):104–126, 1978.

## R1 : Machine-dependent constants

*Collected Algorithms of the ACM*

**A528** Three program packages which provide a framework for a portable Fortran subroutine library: machine-dependent constants, automatic error handling, and dynamic storage allocation using a stack. Developed for use with the PORT library. (See P.A. Fox, A.D. Hall, and N.L. Schryer, ACM TOMS 4 (1978) pp. 176-188.)

*CMLIB Library (MACHCON Sublibrary)*

**I1MACH** Provides integer machine dependent information, e.g. largest integer.

**R1MACH** Returns single precision machine-dependent constants (like largest allowed floating-point number) for the local machine environment.

*IMSL Subprogram Library*

**UGETIO** To retrieve current values and to set new values for input and output unit identifiers.

*IMSL MATH/LIBRARY, STAT/LIBRARY and SFUN/LIBRARY Subprogram Libraries*

**AMACH** Retrieve single-precision machine constants.

**IMACH** Retrieve integer machine constants.

**UMACH** Set or retrieve input or output device unit numbers.

*NAG Subprogram Library*

**X01AAF** Returns the value of the constant pi.

**X01ABF** Returns the value of Euler's constant, lambda.

**X02AAF** Returns the value $\epsilon$, where $\epsilon$ is the smallest positive real machine number such that $1+\epsilon > 1$.

**X02ABF** Returns the value of RMIN, the smallest positive real floating-point number such that RMIN and -RMIN are exactly representable on the computer.

**X02ACF** Returns the value of RMAX where RMAX is the largest computable real floating-point number such that RMAX and -RMAX are exactly representable on the computer.

**X02ADF** Returns the value of TOL, the ratio RMIN/EPS (see X02ABF and X02AAF).

**X02AEF** Returns the largest negative real value ENEG such that exp(ENEG) can be successfully evaluated without underflow by the compiler-supplied EXP routine (DEXP for double precision implementations).

**X02AFF** Returns the value of the largest positive permitted argument for EXP (or DEXP).

**X02AGF** Returns the value of the smallest positive representable floating-point number R such that -R, 1.0/R and -1.0/R are all computable without underflow or overflow.

**X02AHF** Returns the largest positive REAL argument for which the SIN and COS routines return a result with some meaningful accuracy.

**X02AJF** Returns the machine precision, i.e., $B^{1-P}/2$ if ROUNDS is true or $B^{1-P}$ otherwise (see the Chapter Introduction in the NAG Fortran Library Manual).

**X02AKF** Returns the smallest positive model number, i.e., $B^{EMIN-1}$.

**X02ALF** Returns the largest positive model number, i.e., $(1\text{-}B^{-P})B^{EMAX}$ (see the Chapter Introduction in the NAG Fortran Library Manual).

**X02AMF** Returns the safe range parameter as defined in the Chapter Introduction in the NAG Fortran Library Manual.

**X02BAF** Returns the value of BASE, the base of the arithmetic used on the computer.

**X02BBF** Returns the largest positive integer value.

**X02BCF** Returns the value of MAXPW2, the largest integer power to which 2.0 may be raised without overflow.

**X02BDF** Returns the value of MINPW2, the largest negative integer power to which 2.0 may be raised without underflow.

**X02BEF** Maximum number of decimal digits that can be represented.

**X02BHF** Returns the model parameter B (see the Chapter Introduction in the NAG Fortran Library Manual).

**X02BJF** Returns the model parameter P (see the Chapter Introduction in the NAG Fortran Library Manual).

**X02BKF** Returns the model parameter EMIN (see the Chapter Introduction in the NAG Fortran Library Manual).

**X02BLF** Returns the model parameter EMAX (see the Chapter Introduction in the NAG Fortran Library Manual).

**X02CAF** Returns an estimate of the active set size in a paged environment; in a non-paged environment, it returns zero.

**X02DAF** Switch for taking precautions to avoid underflow.

**X02DJF** Returns the model parameter ROUNDS (see the Chapter Introduction in the NAG Fortran Library Manual).

*PORT Subprogram Library*

**I1MACH** Provides integer machine dependent information, e.g. largest integer.

**R1MACH** Provides the single precision machine-dependent constants required to adapt PORT library programs to individual computers.

*Scientific Desk PC Subprogram Library*

**R1MAC** Returns single precision machine-dependent constants (like largest allowed floating-point number) for the local machine environment.

**R1MACI** Returns integer machine-dependent constants (like largest allowed integer) for the local machine environment.

| R2 : | Error checking (e.g., check monotonicity) |
|---|---|

*PORT Subprogram Library*

**MONOI** Test if an integer vector is monotone increasing or decreasing.

**MONOR** Test if a real vector is monotone increasing or decreasing.

**SMONOI** Test if an integer vector is strictly monotone increasing or decreasing.

**SMONOR** Test if a real vector is strictly monotone increasing or decreasing.

## R3: Error handling

Most debugging aids for the casual programmer are very system-dependent. Within large collections of programs such as the commercial libraries certain conventions have been established for reporting errors that occur. For the most part library routines do this by calling a subroutine which performs various actions depending on the severity of the error. Library users do not often need to know about these programs. Programmers who are developing a package for use by others may wish to take advantage of some of these utilities, however. One such error handler is XERROR, a public-domain package available in (and used by) CMLIB.

In some cases it is necessary for casual users to change some of the defaults in use by the error handler. For example, every error may cause the program to abort. This is easily changed, and the way that this is done depends upon the particular library in use. For detailed information about the error handlers in the IMSL, NAG, and PORT libraries, one should consult the appropriate library reference manual (see the *Application Package Reference* section of GAMS).

## R3 : Error handling

*Collected Algorithms of the ACM*

**A528** Three program packages which provide a framework for a portable Fortran subroutine library: machine-dependent constants, automatic error handling, and dynamic storage allocation using a stack. Developed for use with the PORT library. (See P.A. Fox, A.D. Hall, and N.L. Schryer, ACM TOMS 4 (1978) pp. 176-188.)

## R3a : Set criteria for fatal errors

*CMLIB Library (XERROR Sublibrary)*

**XSETF** Set KONTRL for XERROR, default is 2.

*IMSL Subprogram Library*

**UERSET** Set message level for IMSL routine UERTST.

*PORT Subprogram Library*

**ENTSRC** Saves current recovery mode status and sets a new one for PORT library programs.

**RETSRC** Test and reset error recovery mode for PORT library programs.

*Scientific Desk PC Subprogram Library*

**R3A** Accesses or changes an indicator which causes specific types of error messages to be printed.

## R3b : Set unit number for error messages

*CMLIB Library (XERROR Sublibrary)*

**XSETUA** Set up to 5 output unit numbers.

**XSETUN** Set one output unit number.

*IMSL Subprogram Library*

**UGETIO** To retrieve current values and to set new values for input and output unit identifiers.

*NAG Subprogram Library*

**X04AAF** Returns the value of the current error message unit number, or sets the current error message unit number to a new value.

**X04ABF** Returns the value of the current advisory message unit number, or sets the current advisory message unit number to a new value.

*Scientific Desk PC Subprogram Library*

**R3B** Accesses the current input and output unit reference numbers for error messages. Can also be used to reset the unit reference numbers.

**R3c : Other utilities for error handling**

*CMLIB Library (XERROR Sublibrary)*

**NUMXER** Get most current message number.

**XERABT** Terminate run and print traceback. (Requires system dependent programming to execute properly, else just STOPs.)

**XERCLR** Clear current message number.

**XERCTL** Perform special error processing of one message.

**XERDMP** Print error summary and clear tables.

**XERMAX** Set limit of MAX times each message can be printed.

• **XERROR** Process an error message.

**XERRWV** Process a message with numeric values.

**XGETF** Get current value of KONTRL.

**XGETUA** Get current output unit numbers.

**XGETUN** Get current output unit number.

*IMSL Subprogram Library*

**UERTST** Print a message reflecting an error condition.

*IMSL MATH/LIBRARY, STAT/LIBRARY and SFUN/LIBRARY Subprogram Libraries*

**ERSET** Set error handler default print and stop actions.

**IERCD** Retrieve the code for an informational error.

*NAG Subprogram Library*

**P01AAF** Returns the value of IERROR or terminates the program, printing a failure message.

**P01ABF** Either returns the value of IERROR (soft failure), or terminates execution of the program (hard failure). Diagnostic messages may be output.

*PORT Subprogram Library*

**ENTER** Save current error recovery mode and storage allocation status for PORT library programs.

**EPRINT** Print the current error message if the program is in the error state for PORT library programs.

**ERROFF** Turns off the error state for PORT library programs.

**LEAVE** Restores prior error recovery mode and resets the stack for PORT library programs.

**NERROR** Provides the current error number for PORT library programs.

**SETERR** Sets the error indicator and depending on options prints a message and provides a dump for PORT library programs.

*Scientific Desk PC Subprogram Library*

**R3C** Prints an error message on the current output device, that is, the device associated with the output reference number supplied by R3B. The type of error message to print is determined by a call to R3A.

## R4: Documentation retrieval

This section contains modules whose sole purpose is to provide information about programming conventions within a library or package.

| **R4 :** | **Documentation retrieval** |
|---|---|

*Collected Algorithms of the ACM*

**A606** NITPACK: Fortran subprograms for implementing decision trees. NITREE converts a paper tree into a format that may be transmitted to a user at a computer terminal. NIT performs the transmission. (see P.W. Gaffney et al., ACM TOMS 9 (1983) pp. 418-426.)

*IMSL Subprogram Library*

**UHELP** Display methods of obtaining information on IMSL conventions regarding various subjects. Provide means for individual sites to supply users with site specific information.

**UHELP1** Write information regarding IMSL conventions and notation to an output file.

**UHELP2** Write information regarding IMSL input and output conventions.

**UHELP3** Write information regarding IMSL error detecting facilities.

**UHELP4** Write information regarding matrix/vector storage modes used in IMSL subroutines.

# S: Software Development Tools

This class is designed to contain software tools which ease the process of program development and maintenance. The types of jobs performed by such tools are program transformation (e.g., convert to double precision), static analysis (e.g., flow analysis, interface analysis), and dynamic analysis (e.g., tracing, timing, assertion checking).

## S1 : Program transformation tools

*Collected Algorithms of the ACM*

**A532** A software package for roundoff analysis of noniterative numerical methods. (See W. Miller and D. Spooner, ACM TOMS 4 (1978) pp. 388-390.)

**A622** A simple macroprocessor for use in manipulating Fortran code as well as for general text processing. (See J.R. Rice, C. Ribbens, and W.A. Ward, ACM TOMS 10 (1984) pp. 410-416.)

## S2 : Static program analysis tools

*Collected Algorithms of the ACM*

**A605** PBASIC: a Fortran program which checks a BASIC program for adherence to the American National Standard Minimal Standard for BASIC. (See T.R. Hopkins, ACM TOMS 9 (1983) pp. 391-394.)

## S3 : Dynamic program analysis tools

*IMSL MATH/LIBRARY and IMSL STAT/LIBRARY Subprogram Libraries*

**CTIME** Return CPU time used in seconds.

*MAGEV Cyber 205 Subprogram Library*

**QUADIT** To provide a printed report about the utilization of system resources.

# Z: Other Software

This class contains software which does not fit anywhere else.

| **Z :** | **Other** |
|---|---|

*Collected Algorithms of the ACM*

**A536** PURDY: a Fortran subroutine for evaluating Purdy's irreversible enciphering function. It serves as a machine independent model for studying the evaluation of polynomials mod P and for the implementation of more efficient machine dependent system utility programs for enciphering passwords. (See H.D. Knoble, ACM TOMS 5 (1979) pp. 108-111.)

**A568** PDS: a portable file directory system implemented in Fortran. (See D.R. Hanson, ACM TOPLAS vol. 3, pp. 162-167)

**A607** TES (Text Exchange System): transportable Fortran programs for management and exchange of programs and other text. TES defines a format for information storage and includes two programs, the first creates, reads, and maintains TES files, and the second permits TES to be distributed on a tape in TES format. (See W.V. Snyder and R.J. Hanson, ACM TOMS 9 (1983) pp. 427-440.)

**A620** Data files containing references and keywords for the Collected Algorithms from ACM. (See J.R. Rice and R.J. Hanson, ACM TOMS 10 (1984) pp. 359-360.)

**A622** A simple macroprocessor for use in manipulating Fortran code as well as for general text processing. (See J.R. Rice, C. Ribbens, and W.A. Ward, ACM TOMS 10 (1984) pp. 410-416.)

# Module Dictionary

*The module is the smallest unit cataloged in GAMS. A module may be a Fortran subprogram, a stand-alone program, or a command in an application system. In this section we present a summary description of each module in alphabetical order. The following legend is an explanation of the information included in the summaries.*

**Legend**

**NAME** A brief description of what the module does. *Information on the type of module.* **Classes:** one or more problem classifications. **Usage:** how to use the module. **Also see:** names of related modules. **Precision:** the precision in which the computation is done. **Availability:** a list of local computers on which the software has been implemented.

*The following points should be noted.*

**Type** *indicates (a) whether the module is portable or proprietary, (b) whether the module is a subprogram, a stand-alone program, or a command in an interactive system, and (c) the name of the package containing the module. The use of proprietary software is governed by a licensing agreement, while the use of software designated as portable is unrestricted. If the module is a subprogram the language in which it is programmed is also given.*

**Usage** *tells what a user needs to write to invoke the module. For subprograms this is a sample subprogram call. If the number of parameters is large, only the number is listed.*

**Precision** *indicates the precision in which the computation is done, i.e., single, double, or half. If the precision is machine-dependent, then this field is omitted, and information about precision is given under* **Availability** *instead.*

**Availability** *lists NIST computers on which the module is available. Additional notes about the implementation of the module on specific computers may also be found here. Specific information about how to access application packages on each NIST computer can be found in the GAMS Application Package Reference. The possible computers are as follows.*

**855NOS** *CYBER 180/855 (NOS operating system) located in Gaithersburg, MD. For further information call 301-975-2968 (FTS-879-2968).*

**855VE** *CYBER 180/855 (NOS/VE operating system) located in Gaithersburg, MD. For further information call 301-975-2968 (FTS-879-2968).*

**205** *CYBER 200/205 (VSOS operating system) located in Gaithersburg, MD. For further information call 301-975-2968 (FTS-879-2968).*

**CAMVAX** *VAX 11/785 (VMS operating system) located in Gaithersburg, MD. For further information call 301-975-3830 (FTS-879-3830). Access restricted.*

**PC** *PCs and clones (DOS operating system). For further information call the Micro-Resource Site, Gaithersburg, MD, 301-975-3837 (FTS-879-3837).*

**840NOS** *CYBER 180/840 (NOS operating system) located in Boulder, CO. For further information call 303-497-3566 (FTS-320-3566).*

# A

**A02AAF.** Square root of a complex argument. *Proprietary Fortran subroutine in NAG.* **Classes:** C2 **Usage:** CALL A02AAF (XR, XI, YR, YI) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**A02ABF.** Modulus of a complex number. *Proprietary Fortran function in NAG.* **Classes:** A4a **Usage:** R = A02ABF(XR, XI) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**A02ACF.** Quotient of complex numbers. *Proprietary Fortran subroutine in NAG.* **Classes:** A4a **Usage:** CALL A02ACF (XR, XI, YR, YI, ZR, ZI) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**A10.** Use inverse interation via Gaussian elimination. Illustrates use of algorithm A10GII. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 90-91.) *Portable Fortran software in NASHLIB library.* **Classes:** D4a2 **Usage:** CALL A10GII(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A11A12.** To standardize a complex vector and compute the residuals of a complex eigenvalue and eigenvector. Illustrates use of algorithms A11VS and A12CVR. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 94-95.) *Portable Fortran software in NASHLIB library.* **Classes:** D4a2 **Usage:** CALL A11VS(*4 parameters*), CALL A12CVR(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A13.** Use the singular value decomposition to find the eigenvalues and eigenvectors of a real symmetric matrix. Illustrates use of algorithm A13ESV. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 101-103.) *Portable Fortran software in NASHLIB library.* **Classes:** D4a1 **Usage:** CALL A13ESV(N, A, NA, EPS, H, ISWP, IPR, Z) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A14A15.** Use the Jacobi algorithm to find the eigenvalues and eigenvectors of a real symmetric matrix. Use the Jacobi algorithm to solve the generalized eigenvalue problem for real symmetric matrices. Illustrates use of algorithms A14JE and A15GSE. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 106-115.) *Portable Fortran software in NASHLIB library.* **Classes:** D4a1, D4b1 **Usage:** CALL A14JE(*9 parameters*), CALL A15GSE(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A16.** Perform a grid or equal-interval line search. Illustrates use of algorithm A16GS. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, page 125.) *Portable Fortran software in NASHLIB library.* **Classes:** G1a2 **Usage:** CALL A16GS(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A17.** Line search using parabolic inverse interpolation. Illustrates use of algorithm A17LS. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 117-118.) *Portable Fortran software in NASHLIB library.* **Classes:** G1a1a, G1a2 **Usage:** CALL A17LS(B, ST, FUNS, IFN, NOCOM, IPR) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A18.** Finds roots using false position and bisection. Illustrates use of algorithm A18RF. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 136-137.) *Portable Fortran software in NASHLIB library.* **Classes:** F1b **Usage:** CALL A18RF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A19A20.** Finds the minimum of a function of n variables using a Nelder Mead algorithm and axial search. Illustrates use of algorithms A19NM and A20AS. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 146-149.) *Portable Fortran software in NASHLIB library.* **Classes:** G1b2 **Usage:** CALL A19NM(*11 parameters*), CALL A20AS(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A1A2.** Compute singular value decomposition and use it to solve linear least squares problem. Illustrates use of algorithms A1SVD and A2LSVD. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 30-34.) *Portable Fortran software in NASHLIB library.* **Classes:** D6, D9a1 **Usage:** CALL A1SVD(*9 parameters*), CALL A2LSVD(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A21.** Finds the minimum of a function of n variables using a variable metric algorithm. Illustrates use of algorithm A21VM. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 159-160.) *Portable Fortran software in NASHLIB library.* **Classes:** G1b1b **Usage:** CALL A21VM(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A22.** Finds the minimum of a function of n variables using a conjugate gradient algorithm. Illustrates use of algorithm A22CGM. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 165-167.) *Portable Fortran software in NASHLIB library.* **Classes:** G1b1b **Usage:** CALL A22CGM(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A23.** Use Marquardt algorithm to find minimum of a sum of squares (nonlinear least squares). Illustrates use of algorithm A23MRT. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 175-177.) *Portable Fortran software in NASHLIB library.* **Classes:** K1b1a2 **Usage:** CALL A23MRT(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A24.** Use conjugate gradient algorithm to find the solution of a set of linear equations with nonnegative definite coefficient matrix. Illustrates use of algorithm A24CG. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 197-198.) *Portable Fortran software in NASHLIB library.* **Classes:** D2b1b **Usage:** CALL A24CG(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A25.** Use the conjugate gradient algorithm to find the minimum of a Rayleigh quotient. Illustrates use of algorithm A25RQM. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 206-207.) *Portable Fortran software in NASHLIB library.* **Classes:** D4a1 **Usage:** CALL A25RQM(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A3.** Givens reduction of a real rectangular matrix. Illustrates use of algorithm A3GR. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 42-43.) *Portable Fortran software in NASHLIB library.* **Classes:** D5 **Usage:** CALL A3GR(M, N, A, NA, Q, NQ, EPS, SAVEQ) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A493.** RPOLY: a subroutine for finding all the zeros of a real polynomial using an algorithm of Jenkins and Traub. (See M.A. Jenkins, ACM TOMS 1 (1975) pp. 178-189.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F1a1 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A494.** PDEONE: solution of systems of nonlinear parabolic partial differential equations in one space dimension using the method of lines. (See R.F. Sincovec and N.K. Madsen, ACM TOMS 1 (1975) pp. 261-263.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2a1a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A495.** CHEB: a subprogram for the solution of overdetermined systems of linear algebraic equations in the Chebyshev norm using a variant of the simplex method. (See I. Barrodale and C. Phillips, ACM TOMS 1 (1975) pp. 264-270.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9a2 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A496.** LZHES/LZIT: subprograms for finding all the eigenvalues and (optionally) eigenvectors of the generalized eigenvalue problem for complex matrices. Based upon the LZ algorithm. (See L.C. Kaufman, ACM TOMS 1 (1975) pp. 271-281.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4b4 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A497.** DMRODE: a subprogram for the automatic integration of functional differential equations, such as retarded ordinary differential equations, Volterra integro-differential equations, and difference differential equations. (See K.W. Neves, ACM TOMS 1 (1975) pp. 369-371.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1a1a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A498.** AIRY: a subprogram to evaluate the Airy functions Ai(z), Bi(z) and their derivatives for all real values of z within computer capability. Based on Chebyshev series approximations. (See P.J. Prince, ACM TOMS 1 (1975) pp. 372-379.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10d **Precision:** Single **Availability:** 855NOS (In source form only.)

**A499.** CONOPT: a subprogram which determines the contour scanning path for a two-dimensional region. The path is designed to help accelerate the propagation of edge effects when solving two-dimensional partial differential equations using iterative methods. (See W. Kinsner and E.D. Torre, ACM TOMS 2 (1976) pp. 82-86.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b4, P **Precision:** Single **Availability:** 855NOS (In source form only.)

**A4A4A.** Givens reduction, singular value decomposition and application to solve a linear least squares problem. Illustrates use of algorithms A4A and A4LSGS. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 47-49.) *Portable Fortran software in NASHLIB library.* **Classes:** D6, D9a1 **Usage:** CALL A4LSGS(*14 parameters*), CALL ROTN(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A4BMD.** Complex multiply. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** A4 **Usage:** CALL A4BMD(X1R, X1I, X2R, X2I, X3R, X3I) **Precision:** Double **Availability:** PC.

**A4BMOD.** Complex modulus. *Proprietary Fortran function in the Scientific Desk.* **Classes:** A4 **Usage:** R = A4BMOD(ZR, ZI)· **Precision:** Double **Availability:** PC.

**A500.** MINI: a subprogram for finding an unconstrained minimum of a multivariate function f(x) to within the accuracy norm(g(x)).lt.eps, where g(x) is the gradient of f(x) and eps is the required accuracy provided by the user. Based on a quasi-Newton method suggested by Shanno and Phua. (See D.F. Shanno and K.H. Phua, ACM TOMS 1 (1975) pp. 87-94.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G1b1b **Precision:** Double **Availability:** 855NOS (In source form only.)

**A501.** APPROX/EXCH: subprograms for computing the best polynomial approximation to a discrete one-dimensional data set in the Chebyshev (minimax) sense. (See J.C. Simpson, ACM TOMS 2 (1976) pp. 95-97.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K2 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A502.** DERPAR: a Fortran subprogram for the evaluation of the dependence of the solution of a nonlinear system on a parameter. The modified method of Davidenko, which applies the implicit function theorem, is used in combination with Newton's method and Adam's integration formulas. (See M. Kubicek, ACM TOMS 2 (1976) pp. 98-107.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Usage:** CALL DERPAR(*20 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A503.** IESIMP and IEGAUS: Fortran subprograms for solving one-dimensional linear Fredholm integral equations of the second kind. The routines are based on the Nystrom method using Simpson's and Gauss quadrature formulas, respectively. The resulting linear systems are solved using an iterative method. (See K. Atkinson, ACM TOMS 2 (1976) pp. 196-199.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I3 **Precision:** Double **Availability:** 855NOS (In source form only.)

**A504.** GERK: a Fortran subprogram to solve nonlinear systems of ordinary differential equations when it is important to have a global error estimate. Integrations are performed on different mesh spacings and global extrapolation is applied to provide an estimate of the global error in the more accurate solution. The integrations are done using Runge-Kutta-Fehlberg methods of 4th and 5th order. (See L.F. Shampine and H.A. Watts, ACM TOMS 2 (1976) pp. 200-203.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1a1a **Usage:** CALL GERK(*11 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A505.** SPN: a Fortran subprogram implementing an insertion sort for linked lists. The insertion method is insensitive to the key distribution and is comparable in running time to Shellsort. (See W. Janko, ACM TOMS 2 (1976) pp. 204-206.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** N6a2a **Usage:** CALL SPN(K, L, II, JJ, MIN) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A506.** HQR3: a Fortran subprogram which reduces an upper Hessenberg matrix to quasi-triangular form by unitary similarity transformations to yield the eigenvalues of the matrix in order of magnitude. HQR3 can be used with EISPACK routines ORTHES and ORTRAN to compute the eigenvalues of a real general matrix. (See G.W. Stewart, ACM TOMS 2 (1976) pp. 275-280.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4c2b **Usage:** CALL HQR3(*11 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A507.** QUINAT: an Algol procedure to determine the interpolating quintic natural spline interpolant to a given set of data points. (See J.G. Herriot and C.H. Reinsch, ACM TOMS 2 (1976) pp. 281-289.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E1a **Usage:** R = QUINAT(*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A508.** REDUCE: a Fortran subprogram for reducing the bandwidth and profile of sparse symmetric matrices using row and column permutations. (See H.L. Crane et al.,ACM TOMS 2 (1976) pp. 375-377.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2e **Usage:** CALL REDUCE(*11 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A509.** A hybrid algorithm for reducing the bandwidth and profile of sparse symmetric matrices. Presented as a modification of algorithm 508 (Fortran subprogram REDUCE). (See N.E. Gibbs, ACM TOMS 2 (1976) pp. 378-387.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2e **Precision:** Single **Availability:** 855NOS (In source form only.)

**A510.** STL2: a Fortran subprogram which determines a piecewise linear approximation within specified tolerances of given data points which is composed of the fewest line segments. The approximant need not be continuous, and distinct tolerances may be specified for each data point. (See D.G. Wilson, ACM TOMS 2 (1976) pp. 388-391.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K4 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A511.** IBESS and JBESS: CDC 6600 Fortran subroutines for Bessel functions Iv(x) and Jv(x), for real x≥0, and real v≥0. (See D.E. Amos, S.L. Daniel, and M.K. Weston, ACM TOMS 3 (1977) pp. 93-95.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10a3, C10b3 **Usage:** CALL IBESS (*5 parameters*) AND CALL JBESS (*4 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A512.** FACTOR, RHS, and SOLVE: Fortran subroutines for solving symmetric positive definite periodic quindiagonal systems of linear equations. (See A. Benson, and D.J. Evans, ACM TOMS 3 (1977) pp. 96-103.) *Portable Fortran* *software in Collected Algorithms of the ACM.* **Classes:** D2b2, I2b4b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A513.** TRANS: a Fortran subroutine for in-situ matrix transposition which makes use of the cyclic structure of the transposition mapping. This is a revision of Algorithm 380. (See E.G. Cate and D.W. Twigg, ACM TOMS 3 (1977) pp. 104-110.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D1b3 **Usage:** CALL TRANS (*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A514.** Algol procedures for piecewise cubic interpolation using local data. (See M.R. Ellis and D.H. McLain, ACM TOMS 3 (1977) pp. 175-179.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E1a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A515.** COMB: a Fortran subroutine which generates a vector from a lexicographical index. That is, let $C_1, C2, \ldots, C_m$ be the set of combinations of n items taken p at a time arranged in lexographical order. Given an integer $i$, this routine finds $C_i$. (See B.P. Buckles and M. Lybanon, ACM TOMS 3 (1977) pp. 180-182.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** B, C1 **Usage:** CALL COMB (N, P, L, C) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A516.** RANKCI: a Fortran subroutine for obtaining confidence intervals and point estimates based on ranks in the two-sample location problem. (See J.W. McKean and T.A. Ryan, Jr., ACM TOMS 3 (1977) pp. 183-185.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L4b1b **Usage:** CALL RANKCI (*11 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A517.** CONDIT and QR2NOZ: a Fortran subroutines for computing the condition numbers of matrix eigenvalues without computing eigenvectors. (See S.P. Chan, R. Feldman, and B.N. Parlett, ACM TOMS 3 (1977) pp. 186-203.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4c2b **Usage:** CALL CONDIT (*7 parameters*) AND CALL QR2NOZ (*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A518.** VMISES: a Fortran function which computes the left tail area of the Von Mises distribution, which is equal to the incomplete modified Bessel function of the first kind and zero-th order ($I_0$). (See G.W. Hill, ACM TOMS 3 (1977) pp. 279-284.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10b1, L5a1v **Usage:** R = VMISES(T, VK) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A519.** RAKK, DURB, and EPST: Fortran subroutines for computing Kolmogorov-Smirnov probabilities with arbitrary boundaries. RAKK is a generalization of Massey's method. DURB is Durbin's method. EPST is the Epanechnikov, Steck method. (See R. Kallman, ACM TOMS 3 (1977) pp. 285-294.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L5a1u **Precision:** Single **Availability:** 855NOS (In source form only.)

**A520.** ARSME: a Fortran subroutine for solving a resource constrained network scheduling problem for the case in which the activities may be arbitrarily interrupted and restarted later with no increase in activity duration. The method used is an automatic revised simplex method. (See J. Weglarzet et al., ACM TOMS 3 (1977) pp. 295-300.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2c5 **Usage:** CALL ARSME (*15 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A521.** INERFC: a Fortran subroutine for evaluating repeated integrals of the coerror function. (See W. Gautschi, ACM TOMS 3 (1977) pp. 301-302.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C8a **Usage:** CALL INERFC (*6 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A522.** ESOLVE: a Fortran subroutines for the exact solution of systems of linear equations with multiple-precision integer coefficients by congruence techniques. (See S. Cabay and T.P.L. Lam, ACM TOMS 3 (1977) pp. 404-410.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a1 **Usage:** CALL ESOLVE (*13 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A523.** CONVEX: a Fortran subroutine for determining which points of a planar set are vertices of the convex hull of the set. (See W. F. Eddy, ACM TOMS 3 (1977) pp. 411-412.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** P **Usage:** CALL CONVEX (*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A524.** MP: Fortran subroutines for performing multiple precision floating point arithmetic and evaluating elementary and special functions. The subroutines are machine-independent and the precision is arbitrary, subject to storage limitations. (See R.P. Brent, ACM TOMS 4 (1978) pp. 71-81.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** A3c **Precision:** Single **Availability:** 855NOS (In source form only.)

**A525.** ADAPT: a Fortran subroutine for approximating a user-defined function by a piecewise polynomial of specified smoothness and degree. The user selects the accuracy required of the approximant as well as the norm by which the error is to be measured. (See J.R. Rice, ACM TOMS 4 (1978) pp. 82-94.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K1a1a1, K2, K3, K4 **Usage:** CALL ADAPT (*5 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A526.** IDBVIP and IDSFFT: Fortran subprograms for bivariate interpolation and smooth surface fitting for irregularly distributed data points. (See H. Akima, ACM TOMS 4 (1978) pp. 160-164.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E2b **Usage:** CALL IDBVIP (*12 parameters*) AND CALL IDSFFT (*13 parameters*)

**Precision:** Single **Availability:** 855NOS (In source form only.)

**A527.** GMA, GMAS, and KPICK: Fortran subroutines for linear systems arising from 5-point discretizations of separable or constant coefficient elliptic boundary-value problems on rectangular domains. A Dirichlet, Neumann, or mixed boundary condition may be independently specified on each side of the rectangle, or periodic boundary conditions may be specified on opposing sides. Implements the generalized marching algorithm. (See R.E. Bank, ACM TOMS 4 (1978) pp. 165-176.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1a, I2b1a2, I2b4b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A528.** Three program packages which provide a framework for a portable Fortran subroutine library: machine-dependent constants, automatic error handling, and dynamic storage allocation using a stack. Developed for use with the PORT library. (See P.A. Fox, A.D. Hall, and N.L. Schryer, ACM TOMS 4 (1978) pp. 176-188.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** R1, R3, N4 **Availability:** 855NOS (In source form only.)

**A529.** MC13D: a Fortran subroutine for finding symmetric permutations to block triangular form. That is, given the column numbers of the nonzeros in each row of a sparse matrix, this subroutine finds a symmetric permutation that makes the matrix block lower triangular. (See I.S. Duff and J.K. Reid, ACM TOMS 4 (1978) pp. 189-192.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2e **Usage:** CALL MC13D(*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A530.** TRIZD, IMZD, and TBAKZD: FORTRAN subroutines for computing the eigenvalues and eigenvectors of real skew-symmetric matrices or real tridiagonal symmetric matrices with constant diagonals. Based upon orthogonal similarity transformations. (See R.C. Ward and L.J. Gray, ACM TOMS 4 (1978) pp. 286-289.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4a1, D4a2, D4a5 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A531.** GCONTR: a Fortran subroutine for determining sequences of points in the plane which may be used to draw contours through equal values of a surface. (See W.V. Snyder, ACM TOMS 4 (1978) pp. 290-294.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** Q **Usage:** CALL GCONTR (*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A532.** A software package for roundoff analysis of noniterative numerical methods. (See W. Miller and D. Spooner, ACM TOMS 4 (1978) pp. 388-390.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** A3a, S1 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A533.** NSPIV: a Fortran subroutine for solving a sparse systems of linear equations by sparse Gaussian elimination with partial pivoting. (See A.H. Sherman, ACM TOMS 4 (1978) pp. 391-398.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a4 **Usage:** CALL NSPIV (*13 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A534.** STINT: A Fortran subprogram for integrating a set of first order ordinary differential equations using stiffly stable, cyclic composite linear multistep methods. (See J.M. Tendler, T.A. Bickart, and Z. Picel, ACM TOMS 4 (1978) pp. 399-403.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1a2 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A535.** CQZHES, CQSVEC, and CQZVAL: Fortran subroutines implementing the QZ algorithm for solving the generalized eigenvalue problem for complex matrices. (See B.S. Garbow, ACM TOMS 4 (1978) pp. 404-410.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4b4 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A536.** PURDY: a Fortran subroutine for evaluating Purdy's irreversible enciphering function. It serves as a machine independent model for studying the evaluation of polynomials mod P and for the implementation of more efficient machine dependent system utility programs for enciphering passwords. (See H.D. Knoble, ACM TOMS 5 (1979) pp. 108-111.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** B, Z **Usage:** CALL PURDY (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A537.** CHARMA: a Fortran subprogram for calculating the characteristic values of Mathieu's differential equation for odd or even solutions. (See W.R. Leeb, ACM TOMS 5 (1979) pp. 112-117.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C17, I1b3 **Usage:** CALL CHARMA (*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A538.** SIMITZ: a Fortran subroutine for computing eigenvalues largest in magnitude and corresponding eigenvectors of a real matrix symmetric relative to a user-defined inner product. Based upon the simultaneous iteration algorithm. (See P.J. Nikolai, ACM TOMS 5 (1979) pp. 118-125.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4b1 **Usage:** CALL SIMITZ (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A539.** BLAS (Basic Linear Algebra Subprograms): a package of 38 Fortran subprograms, to perform basic operations of numerical linear algebra, including dot product, Givens transformations, vector copy and swap, vector norm, vector scaling, and determination of the index of the vector component of the largest magnitude. Includes test programs and assembly language versions for the IBM 360, CDC 6000, and Univac 1100. (See C.L. Lawson et al., ACM TOMS 5 (1979) pp. 324-325.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D1a

**Availability:** 855NOS (In source form only.)

**A540.** PDECOL: a Fortran subprogram for solving coupled systems of nonlinear partial differential equations in one space and one time dimension. The solution method uses finite element collocation based upon piecewise polynomials for spatial discretization. The time discretization is performed by general-purpose software for ordinary initial value problems. (See N.K. Madsen and R.F. Sincovec, ACM TOMS 5 (1979) pp. 326-351.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2a1a, I2a2 **Usage:** CALL PDECOL (*13 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A541.** FISHPAK: Fortran subroutines for solving separable elliptic partial differential equations. Drivers are available for the Helmholtz equation in Cartesian, polar, surface spherical coordinates, cylindrical and interior spherical coordinates. In addition, subprograms for solving systems of linear equations resulting from finite difference approximations to general separable problems are included. (See P.N. Swarztrauber and R.A. Sweet, ACM TOMS 5 (1979) pp. 352-364.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1a, I2b4b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A542.** GAM: a Fortran subroutine based on Taylor's series and continued fractions for evaluating Tricomi's incomplete gamma function and the complementary incomplete gamma function. (See W. Gautschi, ACM TOMS 5 (1979) pp. 482-489.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C7e **Usage:** CALL GAM (*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A543.** FFT9: a Fortran subroutine for the Dirichlet problem for the Helmholtz equation on a rectangle. The program is based upon 4th and 6th order accurate 9-point finite difference approximations and fast Fourier solution techniques. (See E.N. Houstis and T.S. Papatheodorou, ACM TOMS 5 (1979) pp. 490-493.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A544.** L2A and L2B: Fortran subroutines for solving weighted least squares problems by modified Gram-Schmidt with iterative refinement. The types of problems solved include overdetermined and underdetermined systems of linear equations, and problems where the solution is subject to linear equality constraints. The covariance matrix of the solution vector is computed. (See R.H. Wampler, ACM TOMS 5 (1979) pp. 494-499.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9b1 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A545.** CMFFT and RMFFT: Fortran subprograms for computing real and complex fast Fourier transforms of a one-dimensional or multidimensional data set. The programs are designed to minimize I/O for the case where the data lie on mass storage. (See D. Fraser, ACM TOMS 5 (1979) pp. 500-517.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** J1a1, J1a2, J1b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A546.** SOLVEBLOK: a Fortran subprogram for solving almost block diagonal linear systems. Such matrices arise naturally in piecewise polynomial interpolation or approximation and in finite element methods for two-point boundary value problems. (See C. de Boor and R. Weiss, ACM TOMS 6 (1980) pp. 88-91.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a2, E3d, I1c **Usage:** CALL SLVBLK (*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A547.** DCSINT and DCSSMO: Fortran subroutines for discrete cubic spline interpolation and smoothing. (See C.S. Duris, ACM TOMS 6 (1980) pp. 92-103.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E1a **Usage:** CALL DCSINT(*10 parameters*) AND CALL DCSSMO(*10 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A548.** ASSCT: a Fortran subroutine for solving the square assignment problem. (See G. Carpaneto and P. Toth, ACM TOMS 6 (1980) pp. 104-111.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2b **Usage:** CALL ASSCT(N, A, C, T) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A549.** Fortran functions for evaluating Weierstrass's P-functions in the equiharmonic and lemniscatic cases. (See U. Eckhardt, ACM TOMS 6 (1980) pp. 112-120.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C15 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A550.** PROPS and SRFINT: Fortran subroutines for computing surface area, centroid, volume, weight, moments, and products of inertia of solid polyhedra. The routines operate directly on geometric definitions consisting of vertex coordinates and face lists, in a format similar to that employed in graphics programs (See A.M. Messner and G.Q. Taylor, ACM TOMS 6 (1980) pp. 121-130.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** P **Precision:** Single **Availability:** 855NOS (In source form only.)

**A551.** L1: a Fortran subroutine for solving an overdetermined system of linear equations in the L1 norm by using a dual simplex algorithm to the linear programming formulation of the given problem. (See N.N. Abdelmalek, ACM TOMS 6 (1980) pp. 228-230.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9a3 **Usage:** CALL L1(*26 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A552.** CL1: a Fortran subroutine for computing an L1 solution to a k-by-n system of linear algebraic equations subject to both linear equality and inequality constraints. The problem is solved using a modification of the simplex method. (See I. Barrodale and F.D.K. Roberts, ACM TOMS 6 (1980) pp. 231-235.) *Portable Fortran* *software in Collected Algorithms of the ACM.* **Classes:** D9a3, D9b3 **Usage:** CALL CL1(*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A553.** M3RK: a Fortran subroutine for solving initial value problems for nonlinear first-order systems of ordinary differential equations which originate from semi-discretization of parabolic partial differential equations. M3RK is based on stabilized, explicit three-step Runge-Kutta formulas of order one and two, and degree 2 through 12. (See J.G. Verwer, ACM TOMS 6 (1980) pp. 236-239.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1a1a, I2a1 **Usage:** CALL M3RK(*16 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A554.** BRENTM: a Fortran subroutine for solving a system of nonlinear equations in n variables by using a modification of Brent's method. (See J.J. More and M.Y. Cosnard, ACM TOMS 6 (1980) pp. 240-251.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Usage:** CALL BRENTM(*15 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A555.** FIXPT: a set of Fortran subroutines for computing fixed points or zeros of an n-dimensional vector function f(x). For the fixed point problem f is assumed to be a $C_2$ map of some ball into itself. For the zero finding problem, f is assumed to be a $C_2$ map such that for some r>0, xf(x)≥0 whenever ||x||=r. (See L.T. Watson and D. Fenner, ACM TOMS 6 (1980) pp. 252-259.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Usage:** CALL FIXPT(*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A556.** EXPINT: a Fortran subroutine for computing sequences of exponential integrals E(n+k,x), k=0, 1, ..., m-1 for n≥1, and x≥0. (See D.E. Amos, ACM TOMS 6 (1980) pp. 420-428.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C5 **Usage:** CALL EXPINT(*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A557.** PAGP: a partitioning algorithm for linear goal programming problems. (See J.L. Arthur and A. Ravindran, ACM TOMS 6 (1980) pp. 429.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2a1 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A558.** LOCATE: a Fortran subroutine for the one-dimensional multifacility location problem with rectilinear distance by a minimum-cut approach. (See T. Cheung, ACM TOMS 6 (1980) pp. 430-431.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2c5 **Usage:** CALL LOCATE(NVPT, NFPT, VWT, VFWT, POST) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A559.** HSQP: a subroutine for computing the stationary point of a quadratic function of n variables subject to linear constraints. (See J.T. Betts, ACM TOMS 6 (1980) pp. 432-436.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2e1, G2e2 **Usage:** CALL HSQP(*16 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A560.** JNF: a Fortran subroutine for computing the Jordan normal form of a complex square matrix. (See B. Kagstrom and A. Ruhe, ACM TOMS 6 (1980) pp. 437-443.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4c5 **Usage:** CALL JNF(*20 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A561.** Fortran subroutines for efficient table maintenance using heaps. (See D.K. Kahaner, ACM TOMS 6 (1980) pp. 444-449.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** N4 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A562.** A Fortran program for calculating the shortest path length from a specific node to all other nodes in a network. (See U. Pape, ACM TOMS 6 (1980) pp. 450-455.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2d1 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A563.** CL1: a set of subroutines for solving overdetermined systems of linear equations in the L1 sense, with or without linear constraints. (See R.H. Bartels and A.R. Conn, ACM TOMS 6 (1980) pp. 609-614.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9b3 **Usage:** CALL CL1(*16 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A564.** L1GNR: a Fortran subroutine for generating test problems for discrete linear $L_1$ approximation problems. (See K.L. Hoffman and D.R. Shier, ACM TOMS 6 (1980) pp. 615-617.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K3, K6d, L6b12, L8a2, L8c3 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A565.** PDETWO/PSETM/GEARB: a Fortran package for solving time-dependent coupled systems of nonlinear partial differential equations which are defined over a two-dimensional rectangular region. (See D.K. Melgaard and R.F. Sincovec, ACM TOMS 7 (1981) pp. 126-135.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2a1b, I2a2 **Usage:** CALL DRIVEP(*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A566.** Fortran subroutines for testing unconstrained optimization software in three areas: zeros of systems of N nonlinear functions in n variables, least squares minimization of M nonlinear functions in N variables, and unconstrained minimization of an objective function with N variables. (See J.J. More, B.S. Garbow, and K.E. Hillstrom, ACM TOMS 7 (1981) pp. 136-140.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F3, G4f **Precision:** Double **Availability:** 855NOS (In source form only.)

**A567.** NORMP: a Fortran subroutine for computing values of normalized Legendre polynomials of varying order and of fixed argument and degree. Also included is a package of six Fortran subroutines to facilitate the use of a special form of computer floating-point arithmetic called extended-range arithmetic. (See D.W. Lozier and J.M. Smith, ACM TOMS 7 (1981) pp. 141-146.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** A3d, C9, C3a2 **Usage:** CALL NORMP(*8 parameters*) **Availability:** 855NOS (In source form only.)

**A568.** PDS: a portable file directory system implemented in Fortran. (See D.R. Hanson, ACM TOPLAS vol. 3, pp. 162-167) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** Z **Precision:** Single **Availability:** 855NOS (In source form only.)

**A569.** COLSYS: Fortran subroutine for solving nonlinear multi-point boundary value problems for mixed order systems of ordinary differential equations. Based upon spline collocation at Gaussian points using a B-spline basis. Approximate solutions are computed on a sequence of automatically selected meshes until a user-specified set of tolerances is satisfied. (See U. Ascher, J. Christiansen, and R.D. Russell, ACM TOMS 7 (1981) pp. 223-229.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1b2 **Usage:** CALL COLSYS (*17 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A570.** LOPSI: a Fortran subprogram which determines approximations to right or left eigenvectors corresponding to the dominant set of eigenvalues of a real symmetric matrix using the method of simultaneous iteration. (See W.J. Stewart and A. Jennings, ACM TOMS 7 (1981) pp. 230-232.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4a7 **Usage:** CALL LOPSI (*28 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A571.** BESRAT, VKAPPA, SPHERR, CAPPA3: Fortran functions providing statistics for von Mises's and Fisher's distributions of directions (the ratio of modified Bessel functions of the first kind). (See G.W. Hill, ACM TOMS 7 (1981) pp. 233-238.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10b, L5a1v **Precision:** Single **Availability:** 855NOS (In source form only.)

**A572.** HELM3D: a Fortran subroutine for solving the Dirichlet problem for the Helmholtz equation on general bounded three-dimensional regions. Based upon second-order accurate finite differences; the resulting linear system of equations is reduced to a capacitance matrix equation that is solved approximately by a conjugate gradient method. (See D.P. O'Leary and O. Widlund, ACM TOMS 7 (1981) pp. 239-246.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1b **Usage:** CALL HELM3D (*24 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A573.** NL2SOL: an adaptive nonlinear least-squares algorithm. (See J.E. Dennis, D.M. Gay, and R.E. Welsch, ACM TOMS 7 (1981) pp. 367-383.) *Portable Fortran software in Collected Algorithms of the ACM.* Classes: K1b1a1 **Usage:** CALL NL2SOL (*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A574.** A package of Fortran subprograms for constructing a shape-preserving osculatory quadratic spline. The spline is a piecewise quadratic Bernstein polynomial with a continuous first derivative which interpolates given function and first derivative values, and preserves monotonicity and convexity in the data. (See D.F. Mcallister and J.A. Roulier, ACM TOMS 7 (1981) pp. 384-386.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E1a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A575.** MC21A: a subroutine for finding a row permutation for a zero-free diagonal. That is, given the pattern of nonzeros of a sparse matrix, this routine attempts to find a permutation of its rows that makes the matrix have no zeros on its diagonal. (See I.S. Duff, ACM TOMS 7 (1981) pp. 387-390.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2e **Usage:** CALL MC21A (*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A576.** MODGE and REFINE: Fortran subroutines for solving an n-by-n system of (possibly singular) linear algebraic equations. The algorithm consists of Gaussian elimination combined with a new pivoting strategy which is particularly well suited to problems where residuals can be made small by solving for fewer than n of the unknowns. (See I. Barrodale and G.F. Stuart, ACM TOMS 7 (1981) pp. 391-397.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a1, D9a1, D9a4 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A577.** RC, RF, RD, RJ: Fortran functions for symmetric incomplete elliptic integrals of the first, second, and third kinds. (See B.C. Carlson and E.M. Notis, ACM TOMS 7 (1981) pp. 398-403.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C14 **Precision:** Double **Availability:** 855NOS (In source form only.)

**A578.** Fortran subroutines for solving real linear equations in a paged virtual store. BLCFAC performs Gaussian elimination with partial pivoting on a real square matrix A with operations on blocks of consecutive columns grouped together to minimize paging on a machine with a paged virtual memory. BLCSOL solves systems based upon A or A-transpose with multiple right-hand sides using a similar strategy. (See J.J. Du Croz et al., ACM TOMS 7 (1981) pp. 537-189.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a1 **Usage:** CALL BLCFAC (*5 parameters*) AND CALL BLCSOL (*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A579.** CPSC: a Fortran subroutine for evaluating the leading coefficients in a power series expansion of an analytic function. (See B. Fornberg, ACM TOMS 7 (1981) pp. 542-547.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K4 **Usage:** CALL CPSC (F, Z, N, IC, R, RS, ER) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A580.** QRUP: a set of Fortran subroutines which compute a QR factorization of an m-by-n matrix A based on the Gramm-Schmidt orthogonalization process. They provide for an update of the factorization when rows or columns are added or deleted from A, or when A is modified by the addition of a rank one matrix. (See A. Buckley, ACM TOMS 7 (1981) pp. 548-549.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D7c **Precision:** Single **Availability:** 855NOS (In source form only.)

**A581.** HYBSVD, MGNSVD, and GRSVD: Fortran subprograms for computing the singular value decomposition of a general rectangular matrix. The algorithm is a modification of the Golub-Reinsch procedure that is more efficient when the matrix has more rows than columns. (see T. F. Chan, ACM TOMS 8 (1982) pp. 84-88.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D6 **Usage:** CALL HYBSVD (*18 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A582.** GPSKCA: Fortran subroutine for bandwidth or profile reduction of structurally symmetric sparse matrices. (See J.G. Lewis, ACM TOMS 8 (1982) pp. 190-194.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2e **Precision:** Single **Availability:** 855NOS (In source form only.)

**A583.** LSQR: Fortran subprogram for solving overdetermined or underdetermined sparse systems of linear equations, sparse least squares problems, and damped sparse least squares problems. (See C.C. Paige and M.A. Saunders, ACM TOMS 8 (1982) pp. 195-209.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9a1 **Usage:** CALL LSQR (*24 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A584.** CUBTRI: a Fortran subroutine for adaptive cubature over a triangle. (See D.P. Laurie, ACM TOMS 8 (1982) pp. 210-218.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** H2b2a1 **Usage:** CALL CUBTRI (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A585.** EXTRAP: A Fortran subroutine for sequence extrapolation (based upon the E-algorithm) and generalized interpolation by a linear combination of functions forming a Chebyshev system (based upon the Muhlbach-Neville-Aitken algorithm). Includes the Epsilon Algorithm of Wynn as a special case. (See C. Brezinski, ACM TOMS 8 (1982) pp. 290-301.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** A7, E1c **Usage:** CALL EXTRAP (*7 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A586.** ITPACK 2C: a set of seven Fortran subprograms (JCG, JSI, SOR, SSORCG, SSORSI, RSCG, and RSSI) for solving large sparse linear systems by adaptive accelerated iterative methods. (See D.R. Kincaid et al., ACM TOMS 8 (1982) pp. 302-322.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2b4 **Usage:** CALL <SUBROUTINE> (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A587.** LSEI and WNNLS: Fortran subprograms for solving least squares problems with linear equality and/or inequality constraints. (See R.J. Hanson and K.H. Haskell, ACM TOMS 8 (1982) pp. 323-333.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9b1 **Usage:** CALL LSEI (*13 parameters*) AND CALL WNNLS (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A588.** HANKEL: a Fortran subroutine for the fast evaluation of complex Hankel transforms of orders 0 and 1 using related and lagged convolutions. (See W.L. Anderson, ACM TOMS 8 (1982) pp. 369-370.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10f, J, J2 **Usage:** CALL HANKEL (*13 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A589.** SICEDR: a Fortran subroutine for improving the accuracy of computed real matrix eigenvalues and improving or computing the associated eigenvector. (See J.J. Dongarra, ACM TOMS 8 (1982) pp. 371-375.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a4, D4c, D4c1b3, D4c2c **Usage:** CALL SICEDR (*19 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A590.** DSUBSP and EXCHQZ: Fortran subroutines for computing deflating subspaces with specified spectrum. (See P. Van Dooren, ACM TOMS 8 (1982) pp. 376-382.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4c1b3 **Usage:** CALL DSUBSP (*10 parameters*) AND CALL EXCHQZ (*10 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A591.** A storage-efficient Fortran program for analysis of variance of balanced data, unbalanced data, and unbalanced data with missing cells. (See W.J. Hemmerle, ACM TOMS 8 (1982) pp. 383-401.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L7d **Precision:** Single **Availability:** 855NOS (In source form only.)

**A592.** RANGE: Given values of an unknown function f at n distinct points on an interval, an integer k, and a finite bound on the kth derivative of f, this routine determines the range of possible values of f(x) in the interval. (See P.W. Gaffney, ACM TOMS 9 (1983) pp. 98-116.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E1a **Usage:** CALL RANGE (*17 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A593.** CMMEXP, CMMIMP, and CMMSIX: Fortran subroutines for solving the Helmholtz equation on bounded nonrectangular planar regions with Dirichlet or Neumann boundary conditions. Solution is based upon the Fourier method extended to nonrectangular regions using the capacitance matrix method. (See W. Proskurowski, ACM

TOMS 9 (1983) pp. 117-124.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A594.** A Fortran program which performs an automatic roundoff error analysis of numerical algorithms. (See J.L. Larson, M.E. Pasternak, and J.A. Wisniewski, ACM TOMS 9 (1983) pp. 125-130.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** A3a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A595.** HC: a Fortran subprogram for finding one or more Hamiltonian circuits in a directed graph. (See S. Martello, ACM TOMS 9 (1983) pp. 131-138.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** P **Usage:** CALL HC (*9 parameters*), PC, AC, VR, VC, P, SUBR, RBUS, TOR) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A596.** PITCON: a Fortran subprogram for continuation. Computes a set of solutions to nonlinear system of equations containing a parameter. On such a solution curve the exact location of target points where a given variable has a specified value can be located. Limit points are also identified. Based upon a local parameterization which uses curvature estimates to control the choice of parameter value. (See W.C. Rheinboldt and J.V. Burkardt, ACM TOMS 9 (1983) pp. 236-241.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Usage:** CALL PITCON (*25 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A597.** RIBESL: a Fortran subprogram for calculating sequences of modified Bessel functions of the first kind (real argument and real order). Optionally, the result may be exponentially scaled. (See W.J. Cody, ACM TOMS 9 (1983) pp. 242-245.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10b3 **Usage:** CALL RIBESL (*6 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A598.** SQUINT: a Fortran subprogram for computing solvents of the matrix equation $AX^2 + BX + C = 0$. (See G.W. Davis, ACM TOMS 9 (1983) pp. 246-254.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D8 **Usage:** CALL SQUINT (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A599.** SEXPO, SGAMMA, SNORM, KPOISS, and SUNIF: Fortran functions which sample from the standard exponential, standard gamma, standard normal, Poisson, and uniform (0,1) distributions. (See J.H. Ahrens, K.D. Kohrt, and U. Dieter, ACM TOMS 9 (1983) pp. 255-257.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L6a7, L6a16, L6a21, L6a5, L6a14 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A5A6.** Perform Gaussian elimination with partial pivoting on a general system of linear equations. Illustrates use of algorithms A5GE and A6BS. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 62-63.) *Portable Fortran software in NASHLIB library.* **Classes:** D2a1 **Usage:** CALL A5GE(*6 parameters*), CALL A6BS(*4 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A600.** QUINAT, QUINEQ, and QUIND: Fortran subprograms for quintic natural spline interpolation. This is a translation of ACM Algorithm 507. (See J.G. Herriot and C.H. Reinsch, ACM TOMS 9 (1983) pp. 258-259.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E1a **Precision:** Single **Availability:** 855NOS (In source form only.)

**A601.** A set of eight Fortran subroutines for multiplying and adding pairs of sparse matrices in special cases, that is, in which one of the pair is full and/or a vector or an elementary matrix. Also provided are routines for transposing a sparse matrix and for multiplying two sparse matrices. (See J.M. McNamee, ACM TOMS 9 (1983) pp. 344-345.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D1b5, D1b6 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A602.** HURRY: a Fortran subprogram for accelerating the convergence of alternating and monotone sequences and series (based on Levin's u transform). The routine estimates truncation and roundoff errors to make a near-optimal stopping decision and provide a good estimate of the accuracy. (See T. Fessler, W.F. Ford, and D.A. Smith, ACM TOMS 9 (1983) pp. 355-357.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** A7 **Usage:** CALL HURRY (*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A603.** COLROW and ARCECO: Fortran subroutines for solving certain almost block diagonal linear systems by modified alternate row and column elimination. Such systems arise when solving boundary-value problems for ordinary differential equations. COLROW is designed for systems whose blocks all have the same dimension; ARCECO is designed for systems whose blocks may have different dimensions. (See J.C. Diaz, G. Fairweather, and P. Keast, ACM TOMS 9 (1983) pp. 376-380.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a2, E3d, I1c **Usage:** CALL COLROW (*14 parameters*) AND CALL ARCECO (*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A604.** EXTREM: a Fortran subprogram for the calculating extremal polynomials. If L is a linear functional on polynomials of degree n or less, then p is extremal if it is a polynomial of (Chebyshev) norm one at which L takes on its norm. (See F.W. Sauer, ACM TOMS 9 (1983) pp. 381-383.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K2, D2e, C3b **Usage:** CALL EXTREM (*15 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A605.** PBASIC: a Fortran program which checks a BASIC program for adherence to the American National Standard Minimal Standard for BASIC. (See T.R. Hopkins, ACM TOMS 9 (1983) pp. 391-394.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** S2 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A606.** NITPACK: Fortran subprograms for implementing decision trees. NITREE converts a paper tree into a format that may be transmitted to a user at a computer terminal. NIT performs the transmission. (see P.W. Gaffney et al., ACM TOMS 9 (1983) pp. 418-426.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** R4, N4 **Usage:** CALL NITREE (*2 parameters*) AND CALL NIT (*2 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A607.** TES (Text Exchange System): transportable Fortran programs for management and exchange of programs and other text. TES defines a format for information storage and includes two programs, the first creates, reads, and maintains TES files, and the second permits TES to be distributed on a tape in TES format. (See W.V. Snyder and R.J. Hanson, ACM TOMS 9 (1983) pp. 427-440.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** N4, Z **Precision:** Single **Availability:** 855NOS (In source form only.)

**A608.** HGW: A Fortran subprogram to compute an approximate solution to the extended Koopmans-Beckmann quadratic assignment problem. (See D.H. West, ACM TOMS 9 (1983) pp. 461-466.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2b **Usage:** CALL HGW (*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A609.** BSKIN: a Fortran subprogram which computes the Bickley functions Ki(n,x), repeated integrals of the K0 Bessel function, for non-negative integers n and reals x. A double precision version is included. (See D.E. Amos, ACM TOMS 9 (1983) pp. 480-493.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10f **Usage:** CALL BSKIN (X, N, KODE, M, Y, IERR) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A610.** PSIFN: a Fortran subprogram for calculating derivatives of the psi function. A double precision version is included. (See D.E. Amos, ACM TOMS 9 (1983) pp. 494-502.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C7c **Precision:** Single **Availability:** 855NOS (In source form only.)

**A611.** SMSNO, SUMSL, and HUMSL: Fortran subprograms for solving general unconstrained minimization problems using a model/trust-region approach. The routines offer the option of providing function and gradient, or function, gradient, and Hessian values by reverse communication. (See D.M. Gay, ACM TOMS 9 (1983) pp. 503-524.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G1b1b, G1b1c **Usage:** CALL SUMSL (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A612.** TRIEX: a Fortran subroutine for integration over a triangle. Uses an adaptive subdivisional strategy with global acceptance criteria and incorporates the epsilon algorithm to speed convergence. (see E. de Doncker and I. Robinson, ACM TOMS 10 (1984) pp. 17-22.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** H2b2a1 **Usage:** CALL TRIEX: (*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only. Both single and double precision versions are available.)

**A613.** MSTPAC: a Fortran subroutine for calculating the minimum spanning tree for moderate integer weights in a connected undirected graph represented in a forward star data structure. (See R.E. Haymond, J.P. Jarvis, and D.R. Shier, ACM TOMS 10 (1984) pp. 108-111.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2d2 **Usage:** CALL MSTPAC (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A614.** INTHP: a Fortran subroutine for automatic numerical integration in Hp. The functions may have singularities at one or both endpoints of an interval. Each of finite, semi-infinite, and infinite intervals are admitted. (See K. Sikorski, F. Stenger, and J. Schwing, ACM TOMS 10 (1984) pp. 152-160.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** H2a1a1, H2a3a1, H2a4a1 **Usage:** CALL INTHP (*9 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A615.** KBEST: a Fortran subprogram to determine the best subset of parameters to fit a linear regression under a least absolute value criterion. It utilizes the simplex method of linear programming within a branch-and-bound algorithm to solve the best subset problem. (See R.D. Armstrong, P.O. Beck, and M.T. Kung, ACM TOMS 10 (1984) pp. 202-206.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L8c3 **Usage:** CALL KBEST (*14 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A616.** DAFNE: a Fortran subprogram for solving nonlinear systems based on the numerical solution of a Cauchy problem for a system of ordinary differential equations inspired by classical mechanics. (See F. Aluffi-Pentini, V. Parisi, and F. Zirilli, ACM TOMS 10 (1984) pp. 317-324.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Usage:** R = HLQEST(X, N, LB, RB, Q) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A617.** DAFNE: a Fortran subprogram for solving nonlinear systems based on the numerical solution of a Cauchy problem for a system of ordinary differential equations inspired by classical mechanics. (See F. Aluffi-Pentini, V. Parisi, and F. Zirilli, ACM TOMS 10 (1984) pp. 317-324) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Usage:** CALL DAFNE (*23 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A618.** DSM and FDJS: Fortran subroutines for estimating sparse Jacobian matrices. (See T.J. Coleman, B.S. Garbow, and J.J. More, ACM TOMS 10 (1984) pp. 346-347.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F3 **Usage:** CALL DSM (*13 parameters*) AND CALL FDJS (*10 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A619.** DLAINV: a Fortran subroutine for the automatic numerical inversion of the Laplace transform using the Durbin formula in combination with the epsilon algorithm. (See R. Piessens and R. Huysmans, ACM TOMS 10 (1984) pp. 348-353.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** J3 **Precision:** Double **Availability:** 855NOS (In source form only.)

**A620.** Data files containing references and keywords for the Collected Algorithms from ACM. (See J.R. Rice and R.J. Hanson, ACM TOMS 10 (1984) pp. 359-360.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** Z **Availability:** 855NOS (In source form only.)

**A621.** BDMG: a Fortran subprogram with low storage requirements for two-dimensional nonlinear parabolic differential equations on rectangular spatial domains with mixed linear boundary conditions. (See B.P. Sommeijer and P.J. van der Houven, ACM TOMS 10 (1984) pp. 378-396.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2a1b **Usage:** CALL BDMG (*16 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A622.** A simple macroprocessor for use in manipulating Fortran code as well as for general text processing. (See J.R. Rice, C. Ribbens, and W.A. Ward, ACM TOMS 10 (1984) pp. 410-416.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** Z, S1 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A623.** A set of Fortran programs for constructing an interpolant with one continuous derivative from data values associated with arbitrarily distributed nodes on the surface of a sphere. (See R.J. Renka, ACM TOMS 10 (1984) pp. 417-436 and 437-439.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E2b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A624.** A set of Fortran programs for triangulation and interpolation at arbitrarily distributed points in the plane. (See R.J. Renka, ACM TOMS 10 (1984) pp. 440-442.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E2b, P **Precision:** Single **Availability:** 855NOS (In source form only.)

**A625.** A Fortran subprogram which relates a general two-dimensional domain to a rectangular grid laid over it. (See J.R. Rice, ACM TOMS 10 (1984) pp. 443-452 and 453-462.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b4, P **Precision:** Single **Availability:** 855NOS (In source form only.)

**A626.** TRICP: a Fortran subprogram for computing contours of a function defined by a set of irregularly distributed data points in the plane. (See A. Preusser, ACM TOMS 10 (1984) pp. 178-189.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** Q **Precision:** Single **Availability:** 855NOS (In source form only.)

**A627.** VE1: a Fortran subroutine for solving Volterra integral equations. (See J.M. Bownds and L. Applebaum, ACM TOMS 11 (1985) pp. 58-65.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I3 **Usage:** CALL VE1(*14 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A628.** GROEB: a Fortran program for constructing canonical (or Groebner) bases of polynomial ideals. (See F. Winkler et al., ACM TOMS 11 (1985) pp. 66-78.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C3b **Precision:** Single **Availability:** 855NOS (In source form only.)

**A629.** LAPLAC: a Fortran subroutine for the interior Dirichlet problem for Laplace's equation on a general three dimensional domain. Based on integral equation techniques. (See K.E. Atkinson, ACM TOMS 11 (1985) pp. 85-96.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1b **Precision:** Double **Availability:** 855NOS (In source form only.)

**A630.** BBVSCG: a variable storage Fortran subprogram for function minimization. (See A. Buckley and A. Lenir, ACM TOMS 11 (1985) pp. 103-119.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G1b1a, G1b1b **Usage:** CALL BBVSCG (*12 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only. Both single and double precision versions are included.)

**A631.** ZERO1 and ZERO2: Fortran subroutines for finding a bracketed zero by Larkin's method of rational interpolation. ZERO2 requires derivative; ZERO1 does not. (See V. Nortin, ACM TOMS 11 (1985) pp. 120-134.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F1b **Usage:** CALL ZERO1 (*8 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A632.** MKP: a Fortran subroutine for the 0-1 multiple knapsack problem. (See S. Martello and P. Toth, ACM TOMS 11 (1985) pp. 135-140.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2c3 **Usage:** CALL MKP (*8 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A633.** LDA: a Fortran subroutine for linear dependency analysis of multivariate data. (See R.C. Ward, G.J. Davis, and V.E. Kane, ACM TOMS 11 (1985) pp. 170-182.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L8f **Usage:** CALL LDA (*16 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A634.** CONST and EVAL: Fortran subroutines for fitting multinomials in a least-squares sense. (See R.H. Bartels and J.J. Jezioranski, ACM TOMS 11 (1985) pp. 218-228.) *Portable Fortran software in Collected Algorithms of* *the ACM.* **Classes:** K1a1b **Usage:** CALL CONSTR (*17 parameters*) AND CALL EVAL (*13 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A635.** A set of Fortran subroutines for computing the Chebyshev solution of systems of complex linear equations with constraints on the unknowns. Both general linear inequality constraints and simple bound constraints are allowed. (See R.L. Streit, ACM TOMS 11 (1985) pp. 242-249.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9b2 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A636.** DSSM and FDHS: Fortran subroutines for estimating sparse Hessian matrices. (See T.F. Coleman, B.S. Garbow, and J.J. More, ACM TOMS 11 (1985) pp. 363-377 and 378.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G4f **Usage:** CALL DSSM (*14 parameters*) AND CALL FDHS (*13 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**A637.** GENCOL: a Fortran subprogram for linear second-order elliptic problems with general linear boundary conditions on non-rectangular two-dimensional domains. Solves the problem using collocation with bicubic Hermite polynomials. (See E.N. Houstis, W.F. Mitchell, and J.R. Rice, ACM TOMS 11 (1985) pp. 379-412 and 413-415.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a3 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A638.** INTCOL and HERMCOL: Fortran subprograms for linear second-order elliptic problems on rectangular two-dimensional domains. HERMCOL allows general linear boundary conditions while INTCOL required uncoupled boundary conditions. Problem are solved using collocation with bicubic Hermite polynomials. (See E.N. Houstis, W.F. Mitchell, and J.R. Rice, ACM TOMS 11 (1985) pp. 379-412 and 416-418.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a3 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A639.** OSCINT: a Fortran subprogram for the automatic integration of some infinitely oscillating tails. That is, the evaluation of the integral from a to infinity of h(x)j(x), where h(x) is ultimately positive, and j(x) is either a circular function (e.g., cosine) or a first-kind Bessel function of fractional order. (See J. Lyness and G. Hines, ACM TOMS 12 (1986) pp. 24-25.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** H2a3a1 **Usage:** CALL OSCINT(*17 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A640.** SFRMG: A Fortran subprogram which takes real matrices A (n-by-n), B (n-by-m), and C (l-by-n) and forms the complex frequency response matrix CEB, where E is the inverse of (FREQ×I − A), I is the n-by-n identity matrix and FREQ is a complex scalar parameter taking values along the imaginary axis for continuous-time systems and on the unit circle for discrete-time systems. (See A.J. Laub, ACM TOMS 12 (1986) pp. 26-33.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G3, L10c **Precision:** Single **Availability:** 855NOS (In source form only.)

**A641.** EXSOLG: a Fortran subprogram for the exact solution of general (m-by-n) systems of linear equations with integer coefficients. If the system is singular a least square solution is computed. (See J. Springer, ACM TOMS 12 (1986) p. 149.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a1, D9a1 **Usage:** CALL EXSOLG(*25 parameters*) **Availability:** 855NOS (In source form only.)

**A642.** CUBGCV: a Fortran subprogram for the fast O(n) computation of a cubic smoothing spline fitted to n noisy data points, with the degree of smoothing chosen to minimize the expected mean square error at the data points when the variance of the error in the data is known, or the generalized cross validation when it is unknown. The data may be unequally spaced and nonuniformly weighted. Bayesian point error estimates are also calculated. (See M.F. Hutchinson, ACM TOMS 12 (1986) pp. 150-153.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** K5, L8g **Usage:** CALL CUBGCV (*12 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only. Both single and double precision versions are included.)

**A643.** FEXACT: a Fortran subprogram for Fisher's exact test on unordered r-by-c contingency tables. (See C.R. Mehta and N.R. Patel, ACM TOMS 12 (1986) pp. 154-161.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** L9b **Usage:** CALL FEXACT(*5 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**A644.** A portable Fortran package for Bessel functions of a complex argument and nonnegative order. In particular, the Bessel functions $H_1$, $H_2$, I, J, K, and Y, as well as the Airy functions Ai, Bi, and their derivatives are provided in both single and double precision. Exponential scaling and sequence generation are optional. (See D.E. Amos, ACM TOMS 12 (1986) pp. 265-273.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C10a4, C10b4, C10d **Precision:** Single **Availability:** 855NOS (In source form only.)

**A645.** Fortran subroutines for testing programs that compute the generalized inverse of a matrix. (See J.C. Nash and R.L.C. Wang, ACM TOMS 12 (1986) pp. 274-277.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D9c **Precision:** Single **Availability:** 855NOS (In source form only.)

**A646.** PDFIND: a Fortran subprogram for finding a positive definite linear combination of two real symmetric matrices. The algorithm is independent of the data structure used to store the matrices. The combination may be used to solve the generalized eigenproblem $Ax = \lambda Bx$ in case A and B are large and sparse, but neither is positive definite. (See C.R. Crawford, ACM TOMS 12 (1986) pp. 278-282.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D4b1, D4c1c **Usage:** CALL PDFIND(CHLSKY, S, INFO) **Precision:** Double

**Availability:** 855NOS (In source form only.)

**A647.** Fortran subprograms for the generation of sequences of quasirandom vectors with low discrepancy. Such sequences may be used to reduce error bounds for multidimensional integration and global optimization. (See B.L. Fox, ACM TOMS 12 (1986) pp. 362-376.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** H2c, L6b21 **Precision:** Single **Availability:** 855NOS (In source form only.)

**A648.** NSDTST and STDTST: Fortran routines for assessing the performance of initial value solvers for stiff or nonstiff systems. (See W. H. Enright and J. D. Pryce, ACM TOMS 13 (1987) pp. 28-34.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1c **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A649.** FOURCO: A Fortran package for computing the trigonometric Fourier coefficients of a smooth function using Lyness's algorithm. (See G. Giunta and A. Murli, ACM TOMS 13 (1987) pp. 97-107.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** J1a3 **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A650.** Assembly language programs for efficient square root implementation on the Motorola 68000. (See K. C. Johnson, ACM TOMS 13 (1987) pp. 138-151.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C2 **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A651.** HFFT: Fortran routines for solving the Helmholtz equation on bounded two- or three-dimensional rectangular domains. (See R. F. Boisvert, ACM TOMS 13 (1987) pp. 235-249.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I2b1a1a **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A652.** HOMPACK: Fortran routines for globally convergent homotopy algorithms, for finding zeros or fixed points of nonlinear systems of equations. (See L. T. Watson, S. C. Billups, and A. P. Morgan, ACM TOMS 13 (1987) pp. 281-310.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** F2 **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A653.** PC-BLAS: Assembly language version of the Basic Linear Algebra Subprograms for Fortran usage with the INTEL 8087/80287 numeric data processor. (See R. J. Hanson and F. T. Krogh, ACM TOMS 13 (1987) pp. 311-317.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D1a **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A654.** GRATIO and GAMINV: Fortran routines for computing the incomplete gamma function ratios and their inverse. (See A. R. DiDonato and A. H. Morris, Jr., ACM TOMS 13 (1987) pp. 318-319.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** C7e **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A655.** IQPACK: Fortran routines for the stable evaluation of the weights and nodes of interpolatory and Gaussian quadratures with prescribed simple or multiple knots. (See S. Elhay and J. Kautsky, ACM TOMS 13 (1987) pp. 399-415.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** H2c **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A656.** Fortran model implementation and test programs for Level 2 BLAS. (See J. J. Dongarra, J. du Croz, S. Hammarling, and R. J. Hanson, ACM TOMS 14 (1988) pp. 18-32.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D1 **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A657.** CON3D: Fortran software for plotting contour surfaces of a function of three variables. (See G. Sewell, ACM TOMS 14 (1988) pp. 42-44.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** Q **Usage:** CALL CON3D(*18 parameters*) **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A658.** ODESSA: A Fortran ordinary differential equation solver (a modification of LSODE) with explicit simultaneous sensitivity analysis. (See J. R. Leis and M. A. Kramer, ACM TOMS 14 (1988) pp. 61-67.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** I1a1b, I1a2 **Usage:** CALL ODESSA(*19 parameters*) **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A659.** A Fortran implementation of Sobol's quasirandom sequence generator for multivariate quadrature and optimization. (See P. Bratley and B. L. Fox, ACM TOMS 14 (1988) pp. 88-100.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** G2i, H2c **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A660.** QSHEP2D: Fortran routines implementing the quadratic Shepard method for bivariate interpolation of scattered data. (See R. J. Renka, ACM TOMS 14 (1988) pp. 149-150.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E2b **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A661.** QSHEP3D: Fortran routines implementing the quadratic Shepard method for trivariate interpolation of scattered data. (See R. J. Renka, ACM TOMS 14 (1988) pp. 151-152.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** E2b **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A662.** MODUL1 and MODUL2: Fortran routines for the numerical inversion of the Laplace transform based on Weeks' method. (See B. S. Garbow, G. Giunta, and J. N. Lyness, ACM TOMS 14 (1988) pp. 171-176.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** J3 **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A663.** CWI BLAS: Basic Linear Algebra Subprograms in Fortran 200 for the Cyber 205. (See M. Louter-Nool, ACM TOMS 14 (1988) pp. 177-195.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D1a **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A664.** GBSOL: A Fortran implementation of a Gauss algorithm to solve systems with large banded matrices using random-access disk storage. (See G. Schrauf, ACM TOMS 14 (1988) pp. 257-260.) *Portable Fortran software in Collected Algorithms of the ACM.* **Classes:** D2a2 **Usage:** CALL GBSOL(*14 parameters*) **Precision:** Single **Availability:** 855NOS (Source not available at time of publication.)

**A7A8.** Use Cholesky algorithm to solve a symmetric system of linear equations. Illustrates use of algorithms A7CH and A8CS. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, pages 74-75.) *Portable Fortran software in NASHLIB library.* **Classes:** D2b1b **Usage:** CALL A7CH(*4 parameters*), CALL A8CS(*4 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**A9.** Use modification of Gauss Jordan algorithm to perform the Bauer Reinsch inversion of positive definite symmetric matrix. Illustrates use of algorithm A9GJ. (See J.C. Nash, Compact Numerical Methods for Computers: Linear Algebra and Function Minimization, John Wiley, New York, 1979, page 84.) *Portable Fortran software in NASHLIB library.* **Classes:** D4a2 **Usage:** CALL A9GJ(A, N2, N, INDEF, X) **Precision:** Single **Availability:** 855NOS, 840NOS. *On 855NOS:* In source form only.

**ABALD.** Analyze a balanced complete experimental design for a fixed, random, or mixed model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7d1 **Usage:** CALL ABALD (*16 parameters*) **Precision:** Single (Double: DABALD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ABIBD.** Analyze a balanced incomplete block design or a balanced lattice design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7d2 **Usage:** CALL ABIBD (*16 parameters*) **Precision:** Single (Double: DABIBD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ABIBN.** Analysis of balanced incomplete block and balanced lattice designs. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d2 **Usage:** CALL ABIBN (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACBCB.** Add two complex band matrices, both in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b5 **Usage:** CALL ACBCB (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACECLUS.** Obtains approximate estimates of the pooled within-cluster covariance matrix when the clusters can be assumed multivariate normal with equal covariance matrices. Neither cluster membership nor the number of clusters need to be known. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L14d **Precision:** Single **Availability:** CAMVAX.

**ACF.** Compute the sample autocorrelation function of a stationary time series. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2a1 **Usage:** CALL ACF (*10 parameters*) **Precision:** Single (Double: DACF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACF.** Computes and graphs the autocorrelations of a time series, and optionally saves results. *Command in MINITAB proprietary interactive system.* **Classes:** L10a2a1 **Usage:** ACF [with up to K lags] for series in C [put results in C] **Precision:** Single **Availability:** 855NOS.

**ACF.** Compute and print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACF (Y, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACFD.** Compute and print a two-part auto- and partial correlation analysis of a sequence of differenced series, select the order of an autoregressive process which models each series, and estimate the parameters of these models. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACFD (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACFF.** Compute and print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACFF (YFFT, N, LYFFT, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACFFS.** Compute and optionally print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use FFT for computations; use user-supplied control values; return autocovariance function, and order and parameter estimates of selected autoregressive model. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACFFS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACFM.** Compute and print a two-part auto- and partial correlation analysis of a series with missing observations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACFM (Y, YMISS, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACFMS.** Compute and optionally print a two-part auto- and partial correlation analysis of a series with missing observations; use user-supplied control values; return autocovariance function. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACFMS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACFS.** Compute and optionally print a two-part auto- and partial correlation analysis of a series, select the order of an autoregressive process which models the series, and estimate the parameters of this model; use user-supplied control values; return autocovariance function, and order and parameter estimates of selected autoregressive model. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2a, L10a2c1 **Usage:** CALL ACFS (*9 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ACHAR.** Return the character whose ASCII value is the input integer argument. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N3 **Usage:** H = ACHAR(I) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACOS.** arccos(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ACOS(X) **Precision:** Single (Double: DACOS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACOSH.** arccosh(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** R = ACOSH(X) **Precision:** Single (Double: DACOSH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ACOSH.** arccosh(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ACOSH(X) **Precision:** Single (Double: DACOSH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACOSH.** arccosh(x). *Proprietary Fortran function in PORT.* **Classes:** C4c **Usage:** R = ACOSH(X) **Precision:** Single (Double: DACOSH) **Availability:** 855NOS, 205.

**ACRDAN.** Analysis of one-way classification design data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a1 **Usage:** CALL ACRDAN (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACTBL.** Produce population and cohort life tables. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L15 **Usage:** CALL ACTBL (*10 parameters*) **Precision:** Single (Double: DACTBL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ACTRST.** Contrast estimates and sums of squares. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a1 **Usage:** CALL ACTRST (T, NR, N, ID, P, IP, Q, SQ) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ADAPT.** Computes the definite integral of a user specified function over a hyper-rectangular region in dimension 2 through 20. User specifies tolerance. A restarting feature is useful for continuing a computation without wasting previous function values. *Portable Fortran subroutine in CMLIB (ADAPT sublibrary).* **Classes:** H2b1a1 **Usage:** CALL ADAPT(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ADNODE.** Adds a node to a triangulation of a set of points in the plane producing a new triangulation. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL ADNODE (KK, X, Y, IADJ, IEND, IER) **Also see:** TRMESH **Precision:** Single **Availability:** 855NOS.

**AFACN.** Full factorial plan analysis. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d1 **Usage:** CALL AFACN (IOPT, NF, NL, Y, SS, NDF, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AFACT.** Full factorial plan analysis – easy to use version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d1 **Usage:** CALL AFACT (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AGBACP.** Analysis of balanced complete experimental design structure data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d1 **Usage:** CALL AGBACP (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AGGREGATE.** Provides summary statistics (sum, mean, standard deviation, percent of observations with specified range) for the values of one variable across groups of sequential observations. Option: missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1c1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**AGLMOD.** General linear model analysis. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d3 **Usage:** CALL AGLMOD (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AGVACL.** One or two-sided interval estimate of a variance component. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a1 **Usage:** CALL AGVACL(V, FDF, S, CF, IOP, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AGXPM.** Expected mean squares for balanced complete design models. *Proprietary Fortran subroutine in IMSL.*

**Classes:** L7d1 **Usage:** CALL AGXPM (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AI.** Airy function Ai(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10d **Usage:** R = AI(X) **Precision:** Single (Double: DAI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**AI.** Airy function Ai(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = AI(X) **Precision:** Single (Double: DAI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AID.** Forms a tree of clusters by splitting cases on values of individual variables to minimize the sum of the squared deviations from the cluster means. The tree can be used to predict the value of a different variable for a new case. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a4 **Usage:** CALL AID(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**AID.** Derivative of the Airy function Ai(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = AID(X) **Precision:** Single (Double: DAID) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AIDE.** Exponentially scaled derivative of the Airy function exp(x)dAi(x)/ dx. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = AIDE(X) **Precision:** Single (Double: DAIDE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AIE.** Exponentially scaled Airy function exp(x)Ai(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10d **Usage:** R = AIE(X) **Precision:** Single (Double: DAIE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**AIE.** Exponentially scaled Airy function exp(x)Ai(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = AIE(X) **Precision:** Single (Double: DAIE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AIME.** Compute and print a five-part least squares analysis of the parameter estimates of an ARIMA model; return parameter estimates and residuals. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2d2 **Usage:** CALL AIME (*8 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AIMEC.** Compute and optionally print a five-part least squares analysis of the parameter estimates of an ARIMA model using user-supplied control values; return parameter estimates and residuals. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2d2 **Usage:** CALL AIMEC (*17 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AIMES.** Compute and optionally print a five-part least squares analysis of the parameter estimates of an ARIMA model using user-supplied control values; return parameter estimates, residuals, number of parameters estimated, residual standard deviation, predicted values, standard deviations of the predicted values and variance-covariance matrix of the estimated parameters. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2d2 **Usage:** CALL AIMES (*25 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AIMF.** Compute and print the minimum mean square error forecasts obtained using an ARIMA model. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2d3 **Usage:** CALL AIMF (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AIMFS.** Compute and optionally print the minimum mean square error forecasts obtained using an ARIMA model; return forecasts and their standard errors. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a2d3 **Usage:** CALL AIMFS (*14 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AINT.** Integer part of x. Input and output are real. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C1 **Usage:** R = AINT(X) **Precision:** Single (Double: DINT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKEI0.** Kelvin function $kei_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = AKEI0(X) **Precision:** Single (Double: DKEI0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKEI1.** Kelvin function $kei_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = AKEI1(X) **Precision:** Single (Double: DKEI1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKEIP0.** Derivative of the Kelvin function $kei_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = AKEIP0(X) **Precision:** Single (Double: DKEIP0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKER0.** Kelvin function $ker_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = AKER0(X) **Precision:** Single (Double: DKER0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKER1.** Kelvin function $ker_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = AKER1(X) **Precision:** Single (Double: DKER1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKERP0.** Derivative of the Kelvin function $ker_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = AKERP0(X) **Precision:** Single (Double: DKERP0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKS1DF.** Evaluate the distribution function of the one-sided Kolmogorov-Smirnov goodness of fit D+ or D- test statistic based on continuous data for one sample. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1k **Usage:** R = AKS1DF(NOBS, D) **Precision:** Single (Double: DKS1DF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AKS2DF.** Evaluate the distribution function of the Kolmogorov-Smirnov goodness of fit D test statistic based on continuous data for two samples. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1k **Usage:** R = AKS2DF(NOBSX, NOBSY, D) **Precision:** Single (Double: DKS2DF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALATN.** Analyze a Latin square design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7c **Usage:** CALL ALATN (*9 parameters*) **Precision:** Single (Double: DALATN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALBETA.** ln(Beta(a,b)), for positive real a,b. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7b **Usage:** R = ALBETA(A, B) **Precision:** Single (Double: DLBETA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALBETA.** Logarithm of the complete beta function for positive arguments. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7b **Usage:** R = ALBETA(A, B) **Precision:** Single (Double: DLBETA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALGAMA.** $\ln(|\Gamma(x)|)$. *Proprietary Fortran function in IMSL.* **Classes:** C7a **Usage:** R = ALGAMA(Y) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALGAMS.** $\ln(|\Gamma(x)|)$ and $\mathrm{sign}(\Gamma(x))$. *Portable Fortran subroutine in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** CALL ALGAMS (X, G, S) **Precision:** Single (Double: DLGAMS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALGAMS.** $\ln(|\Gamma(x)|)$ and $\mathrm{sign}(\Gamma(x))$. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** CALL ALGAMS (X, ALGM, S) **Precision:** Single (Double: DLGAMS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALI.** Logarithmic integral, integral from 0 to x of 1/ln(t). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C5 **Usage:** R = ALI(X) **Precision:** Single (Double: DLI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALI.** Logarithmic integral, integral from 0 to x of 1/ln(t). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = ALI(X) **Precision:** Single (Double: DLI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALLAN STANDARD DEVIATION PLOT.** Generate an Allan standard deviation plot with horizontal axis the subsample size and vertical axis the computed Allan standard deviation for the subsample size. *Command(s) in DATAPLOT interactive system.* **Classes:** L10a3 **Usage:** ALLAN STANDARD DEVIATION PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**ALLAN VARIANCE PLOT.** Generate an Allan variance plot with horizontal axis the subsample size and vertical axis the computed Allan variance for the subsample size. *Command(s) in DATAPLOT interactive system.* **Classes:** L10a3 **Usage:** ALLAN VARIANCE PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**ALNGAM.** $\ln(|\Gamma(x)|)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** R = ALNGAM(X) **Precision:** Single (Double: DLNGAM) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALNGAM.** $\ln(|\Gamma(x)|)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** R = ALNGAM(X) **Precision:** Single (Double: DLNGAM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALNREL.** ln(x+1). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** R = ALNREL(X) **Precision:** Single (Double: DLNREL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALNREL.** ln(1+x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** R = ALNREL(X) **Precision:** Single (Double: DLNREL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALOG.** ln(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** R = ALOG(X) **Precision:** Single (Double: DLOG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALOG.** ln(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** R = ALOG(X) **Precision:** Single (Double: DLOG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALOG10.** log(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** R = ALOG10(X) **Precision:** Single (Double: DLOG10) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ALOG10.** log(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** R = ALOG10(X) **Precision:** Single (Double: DLOG10) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ALSQAN.** Analysis of Latin square design data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7c **Usage:** CALL ALSQAN (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AMACH.** Retrieve single-precision machine constants. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** R1 **Usage:** R = AMACH(N) **Precision:** Single (Double: DMACH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AMEANS.** Preparation of a set of unbalanced data for analysis by the method of unweighted means. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d3 **Usage:** CALL AMEANS (Y, N, K, YM, HN, SS, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AMILLR.** Evaluate Mill's ratio (the ratio of the ordinate to the upper tail area of the standardized normal distribution). *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** L5a1n **Usage:** R = AMILLR(X) **Precision:** Single (Double: DMILLR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANCOV1.** Covariance analysis for one-way classification design data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a1 **Usage:** CALL ANCOV1 (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANEST.** Analyze a completely nested random model with possibly unequal numbers in the subgroups. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7d1, L7d3 **Usage:** CALL ANEST (*12 parameters*) **Precision:** Single (Double: DANEST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANESTE.** Analysis of completely nested design data with equal numbers in the subclasses. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d1 **Usage:** CALL ANESTE(NF, NL, Y, S, NDR, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANESTU.** Analysis of completely nested design data with unequal number in the subclasses. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d3 **Usage:** CALL ANESTU(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANORDF.** Normal (Gaussian) distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1n, C8a **Usage:** R = ANORDF(X) **Precision:** Single (Double: DNORDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANORIN.** Inverse of the normal (Gaussian) distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a2n, C8a **Usage:** R = ANORIN(P) **Precision:** Single (Double: DNORIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ANOVA.** Performs analysis of variance on balanced data with up to 5 factors. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviation, residual degrees of freedom, F-tests of significance, as well as replication standard deviation, replication degrees of freedom, and lack of fit F statistic if there is replication. *Command(s) in DATAPLOT interactive system.* **Classes:** L7a1, L7d1 **Usage:** ANOVA <RESPONSE VARIABLE> <LIST OF ONE TO FIVE INDEPENDENT VARIABLES> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**ANOVA.** Performs univariate and multivariate analysis of variance for balanced data, including Latin-square, certain balanced incomplete block designs, completely nested (hierarchical) designs. Options: numerous means comparisons, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L7d1, L7d2, L7e **Precision:** Single **Availability:** CAMVAX.

**ANOVA.** Performs one- to five-way analysis of variance and covariance for factorial designs. Options: decomposition of sum of squares by classical, regression, or hierarchical approach; control of order of entry of covariates and factor main effects. *Proprietary stand-alone program using SPSS command language.* **Classes:** L7a1, L7d1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**ANWAY.** Analyze a balanced n-way classification model with fixed effects. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7d1 **Usage:** CALL ANWAY (*9 parameters*) **Precision:** Single (Double: DANWAY) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AONEC.** Analyze a one-way classification model with covariates. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7a1 **Usage:** CALL AONEC (*21 parameters*) **Precision:** Single (Double: DAONEC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AONEW.** Analyze a one-way classification model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7a1 **Usage:** CALL AONEW (*8 parameters*) **Precision:** Single (Double: DAONEW) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AORDR.** Reordering of the data obtained from any balanced complete experimental design. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7d1 **Usage:** CALL AORDR (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**AOV1.** Compute and print a one-way analysis of variance of the input data. *Portable Fortran subroutine in STARPAC.* **Classes:** L7a **Usage:** CALL AOV1 (Y, TAG, N, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AOV1S.** Compute and optionally print a one-way analysis of variance of the input data; return tag value of each group, number of observations in each group, group averages, and group standard deviations. *Portable Fortran subroutine in STARPAC.* **Classes:** L7a **Usage:** CALL AOV1S (*8 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**AOVONEWAY.** Performs a one-way analysis of variance and prints standard results. *Command in MINITAB proprietary interactive system.* **Classes:** L7a1 **Usage:** AOVOneway on data in C, ..., C **Precision:** Single **Availability:** 855NOS.

**ARBRB.** Add two band matrices, both in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b5 **Usage:** CALL ARBRB (*13 parameters*) **Precision:** Single (Double: DARBRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ARCBAN.** Analysis of two-way classification design data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7b **Usage:** CALL ARCBAN (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ARCOS.** arccos(x). *Proprietary Fortran function in PORT.* **Classes:** C4a **Usage:** R = ARCOS(X) **Precision:** Single (Double: DARCOS) **Availability:** 855NOS, 205.

**AREA.** Given a sequence of points in the plane, this function computes the area bounded by the closed polygonal curve which passes through the points in the specified order. *Portable Fortran function in CGLIB.* **Classes:** P **Usage:** R = AREA(X, Y, NB, NODES) **Also see:** TRMESH **Precision:** Single **Availability:** 855NOS.

**ARFLT.** Perform autoregressive filter operation with user-supplied filter coefficients; return filtered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c3 **Usage:** CALL ARFLT (Y, N, IAR, PHI, YF, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**ARIMA.** Fits non-seasonal and seasonal models to a time series with p the order of the AR part, d the number of differences, q the order of the MA part and with optional seasonality with period S, AR order P, number of differences D, and MA order Q. Options: starting values, forecasting, save results. *Command in MINITAB proprietary interactive system.* **Classes:** L10a2d2, L10a2d3 **Usage:** ARIMa P=K, D=K, Q=K [P=K, D=K, Q=K, S=K] for data in C [put residuals in C [put predicted values in C [put estimated parameters in C]]] [; subcommands CONSTANT or NOCONSTANT; STARTING values in C; FORECAST [forecast origin = K] up to K leads ahead [store forecasts in C [confidence limits in C, C]].] **Precision:** Single **Availability:** 855NOS.

**ARMME.** Compute method of moments estimates of the autoregressive parameters of an ARMA model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2d2 **Usage:** CALL ARMME (*6 parameters*) **Precision:** Single (Double: DARMME) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ARSIN.** arcsin(x). *Proprietary Fortran function in PORT.* **Classes:** C4a **Usage:** R = ARSIN(X) **Precision:** Single (Double: DARSIN) **Availability:** 855NOS, 205.

**ASIN.** arcsin(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ASIN(X) **Precision:** Single (Double: DASIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ASINH.** arcsinh(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** R = ASINH(X) **Precision:** Single (Double: DASINH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ASINH.** arcsinh(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ASINH(X) **Precision:** Single (Double: DASINH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ASINH.** arcsinh(x). *Proprietary Fortran function in PORT.* **Classes:** C4c **Usage:** R = ASINH(X) **Precision:** Single (Double: DASINH) **Availability:** 855NOS, 205.

**ASNKMC.** Student-Newman-Keuls multiple comparison test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a1 **Usage:** CALL ASNKMC (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ASSIGN.** Assigns each case to the cluster whose center is a minimum Euclidean distance from the case. Can be used with CLUSTER subroutine RELOC to form a user-defined K-means package. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL ASSIGN(*9 parameters*) **Also see:** RELOC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**ATAN.** arctan(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ATAN(X) **Precision:** Single (Double: DATAN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ATAN2.** arctan(x/y). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ATAN2(SN, CS) **Precision:** Single (Double: DATAN2) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ATANH.** arctanh(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** R = ATANH(X) **Precision:** Single (Double: DATANH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ATANH.** arctanh(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = ATANH(X) **Precision:** Single (Double: DATANH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ATANH.** arctanh(x). *Proprietary Fortran function in PORT.* **Classes:** C4c **Usage:** R = ATANH(X) **Precision:** Single (Double: DATANH) **Availability:** 855NOS, 205.

**ATWOB.** Analyze a randomized block design or a two-way balanced design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7b **Usage:** CALL ATWOB (*9 parameters*) **Precision:** Single (Double: DATWOB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**AUTOCO.** Computes the sample autocorrelation coefficient of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L10a2a1 **Usage:** CALL AUTOCO(X, N, IWRITE, XAUTOC) **Precision:** Single **Availability:** 855NOS, 840NOS.

**AZA4D.** Complex divide. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** A4 **Usage:** CALL AZA4D(X1R, X1I, X2R, X2I, X3R, X3I, KE) **Precision:** Double **Availability:** PC.

# B

**B2INK.** Computes parameters of a piecewise-polynomial that interpolates a given set of two-dimensional gridded data. (Use B2VAL to evaluate function.) *Portable Fortran subroutine in CMLIB (TENSBS sublibrary).* **Classes:** E2a **Usage:** CALL B2INK(*13 parameters*) **Also see:** B2VAL **Precision:** Single (Double: DB2INK) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**B2VAL.** Evaluates the two-dimensional interpolating function computed by B2INK or one of its partial derivatives. *Portable Fortran subroutine in CMLIB (TENSBS sublibrary).* **Classes:** E3a1, E3a2 **Usage:** CALL B2VAL(*12 parameters*) **Precision:** Single (Double: DB2VAL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**B3INK.** Computes parameters of a piecewise-polynomial that interpolates a given set of three-dimensional gridded data. (Use B3VAL to evaluate function.) *Portable Fortran subroutine in CMLIB (TENSBS sublibrary).* **Classes:** E2a **Usage:** CALL B3INK(*13 parameters*) **Also see:** B3VAL **Precision:** Single (Double: DB3INK) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**B3VAL.** Evaluates the three-dimensional interpolating function computed by B3INK or one of its partial derivatives. *Portable Fortran subroutine in CMLIB (TENSBS sublibrary).* **Classes:** E3a1, E3a2 **Usage:** CALL B3VAL(*12 parameters*) **Precision:** Single (Double: DB3VAL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BAKVEC.** Forms eigenvectors of certain real non-symmetric tridiagonal matrix from symmetric tridiagonal matrix output from FIGI. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL BAKVEC(NM, N, T, E, M, Z, IERR) **Also see:** FIGI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BALANC.** Balances a general real matrix and isolates eigenvalues whenever possible. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1a **Usage:** CALL BALANC(NM, N, A, LOW, IGH, SCALE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**BALBAK.** Forms eigenvectors of real general matrix from eigenvectors of matrix output from BALANC. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL BALBAK(NM, N, LOW, IGH, SCALE, M, Z) **Also see:** BALANC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**BANDR.** Reduces real symmetric band matrix to symmetric tridiagonal matrix and, optionally, accumulates orthogonal similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b1 **Usage:** CALL BANDR(NM, N, MB, A, D, E, E2, MATZ, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BANDV.** Forms eigenvectors of real symmetric band matrix associated with a set of ordered approximate eigenvalues by inverse iteration. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c3 **Usage:** CALL BANDV(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BBDGEL.** To solve the linear system of equations AX = B, where A is a banded block matrix, and X and B are rectangular matrices. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a2 **Usage:** CALL BBDGEL (*12 parameters*) **Precision:** Single **Availability:** 205 (vectorized)

**BBDMPY.** To perform the matrix multiplication C = AB, where A is a rectangular banded block matrix, and B and C are rectangular matrices. A vectorized outer product method is used. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b6 **Usage:** CALL BBDMPY (*12 parameters*) **Precision:** Single **Availability:** 205 (vectorized)

**BCLSF.** Solve a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1b1a1, K1b2a **Usage:** CALL BCLSF (*15 parameters*) **Precision:** Single (Double: DBCLSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCLSJ.** Solve a nonlinear least squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1b1a2, K1b2a **Usage:** CALL BCLSJ (*16 parameters*) **Precision:** Single (Double: DBCLSJ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCOAH.** Minimize a function of N variables subject to bounds on the variables using a modified Newton method and a user-supplied Hessian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2h1a3 **Usage:** CALL BCOAH (*14 parameters*) **Precision:** Single (Double: DBCOAH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCODH.** Minimize a function of N variables subject to bounds on the variables using a modified Newton method and a finite-difference Hessian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2h1a2 **Usage:** CALL BCODH (*13 parameters*) **Precision:** Single (Double: DBCODH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCONF.** Minimize a function of N variables subject to bounds on the variables using a quasi-Newton method and a finite-difference gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1b1a1, K1b2a **Usage:** CALL BCONF (*12 parameters*) **Precision:** Single (Double: DBCONF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCONG.** Minimize a function of N variables subject to bounds on the variables using a quasi-Newton method and a user-supplied gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2h1a2 **Usage:** CALL BCONG (*13 parameters*) **Precision:** Single (Double: DBCONG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCPOL.** Minimize a function of N variables subject to bounds on the variables using a direct search Complex algorithm. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2h1b **Usage:** CALL BCPOL (*10 parameters*) **Precision:** Single (Double: DBCPOL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BCTR.** Perform a forward or an inverse Box-Cox (power) transformation. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L2a, L10a1b **Usage:** CALL BCTR (*7 parameters*) **Precision:** Single (Double: DBCTR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDCOU1.** Tally of observations into a one-way frequency table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL BDCOU1 (X, N, K, DIV, BU, BL, TAB, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDCOU2.** Tally of observations into a two-way frequency table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL BDCOU2 (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDGEL.** To solve the linear system of equations AX = B, where A is a banded matrix, and X and B are rectangular matrices. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a2 **Usage:** CALL BDGEL (*9 parameters*) **Precision:** Single (Half: HBDGEL) **Availability:** 205 (vectorized)

**BDLTV.** Produce letter value summary. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1a1 **Usage:** CALL BDLTV (X, N, NUM, SUMRY, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDTAB.** Computations of frequencies of multivariate data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL BDTAB (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDTRGI.** Transgeneration of the columns of a matrix (in-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L2a **Usage:** CALL BDTRGI (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDTRGO.** Transgeneration of the columns of a matrix (out-of-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L2a **Usage:** CALL BDTRGO (X, NT, ITRG, IT, C, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BDTWT.** Computations of a two-way frequency table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b, L9b **Usage:** CALL BDTWT (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECOR.** Estimates of means, standard deviations, and correlation coefficients (out-of-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1 **Usage:** CALL BECOR (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECORI.** Estimates of means, standard deviations, and correlation coefficients (in-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1 **Usage:** CALL BECORI (X, N, M, IX, XM, S, R, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECOVM.** Means and variance-covariance matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1 **Usage:** CALL BECOVM (X, IX, NBR, TEMP, XM, VCV, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECOVW.** Means and variance-covariance or correlation matrix from data possibly containing missing observations, with weighting on option. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1 **Usage:** CALL BECOVW (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECTR.** Tetrachoric correlation coefficient estimation. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b4 **Usage:** CALL BECTR (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECVL.** Variances and covariances of linear functions (out-of-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1b **Usage:** CALL BECVL (X, M, Y, C, R, V) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BECVLI.** Variances and covariances of linear functions (in-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1b **Usage:** CALL BECVLI (X, N, M, IX, C, R, IOPT, V) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEGRPS.** Moments estimation for grouped data with and without Sheppards corrections. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1a3 **Usage:** CALL BEGRPS (N, C, CI, U, UC, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEI0.** Kelvin function $\mathrm{bei}_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = BEI0(X) **Precision:** Single (Double: DBEI0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEI1.** Kelvin function $\mathrm{bei}_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = BEI1(X) **Precision:** Single (Double: DBEI1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEIGRP.** Estimation of basic statistical parameters using grouped data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1a3 **Usage:** CALL BEIGRP (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEIP0.** Derivative of the Kelvin function $bei_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = BEIP0(X) **Precision:** Single (Double: DBEIP0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEIUGR.** Estimation of basic statistical parameters using ungrouped data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1a1 **Usage:** CALL BEIUGR (Y, N, IOPT, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BELBIN.** Interval estimate of the parameter p of the binomial distribution. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4b **Usage:** CALL BELBIN (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BELPOS.** Interval estimate of the parameter lambda of the Poisson distribution. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4p **Usage:** CALL BELPOS (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMDP.** Median polish of a two-way table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7b, L9d **Usage:** CALL BEMDP (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMIRI.** Estimates means, simple regression coefficients, their intercepts, standard errors of the regression coefficients, and standard deviations for arrays which contain missing values (in-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L8a1a1 **Usage:** CALL BEMIRI (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMIRO.** Estimates means, simple regression coefficients, their intercepts, standard errors of the regression coefficients, and standard deviations for arrays which contain missing values (out-of-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L8a1a1 **Usage:** CALL BEMIRO (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMMI.** Estimates of means, standard deviations, correlation coefficients, and coefficients of skewness and kurtosis from a data matrix containing missing observations (in-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c2 **Usage:** CALL BEMMI (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMMO.** Estimates of means, std. devs., correlation coefficients, and coefficients of skewness and kurtosis from a data matrix containing missing observations (out-of-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c2 **Usage:** CALL BEMMO (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMNON.** Location (mean) inferences using a sample from a normal population with known variance. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4n **Usage:** CALL BEMNON (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEMSON.** Mean and variance inferences using a sample from a normal population. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4n **Usage:** CALL BEMSON (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BENSON.** Variance inferences using a sample from a normal population with known mean. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4n **Usage:** CALL BENSON (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEPAT.** Mean and variance inferences using samples from each of two normal populations with unequal variances. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b1a4 **Usage:** CALL BEPAT (Y, N, IOP, CRIT, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BEPET.** Mean and variance inferences using samples from each of two normal populations with equal variances. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b1a4 **Usage:** CALL BEPET (Y, N, IOP, CRIT, STAT, NDF, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BER0.** Kelvin function $ber_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = BER0(X) **Precision:** Single (Double: DBER0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BER1.** Kelvin function $ber_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = BER1(X) **Precision:** Single (Double: DBER1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BERP0.** Derivative of the Kelvin function $ber_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10c **Usage:** R = BERP0(X) **Precision:** Single (Double: DBERP0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BESCI.** Sequence of modified Bessel functions $I_n(z)$. *Proprietary Fortran subroutine in PORT.* **Classes:** C10b2 **Usage:** CALL BESCI (XR, XI, NB, BR, BI) **Precision:** Single (Double: DBESCI) **Availability:** 855NOS, 205.

**BESCJ.** Sequence of Bessel functions $J_n(z)$. *Proprietary Fortran subroutine in PORT.* **Classes:** C10a2 **Usage:** CALL BESCJ (XR, XI, NB, BR, BI) **Precision:** Single (Double: DBESCJ) **Availability:** 855NOS, 205.

**BESI0.** Modified Bessel function $I_0(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1

**Usage:** R = BESI0(X) **Precision:** Single (Double: DBESI0) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESI0.** Modified Bessel function $I_0(x)$. *Portable Fortran software in NMS library.* **Classes:** C10b1 **Usage:** R = BESI0(X) **Precision:** Single (Double: DBESI0) **Availability:** PC.

**BESI0E.** Exponentially scaled modified Bessel function $\exp(-|x|)I_0(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESI0E(X) **Precision:** Single (Double: DBSI0E) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESI1.** Modified Bessel function $I_1(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESI1(X) **Precision:** Single (Double: DBESI1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESI1E.** Exponentially scaled modified Bessel function $\exp(-|x|)I_1(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESI1E(X) **Precision:** Single (Double: DBSI1E) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESJ.** Sequence of Bessel functions, $J_{\alpha+k-1}(x)$ k=1,...,N for positive real $\alpha$, x. Uses internal double precision arithmetic. *Portable Fortran subroutine in CMLIB (AMOSLIB sublibrary).* **Classes:** C10a3 **Usage:** CALL BESJ(X, ALPHA, N, Y, NZ) **Precision:** Single (Double: DBESJ) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESJ.** Sequence of Bessel functions, $J_{\alpha+k-1}(x)$, k=1,..., n for non-negative $\alpha$ and x. *Portable Fortran software in NMS library.* **Classes:** C10a3 **Usage:** CALL BESJ(X, ALPHA, N, Y, NZ) **Precision:** Single (Double: DBESJ) **Availability:** PC.

**BESJ0.** Bessel function $J_0(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10a1 **Usage:** R = BESJ0(X) **Precision:** Single (Double: DBESJ0) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESJ1.** Bessel function $J_1(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10a1 **Usage:** R = BESJ1(X) **Precision:** Single (Double: DBESJ1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESK0.** Modified Bessel function $K_0(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESK0(X) **Precision:** Single (Double: DBESK0) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESK0E.** Exponentially scaled modified Bessel function $\exp(x)K_0(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESK0E(X) **Precision:** Single (Double: DBSK0E) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESK1.** Modified Bessel function $K_1(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESK1(X) **Precision:** Single (Double: DBESK1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESK1E.** Exponentially scaled modified Bessel function $\exp(x)K_1(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10b1 **Usage:** R = BESK1E(X) **Precision:** Single (Double: DBSK1E) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESKES.** Sequence of exponentially scaled Bessel functions $\exp(x)K_r(x)$, for r real. *Portable Fortran subroutine in CMLIB (FNLIB sublibrary).* **Classes:** C10b3 **Usage:** CALL BESKES(XNU, X, N, BK) **Precision:** Single (Double: DBSKES) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESKS.** Sequence of Bessel functions $K_r(x)$, for r real. *Portable Fortran subroutine in CMLIB (FNLIB sublibrary).* **Classes:** C10b3 **Usage:** CALL BESKS(XNU, X, N, BK) **Precision:** Single (Double: DBESKS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESPAK.** Subroutines for Bessel and modified Bessel functions of complex argument and real (integer or fractional) order. BESPAK consists of four main subroutines, one for each Bessel function, and four subsidiary routines. *Portable Fortran software in BESPAK library.* **Classes:** C10a, C10b **Usage:** CALL BESICF(*4 parameters*), CALL BESJCF(*4 parameters*), CALL BESKCF(*4 parameters*), CALL BESYCF(*4 parameters*) **Also see:** BACKLC GAM1 PHASMP RECIPG **Precision:** Single **Availability:** 855NOS (In source form only.)

**BESRB.** Biserial and point-biserial correlation coefficients for a qualitatively dichotomized variable and a numerically measurable and classified variable. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b4 **Usage:** CALL BESRB (N, A, IA, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BESRI.** Sequence of modified Bessel functions $I_n(x)$. *Proprietary Fortran subroutine in PORT.* **Classes:** C10b1 **Usage:** CALL BESRI (X, NB, B) **Precision:** Single (Double: DBESRI) **Availability:** 855NOS, 205.

**BESRJ.** Bessel functions, J, of real argument and integer order. *Proprietary Fortran subroutine in PORT.* **Classes:** C10a1 **Usage:** CALL BESRJ (X, NB, B) **Precision:** Single (Double: DBESRJ) **Availability:** 855NOS, 205.

**BESRN.** Biserial correlation coefficient for a qualitatively dichotomized variable and a numerically or qualitatively classified variable. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b4 **Usage:** CALL BESRN (N, A, IA, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BESTA2.** Computations of confidence intervals and other basic statistics using output from IMSL routine BESTAT. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4c1a **Usage:** CALL BESTA2 (*11 parameters*) **Also see:** BESTAT **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BESTAT.** Computations of basic univariate statistics from data possibly containing missing values, with weighting on option. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1c1 **Usage:** CALL BESTAT (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**BESY.** Sequence of Bessel functions Y(alpha + k-1)(x), k=1,n for positive x and non-negative alpha. Uses internal double precision arithmetic. *Portable Fortran subroutine in CMLIB (AMOSLIB sublibrary).* **Classes:** C10a3 **Usage:** CALL BESY(X, FNU, N, Y) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESY0.** Modified Bessel function $Y_0(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10a1 **Usage:** R = BESY0(X) **Precision:** Single (Double: DBESY0) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BESY1.** Modified Bessel function $Y_1(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10a1 **Usage:** R = BESY1(X) **Precision:** Single (Double: DBESY1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BETA.** Beta(a,b), for positive real a,b. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7b **Usage:** R = BETA(A, B) **Precision:** Single (Double: DBETA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BETA.** Complete beta function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** C7b **Usage:** R = BETA(A, B) **Precision:** Single (Double: DBETA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BETA PROBABILITY PLOT.** Generates a probability plot for the beta distribution with parameters $\alpha$ and $\beta$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2b **Usage:** BETA PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**BETAI.** Incomplete Beta function. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7f **Usage:** R = BETAI(X, A, B) **Precision:** Single (Double: DBETAI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BETAI.** Incomplete beta function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7b **Usage:** R = BETAI(X, PIN, QIN) **Precision:** Single (Double: DBETAI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BETDF.** Beta probability distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1b, C7b **Usage:** R = BETDF(X, PIN, QIN) **Precision:** Single (Double: DBETDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BETIN.** Inverse of the beta distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a2b, C7b **Usage:** R = BETIN(P, PIN, QIN) **Precision:** Single (Double: DBETIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BETRAN.** Generates a random sample of size N from the beta distribution with parameters ALPHA and BETA. *Fortran subroutine in DATAPAC.* **Classes:** L6a2 **Usage:** CALL BETRAN(N, ALPHA, BETA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**BFQAD.** Integrates function times derivative of B-spline from X1 to X2. The B-spline is in B representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a3, H2a2a1, K6a3 **Usage:** CALL BFQAD(*12 parameters*) **Precision:** Single (Double: DBFQAD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BFS.** Compute and print a bivariate Fourier spectrum analysis of a pair of series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFS (Y1, Y2, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSF.** Compute and print a bivariate Fourier spectrum analysis of a pair of series; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSF (*5 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSFS.** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series using user-supplied control values; use FFT for computations; return squared coherency and phase components of the cross spectrum and the corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSFS (*16 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSM.** Compute and print a bivariate Fourier spectrum analysis of a pair of series with missing observations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSM (Y1, YMISS1, Y2, YMISS2, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSMS.** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series with missing observations using user-supplied control values; return squared coherency and phase components of the cross spectrum and the corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSMS (*17 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSMV.** Compute and print a bivariate Fourier spectrum analysis of a pair of series with missing observations; input covariances rather than original series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSMV (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSMVS.** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series with missing observations using user-supplied control values; input covariances rather than original series; return squared coherency and phase components of the cross spectrum and the corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSMVS (*21 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSS.** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series using user-supplied control values; return squared coherency and phase components of the cross spectrum and the corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSS (*15 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSV.** Compute and print a bivariate Fourier spectrum analysis of a pair of series; input covariances rather than original series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSV (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BFSVS.** Compute and optionally print a bivariate Fourier spectrum analysis of a pair of series using user-supplied control values; input covariances rather than original series; return squared coherency and phase components of the cross spectrum and the corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b3a4 **Usage:** CALL BFSVS (*18 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**BHAKV.** Perform a Bhapkar V test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4c1b **Usage:** CALL BHAKV (NGROUP, NI, Y, V, PROB) **Precision:** Single (Double: DBHAKV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BI.** Airy function Bi(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10d **Usage:** R = BI(X) **Precision:** Single (Double: DBI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BI.** Airy function Bi(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = BI(X) **Precision:** Single (Double: DBI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BID.** Derivative of the Airy function Bi(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = BID(X) **Precision:** Single (Double: DBID) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BIDE.** Exponentially scaled derivative of the Airy function exp(-x)dBi(x)/ dx. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = BIDE(X) **Precision:** Single (Double: DBIDE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BIE.** Exponentially scaled Airy function exp(-x)Bi(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C10d **Usage:** R = BIE(X) **Precision:** Single (Double: DBIE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BIE.** Exponentially scaled Airy function exp(-x)Bi(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10d **Usage:** R = BIE(X) **Precision:** Single (Double: DBIE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BIHISTOGRAM.** Generate a bi-histogram of two-dimensional data. *Command(s) in DATAPLOT interactive system.* **Classes:** L3b1 **Usage:** BIHISTOGRAM <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**BINCDF.** Computes the cumulative distribution function value at X for the binomial distribution with parameters P and N. *Fortran subroutine in DATAPAC.* **Classes:** L5a1b **Usage:** CALL BINCDF(X, P, N, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**BINDF.** Evaluate the binomial distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1b **Usage:** R = BINDF(K, N, P) **Precision:** Single (Double: DBINDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BINES.** Estimate the parameter p of the binomial distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1a4b **Usage:** CALL BINES (*6 parameters*) **Precision:** Single (Double: DBINES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BINOM.** Binomial coefficient. Input is integer, output is real. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C1 **Usage:** R = BINOM(N, M) **Precision:** Single (Double: DBINOM) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BINOM.** Binomial coefficient. Input is integer, output is real. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** C1 **Usage:** R = BINOM(N, M) **Precision:** Single (Double: DBINOM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BINOMIAL.** Prints table of binomial probabilities and cumulative distribution function, and optionally saves results.

*Command in MINITAB proprietary interactive system.* **Classes:** L5a1b **Usage:** BINOmial probabilities for N=K, P=K [store probabilities in C] **Precision:** Single **Availability:** 855NOS.

**BINOMIAL PROBABILITY PLOT.** Generates a probability plot for the binomial distribution with parameters P and N. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2b **Usage:** BINOMIAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> SUBSET/EXCEPT/FOR <QUALIFICATION>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**BINPPF.** Computes the percent point function value at P for the binomial distribution with parameters PPAR and N. *Fortran subroutine in DATAPAC.* **Classes:** L5a2b **Usage:** CALL BINPPF(P, PPAR, N, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**BINPR.** Evaluate the binomial probability function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1b **Usage:** R = BINPR(K, N, P) **Precision:** Single (Double: DBINPR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BINRAN.** Generates a random sample of size N from the binomial distribution with parameters P and NPAR. *Fortran subroutine in DATAPAC.* **Classes:** L6a2 **Usage:** CALL BINRAN(N, P, NPAR, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**BINT4.** Computes B-spline which interpolates given X,Y data with various end conditions. The B representation is used. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E1a **Usage:** CALL BINT4(*13 parameters*) **Also see:** BVALU for evaluation. See package documentation for other facilities. **Precision:** Single (Double: DBINT4) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BINTK.** Produces coefficients of k-th order B-spline with given knots and with values at given points. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E1a **Usage:** CALL BINTK(X, Y, T, N, K, BCOEF, Q, WORK) **Also see:** BVALU for evaluation. See package documentation for other facilities. **Precision:** Single (Double: DBINTK) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BISECT.** Compute eigenvalues of symmetric tridiagonal matrix in given interval using Sturm sequencing. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL BISECT(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BLINF.** Compute the bilinear mode $x^T Ay$. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** R = BLINF(NRA, NCA, A, LDA, X, Y) **Precision:** Single (Double: DBLINF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BLKTRI.** Solves block tridiagonal systems of linear algebraic equations arising from the discretization of separable elliptic partial differential equations. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b4b **Usage:** CALL BLKTRI(*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BLOCAR.** Locally fits autoregressive models to non-stationary time series by a Bayesian procedure. Power spectra for stationary spans are graphically printed out. (This program is tentative and further testing and improvement through practical applications will be necessary.) *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2f **Precision:** Single **Availability:** 855NOS.

**BLOMAR.** Locally fits multivariate autoregressive models to non-stationary time series by a Bayesian procedure. (This program is tentative and further testing and improvement through practical applications will be necessary.) *Portable stand-alone program using TIMSAC command language.* **Classes:** L10c **Precision:** Single **Availability:** 855NOS.

**BNDACC.** Introduce new blocks of data for banded least squares problems. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** D9a1 **Usage:** CALL BNDACC(G, MDG, NB, IP, IR, MT, JT) **Also see:** BNDSOL **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BNDSOL.** Solves least squares problem AX=B for banded matrices. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** D9a1 **Usage:** CALL BNDSOL(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BNODES.** Given a triangulation of a set of points in the plane, returns the nodes on the boundary of the convex hull of the set of points. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL BNODES (*7 parameters*) **Also see:** TRMESH **Precision:** Single **Availability:** 855NOS.

**BNRDF.** Evaluate the bivariate normal distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5b1n **Usage:** R = BNRDF(X, Y, RHO) **Precision:** Single (Double: DBNRDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BOOTSTRAP PLOT.** Generate a bootstrap plot for a given statistic, where the statistic is selected from among 29 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics or is user defined. The generated bootstrap samples are available for further analysis. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1b1, L4b1b **Usage:** BOOTSTRAP <STATISTIC> PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**BOX PLOT.** Plots box plot(s) – median, "body" between lower and upper hinges, tails (between hinges and minimum or maximum) – for either univariate distributional analysis or analysis of 1-factor models. *Command(s) in DATA-*

*PLOT interactive system.* **Classes:** L3a3, L3e4 **Usage:** BOX PLOT <RESPONSE VARIABLE> [<INDEPENDENT VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**BOX-COX LINEARITY PLOT.** Generate a correlation plot for the Box-Cox family of transformations. The horizontal axis is the lambda parameter for the Box-Cox family of transformations and the vertical axis is the correlation coefficient between the transformed variables. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a5 **Usage:** BOX-COX LINEARITY PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**BOX-COX NORMALITY PLOT.** Plots probability plot correlation coefficient vs. Box-Cox transformation parameter $\lambda$ ($\lambda$ ranging from -2 to +2 or in user-set range) – for determining the "best" transformation from the Box-Cox family to normalize data which are commonly neither normal nor symmetric. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3n, L4a1a5 **Usage:** BOX-COX NORMALITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**BOX-JENKINS.** Box-Jenkins modeling and forecasting of time series data. Options: log or power transformation, seasonal or nonseasonal differencing, parameter selection (autoregressive and/or moving average, seasonal or nonseasonal), control of estimation. *Proprietary stand-alone program using SPSS command language.* **Classes:** L10a2d **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**BOXES.** Produces a plot in the form of a 3-dimensional box of three of more variables for each case. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e4 **Usage:** CALL BOXES(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**BOXP.** Print boxplots for one or more samples. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3a3, L3e4 **Usage:** CALL BOXP (NGROUP, NI, X, TITLE) **Precision:** Single (Double: DBOXP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BOXPLOT.** Prints boxplots – median, hinges, inner and outer fences – for one or more levels. Options: form of plots, notches (confidence interval for population medians). *Command in MINITAB proprietary interactive system.* **Classes:** L3a3 **Usage:** BOXPlots for data in C [levels in C] [; subcommands LINES = K; NOTCH the boxplots; LEVELS K, ..., K [for C].] **Precision:** Single **Availability:** 855NOS.

**BQR.** Computes some of the eigenvalues of a real symmetric band matrix using the QR method with shifts of origin. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a6 **Usage:** CALL BQR(NM, N, MB, A, T, R, IERR, NV, RV) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BQUAD.** Adaptively integrates functions which have discontinuities in their derivatives. User can specify these points. *Proprietary Fortran subroutine in PORT.* **Classes:** H2a2a1 **Usage:** CALL BQUAD (F, N, X, EPS, ANS, ERREST) **Precision:** Single (Double: DBQUAD) **Availability:** 855NOS, 205.

**BRANDOM.** Generates K pseudo-random numbers from binomial distribution (number of successes in n Bernoulli trials with probability p of success). *Command in MINITAB proprietary interactive system.* **Classes:** L6a2 **Usage:** BRANdom K binomial experiments with n = K, p = K, put into C **Precision:** Single **Availability:** 855NOS.

**BREAKDOWN.** Computes means, standard deviations, variances and subpopulation sizes for a dependent variable over subgroups defined by as many as five independent variables. Options: one-way ANOVA, test for linearity, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1c1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**BS2DR.** Evaluate the derivative of a two-dimensional tensor-product spline, given its tensor-product B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a2 **Usage:** R = BS2DR(*11 parameters*) **Precision:** Single (Double: DBS2DR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS2IG.** Evaluate the integral of a tensor-product spline on a rectangular domain, given its tensor-product B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a3, H2b1b2 **Usage:** R = BS2IG(*11 parameters*) **Precision:** Single (Double: DBS2IG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS2IN.** Compute a two-dimensional tensor-product spline interpolant, returning the tensor-product B-spline coefficients. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E2a **Usage:** CALL BS2IN (*11 parameters*) **Precision:** Single (Double: DBS2IN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS2VL.** Evaluate a two-dimensional tensor-product spline, given its tensor-product B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a1 **Usage:** R = BS2VL(*9 parameters*) **Precision:** Single (Double: DBS2VL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS3DR.** Evaluate the derivative of a three-dimensional tensor-product spline, given its tensor-product B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a2 **Usage:** R = BS3DR(*16 parameters*) **Precision:** Single (Double: DBS3DR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS3IG.** Evaluate the integral of a tensor-product spline in three dimensions over a three-dimensional rectangle, given its tensor-product B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a3, H2b1b2 **Usage:** R = BS3IG(*16 parameters*) **Precision:** Single (Double: DBS3IG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS3IN.** Compute a three-dimensional tensor-product spline interpolant, returning the tensor-product B-spline coefficients. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E2a **Usage:** CALL BS3IN (*16* *parameters*) **Precision:** Single (Double: DBS3IN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BS3VL.** Evaluate a three-dimensional tensor-product spline, given its tensor-product B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a1 **Usage:** R = BS3VL(*13 parameters*) **Precision:** Single (Double: DBS3VL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSCAT.** Compute the biserial correlation coefficient for a dichotomous variable and a classification variable. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b4 **Usage:** CALL BSCAT (K, A, LDA, STAT) **Precision:** Single (Double: DBSCAT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSCPP.** Convert a spline in B-spline representation to piecewise polynomial representation. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E3c, K6c **Usage:** CALL BSCPP (*7 parameters*) **Precision:** Single (Double: DBSCPP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSDER.** Evaluate the derivative of a spline, given its B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a2, K6a2 **Usage:** R = BSDER(*6 parameters*) **Precision:** Single (Double: DBSDER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSI0.** Modified Bessel function $I_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSI0(X) **Precision:** Single (Double: DBSI0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSI0E.** Exponentially scaled modified Bessel function $\exp(-x)I_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSI0E(X) **Precision:** Single (Double: DBSI0E) **Availability:** 855NOS, 855VE, 205; 840NOS.

**BSI1.** Modified Bessel function $I_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSI1(X) **Precision:** Single (Double: DBSI1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSI1E.** Exponentially scaled modified Bessel function $\exp(-x)I_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSI1E(X) **Precision:** Single (Double: DBSI1E) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSIES.** Sequence of exponentially scaled modified Bessel functions $\exp(-x)I_r(x)$, for r real and x positive. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10b3 **Usage:** CALL BSIES (XNU, X, N, BSI) **Precision:** Single (Double: DBSIES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSINS.** Sequence of modified Bessel functions $I_n(x)$. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** CALL BSINS (X, N, BSI) **Precision:** Single (Double: DBSINS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSINT.** Compute the spline interpolant, returning the B-spline coefficients. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL BSINT (*6 parameters*) **Precision:** Single (Double: DBSINT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSIS.** Sequence of modified Bessel functions $I_r(x)$, for r nonnegative real and x positive. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10b3 **Usage:** CALL BSIS (XNU, X, N, BSI) **Precision:** Single (Double: DBSIS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSITG.** Evaluate the integral of a spline, given its B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a3, H2a2b1 **Usage:** R = BSITG(*6 parameters*) **Precision:** Single (Double: DBSITG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSJ0.** Bessel function $J_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10a1 **Usage:** R = BSJ0(X) **Precision:** Single (Double: DBSJ0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSJ1.** Bessel function $J_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10a1 **Usage:** R = BSJ1(X) **Precision:** Single (Double: DBSJ1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSJNS.** Sequence of Bessel functions $J_n(x)$. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10a1 **Usage:** CALL BSJNS (X, N, BS) **Precision:** Single (Double: DBSJNS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSJS.** Sequence of Bessel functions $J_r(x)$, for r real and positive. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10a3 **Usage:** CALL BSJS (XNU, X, N, BS) **Precision:** Single (Double: DBSJS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSK0.** Bessel function $K_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSK0(X) **Precision:** Single (Double: DBSK0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSK0E.** Exponentially scaled modified Bessel function $\exp(x)K_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSK0E(X) **Precision:** Single (Double: DBSK0E) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSK1.** Bessel function $K_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSK1(X) **Precision:** Single (Double: DBSK1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSK1E.** Exponentially scaled modified Bessel function $\exp(x)K_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10b1 **Usage:** R = BSK1E(X) **Precision:** Single (Double: DBSK1E) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSKES.** Sequence of exponentially scaled modified Bessel functions $\exp(x)K_r(x)$, for real r. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10b3 **Usage:** CALL BSKES (XNU, X, NIN, BKE) **Precision:**

Single (Double: DBSKES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSKS.** Sequence of modified Bessel functions $K_r(x)$, for real r. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10b3 **Usage:** CALL BSKS (XNU, X, NIN, BK) **Precision:** Single (Double: DBSKS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSLS2.** Compute a two-dimensional tensor-product spline approximant using least squares, returning the tensor product B-spline coefficients. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1a1b **Usage:** CALL BSLS2 (*15 parameters*) **Precision:** Single (Double: DBSLS2) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSLSQ.** Compute a B-spline least squares spline approximation to given data. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1a1a1 **Usage:** CALL BSLSQ (*8 parameters*) **Precision:** Single (Double: DBSLSQ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSNAK.** Compute the not-a-knot spline knot sequence. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E3b **Usage:** CALL BSNAK (NDATA, XDATA, KORDER, XKNOT) **Precision:** Single (Double: DBSNAK) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSOPK.** Compute the optimal spline knot sequence. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E3b **Usage:** CALL BSOPK (NDATA, XDATA, KORDER, XKNOT) **Precision:** Single (Double: DBSOPK) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSPBS.** Compute the biserial and point-biserial correlation coefficients for a dichotomous variable and a numerically measurable classification variable. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b4 **Usage:** CALL BSPBS (K, A, LDA, STAT) **Precision:** Single (Double: DBSPBS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSPDR.** Constructs divided difference table from B representation of B-spline for a derivative calculation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3c **Usage:** CALL BSPDR(T, A, N, K, NDERIV, AD) **Precision:** Single (Double: DBSPDR) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BSPEV.** Calculates the value of a spline and its derivatives at X from its B representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL BSPEV(*9 parameters*) **Precision:** Single (Double: DBSPEV) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BSPL1.** Evaluates, at a given set of points in a specified mesh interval, basis splines together with selected orders of derivatives. *Proprietary Fortran subroutine in PORT.* **Classes:** E3c, K6c **Usage:** CALL BSPL1 (*9 parameters*) **Precision:** Single (Double: DBSPL1) **Availability:** 855NOS, 205.

**BSPLD.** Evaluates at a given set of points in a specified mesh interval, basis splines and their derivatives. *Proprietary Fortran subroutine in PORT.* **Classes:** E3c, K6c **Usage:** CALL BSPLD (K, T, N, X, NX, ILEFT, MD, BX) **Precision:** Single (Double: DBSPLD) **Availability:** 855NOS, 205.

**BSPLI.** Obtains the integrals of basis splines, from the left-most mesh point to a specified set of points. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a3, H2a2b1, K6a3 **Usage:** CALL BSPLI (K, T, N, X, NX, ILEFT, BIX) **Precision:** Single (Double: DBSPLI) **Availability:** 855NOS, 205.

**BSPLN.** Evaluates at a given set of points in a specified mesh interval, all the basis splines which are nonzero in that interval. *Proprietary Fortran subroutine in PORT.* **Classes:** E3c, K6c **Usage:** CALL BSPLN (K, T, N, X, NX, ILEFT, BX) **Precision:** Single (Double: DBSPLN) **Availability:** 855NOS, 205.

**BSPPP.** Converts from B representation of B-spline to piecewise polynomial representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3c, K6c **Usage:** CALL BSPPP(T, A, N, K, LDC, C, XI, LXI, WORK) **Precision:** Single (Double: DBSPPP) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BSPVD.** Calculates value and derivatives of order less than NDERIV of all B-spline basis functions which do not vanish at X. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3c, K6c **Usage:** CALL BSPVD(*8 parameters*) **Precision:** Single (Double: DBSPVD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BSPVN.** Calculates the value of all (possibly) nonzero B-spline basis functions at X of a given order. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3c, K6c **Usage:** CALL BSPVN(*9 parameters*) **Precision:** Single (Double: DBSPVN) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BSQAD.** Computes the integral of a B-spline from X1 to X2. The B-spline must be in B representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a3, H2a2b1, K6a3 **Usage:** CALL BSQAD(*8 parameters*) **Precision:** Single (Double: DBSQAD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BSUBST.** Produces Bayesian estimates of time series models such as pure AR models, AR models with non-linear terms, AR models with polynomial type mean value functions, etc. The goodness of fit of a model is checked by the analysis of several steps ahead prediction errors. By preparing an external subroutine SETX, any time series model which is linear in its parameters can be analysed. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2c1 **Precision:** Single **Availability:** 855NOS.

**BSVAL.** Evaluate a spline, given its B-spline representation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a1, K6a1 **Usage:** R = BSVAL(*5 parameters*) **Precision:** Single (Double: DBSVAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSVLS.** Compute the variable knot B-spline least squares approximation to given data. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1a1a1 **Usage:** CALL BSVLS (*10 parameters*) **Precision:** Single (Double: DBSVLS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSY0.** Bessel function $Y_0(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10a1 **Usage:** R = BSY0(X) **Precision:** Single (Double: DBSY0) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSY1.** Bessel function $Y_1(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C10a1 **Usage:** R = BSY1(X) **Precision:** Single (Double: DBSY1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BSYS.** Sequence of Bessel functions $Y_r(x)$, for real nonnegative r and positive x. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10a3 **Usage:** CALL BSYS (XNU, X, N, BSY) **Precision:** Single (Double: DBSYS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BTRIALS.** Generates pseudo-random sequence of K 0's and 1's, with the probability p of a 1. *Command in MINITAB proprietary interactive system.* **Classes:** L6a2 **Usage:** BTRIals K Bernoulli trials with p = K, put into C **Precision:** Single **Availability:** 855NOS.

**BUILD.** Builds clusters by the K-means algorithm, printing the results for all intermediate iterations. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL BUILD(*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**BURAM.** Finds the best uniform rational approximation to a given function on a specified mesh. *Proprietary Fortran subroutine in PORT.* **Classes:** K2 **Usage:** CALL BURAM (*8 parameters*) **Also see:** TCHBP **Precision:** Single (Double: DBURAM) **Availability:** 855NOS, 205.

**BURG.** Computes the coefficients of a finite length causal forward or backward prediction filter and uses both the forward and backward predictions in a symmetric manner to generate the maximum entropy spectrum by means of a Toeplitz recursion. *Portable Fortran subroutine in CMLIB (MXENTRP sublibrary).* **Classes:** L10a3a5 **Usage:** CALL BURG(LX, X, F, B, LA, A, M, S, Y, TABLE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BURM1.** Finds the best uniform rational approximation to a given function on a specified mesh, starting from a given initial approximation. *Proprietary Fortran subroutine in PORT.* **Classes:** K2 **Usage:** CALL BURM1 (*10 parameters*) **Also see:** TCHBP **Precision:** Single (Double: DBURM1) **Availability:** 855NOS, 205.

**BVALU.** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its B representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL BVALU(T, A, N, K, IDERIV, X, INBV, WORK) **Precision:** Single (Double: DBVALU) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**BVPFD.** Solve a system of differential equations with boundary conditions at two points, using a variable order, variable step-size finite-difference method with deferred corrections. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I1b2 **Usage:** CALL BVPFD (*24 parameters*) **Precision:** Single (Double: DBVPFD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BVPMS.** Solve a system of differential equations with boundary conditions at two points, using a multiple shooting method. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I1b2 **Usage:** CALL BVPMS (*18 parameters*) **Precision:** Single (Double: DBVPMS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**BVSUP.** Solves boundary value problems for a linear system of ODEs using superposition, orthogonalization, and variable step integration. *Portable Fortran subroutine in CMLIB (BVSUP sublibrary).* **Classes:** I1b1 **Usage:** CALL BVSUP(*21 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

# C

**C02ADF.** Finds all the roots of a complex polynomial equation, using the method of Grant and Hitchins. *Proprietary Fortran subroutine in NAG.* **Classes:** F1a2 **Usage:** CALL C02ADF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C02AEF.** Finds all the roots of a real polynomial equation, using the method of Grant and Hitchins. *Proprietary Fortran subroutine in NAG.* **Classes:** F1a1 **Usage:** CALL C02AEF (A, N, REZ, IMZ, TOL, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05ADF.** Locates a zero of a continuous function in a given interval (a, b) by a combination of the methods of linear interpolation, extrapolation and bisection. *Proprietary Fortran subroutine in NAG.* **Classes:** F1b **Usage:** CALL C05ADF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05AGF.** Locates a simple zero of a continuous function from a given starting value, using a binary search to locate an interval containing a zero of the function, then a combination of the methods of linear interpolation, extrapolation and bisection to locate the zero precisely. *Proprietary Fortran subroutine in NAG.* **Classes:** F1b **Usage:** CALL C05AGF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05AJF.** Attempts to locate a zero of a continuous function by a continuation method using a secant iteration. *Proprietary Fortran subroutine in NAG.* **Classes:** F1b **Usage:** CALL C05AJF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05AVF.** Attempts to locate an interval containing a simple zero of a continuous function f(x) using a binary search. It uses 'reverse communication' for evaluating the function. *Proprietary Fortran subroutine in NAG.* **Classes:** F1b **Usage:** CALL C05AVF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05AXF.** Attempts to locate a zero of a continuous function using a continuation method based on a secant iteration. *Proprietary Fortran subroutine in NAG.* **Classes:** F1b **Usage:** CALL C05AXF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05AZF.** Locates a simple zero of a continuous function on a given interval by a combination of the methods of linear interpolation, linear extrapolation and bisection. It uses reverse communication for evaluating the function. *Proprietary Fortran subroutine in NAG.* **Classes:** F1b **Usage:** CALL C05AZF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05NBF.** Is an easy-to-use routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method. *Proprietary Fortran subroutine in NAG.* **Classes:** F2 **Usage:** CALL C05NBF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05NCF.** Is a comprehensive routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method. *Proprietary Fortran subroutine in NAG.* **Classes:** F2 **Usage:** CALL C05NCF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05PBF.** Is an easy-to-use routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method. The user must provide the Jacobian. *Proprietary Fortran subroutine in NAG.* **Classes:** F2 **Usage:** CALL C05PBF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05PCF.** Is a comprehensive routine to find a zero of a system of N nonlinear functions in N variables by a modification of the Powell hybrid method. The user must provide the Jacobian. *Proprietary Fortran subroutine in NAG.* **Classes:** F2 **Usage:** CALL C05PCF (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C05ZAF.** Checks the user-provided gradients of M non-linear functions in N variables for consistency with the functions themselves. The routine must be called twice. *Proprietary Fortran subroutine in NAG.* **Classes:** F3 **Usage:** CALL C05ZAF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06BAF.** Accelerates the convergence of a given convergent sequence to its limit. *Proprietary Fortran subroutine in NAG.* **Classes:** A7 **Usage:** CALL C06BAF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06DBF.** Returns the value of the sum of a Chebyshev series through the routine name. *Proprietary Fortran function in NAG.* **Classes:** C3a2 **Usage:** R = C06DBF(X, C, N, S) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06EAF.** Calculates the discrete Fourier transform of a sequence of N real data values. (No extra workspace required.) *Proprietary Fortran subroutine in NAG.* **Classes:** J1a1 **Usage:** CALL C06EAF (X, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06EBF.** Calculates the discrete Fourier transform of a Hermitian sequence of N complex data values. (No extra workspace required.) *Proprietary Fortran subroutine in NAG.* **Classes:** J1a2 **Usage:** CALL C06EBF (X, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06ECF.** Calculates the discrete Fourier transform of a sequence of N complex data values. (No extra workspace required.) *Proprietary Fortran subroutine in NAG.* **Classes:** J1a2 **Usage:** CALL C06ECF (X, Y, N, IFAIL) **Preci-**

**sion:** Single **Availability:** 855NOS, 855VE, 205.

**C06EKF.** Calculates the circular convolution or correlation of two real vectors of period N. (No extra workspace is required.) *Proprietary Fortran subroutine in NAG.* **Classes:** D1a10, J2 **Usage:** CALL C06EKF (JOB, X, Y, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FAF.** Calculates the discrete Fourier transform of a sequence of N real data values (using a work array for extra speed). *Proprietary Fortran subroutine in NAG.* **Classes:** J1a1 **Usage:** CALL C06FAF (X, N, WORK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FBF.** Calculates the discrete Fourier transform of a Hermitian sequence of N complex data values. (Uses a work array for extra speed.) *Proprietary Fortran subroutine in NAG.* **Classes:** J1a2 **Usage:** CALL C06FBF (X, N, WORK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FCF.** Calculates the discrete Fourier transform of a sequence of N complex data values. (Uses a work array for extra speed.) *Proprietary Fortran subroutine in NAG.* **Classes:** J1a2 **Usage:** CALL C06FCF (X, Y, N, WORK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FFF.** Computes the discrete Fourier transform of one variable in a multivariate sequence of complex data values. *Proprietary Fortran subroutine in NAG.* **Classes:** J1a2 **Usage:** CALL C06FFF (*9 parameters*) **Also see:** C06CGF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FJF.** Computes the multi-dimensional discrete Fourier transform of a multivariate sequence of complex data values. *Proprietary Fortran subroutine in NAG.* **Classes:** J1b **Usage:** CALL C06FJF (*8 parameters*) **Also see:** C06GCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FKF.** Calculates the circular convolution or correlation of two real vectors of period N. (Uses a work array for extra speed.) *Proprietary Fortran subroutine in NAG.* **Classes:** D1a10, J2 **Usage:** CALL C06FKF (JOB, X, Y, N, WORK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FPF.** Computes the discrete Fourier transforms of M sequences, each containing N real data values. This routine is designed to be particularly efficient on vector processors. *Proprietary Fortran subroutine in NAG.* **Classes:** J1a1 **Usage:** CALL C06FPF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FQF.** Computes the discrete Fourier transforms of M Hermitian sequences, each containing N complex data values. This routine is designed to be particularly efficient on vector processors. *Proprietary Fortran subroutine in NAG.* **Classes:** J1a1 **Usage:** CALL C06FQF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06FRF.** Computes the discrete Fourier transforms of M sequences, each containing N complex data values. This routine is designed to be particularly efficient on vector processors. *Proprietary Fortran subroutine in NAG.* **Classes:** J1a2 **Usage:** CALL C06FRF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06GBF.** Forms the complex conjugate of a Hermitian sequence of N data values. *Proprietary Fortran subroutine in NAG.* **Classes:** A4a, A7 **Usage:** CALL C06GBF (X, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06GCF.** Forms the complex conjugate of a sequence of N data values. *Proprietary Fortran subroutine in NAG.* **Classes:** A4a, A7 **Usage:** CALL C06GCF (Y, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06GQF.** Forms the complex conjugates of M Hermitian sequences, each containing N data values. *Proprietary Fortran subroutine in NAG.* **Classes:** A4a, A7, J1a2 **Usage:** CALL C06GQF (M, N, X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06GSF.** Takes M Hermitian sequences, each containing N data values, and forms the real and imaginary parts of the M corresponding complex sequences. *Proprietary Fortran subroutine in NAG.* **Classes:** A4a, A7, J1a2 **Usage:** CALL C06GSF (M, N, X, U, V, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C06LAF.** Estimates values of the inverse Laplace transform of a given function using a Fourier series approximation. Real and imaginary parts of the function, and a bound on the exponential order of the inverse, are required. *Proprietary Fortran subroutine in NAG.* **Classes:** J3 **Usage:** CALL C06LAF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**C10A3J.** Bessel function $J_{n+\alpha}(x)$, x and $n + \alpha$ non-negative. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10a3 **Usage:** CALL C10A3J(X, ALPHA, NBES, BESJ, KE) **Precision:** Double **Availability:** PC.

**C10AHD.** Complex Bessel function $H_1$, $H_2$ (real order). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10a3 **Usage:** CALL C10AHD(*10 parameters*) **Precision:** Double **Availability:** PC.

**C10AJ0.** Bessel function $J_0(x)$. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10a1 **Usage:** R = C10AJ0(X) **Precision:** Single **Availability:** PC.

**C10AJ1.** Bessel function $J_1(x)$. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10a1 **Usage:** R = C10AJ1(X) **Precision:** Single **Availability:** PC.

**C10AJD.** Complex Bessel function J (real order). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10a3 **Usage:** CALL C10AJD(*9 parameters*) **Precision:** Double **Availability:** PC.

**C10AY0.** Bessel function $Y_0$ (and $J_0$). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10a1 **Usage:** CALL C10AY0(ARG, RESULT, JINT, KE) **Precision:** Double **Availability:** PC.

**C10AY1.** Bessel function $Y_1$ (and $J_1$). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10a1 **Usage:** CALL C10AY1(ARG, RESULT, JINT, KE) **Precision:** Double **Availability:** PC.

**C10AYD.** Complex Bessel function Y (real order). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10a3 **Usage:** CALL C10AYD(*11 parameters*) **Precision:** Double **Availability:** PC.

**C10B3.** Bessel functions $I_{n+\alpha}(x)$ for non-negative x and order $n + \alpha$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b3 **Usage:** CALL C10B3(X, ALPHA, NB, IZE, B, KE) **Precision:** Single **Availability:** PC.

**C10B3K.** Modified Bessel functions of the second kind, $K_{n+\alpha}(x)$, for non-negative x and order $n + \alpha$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b3 **Usage:** CALL C10B3K(X, ALPHA, NB, IZE, BK, KE) **Precision:** Single **Availability:** PC.

**C10BI0.** Bessel function $I_0(x)$. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10b1 **Usage:** R = C10BI0(X, JINT, KE) **Precision:** Single **Availability:** PC.

**C10BI1.** Bessel function $I_1(x)$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b1 **Usage:** CALL C10BI1(ARG, RESULT, JINT) **Precision:** Single **Availability:** PC.

**C10BID.** Complex Bessel function I (real order). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b4 **Usage:** CALL C10BID(*9 parameters*) **Precision:** Double **Availability:** PC.

**C10BK0.** Bessel function $K_0(x)$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b1 **Usage:** CALL C10BK0(ARG, RESULT, JINT) **Precision:** Single **Availability:** PC.

**C10BK1.** Bessel function $K_1(x)$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b1 **Usage:** CALL C10BK1(ARG, RESULT, JINT) **Precision:** Single **Availability:** PC.

**C10BKD.** Complex Bessel function K (real order). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10b4 **Usage:** CALL C10BKD(*9 parameters*) **Precision:** Double **Availability:** PC.

**C10DA.** Airy function Ai(x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10d **Usage:** R = C10DA(X) **Precision:** Single **Availability:** PC.

**C10DAD.** Complex Airy function Ai(z) . *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10d **Usage:** CALL C10DAD(*8 parameters*) **Precision:** Double **Availability:** PC.

**C10DAE.** Exponentially scaled Airy function Ai(x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10d **Usage:** R = C10DAE(X) **Precision:** Single **Availability:** PC.

**C10DAM.** Airy modulus and phase. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10d **Usage:** CALL C10DAM(X, AMPL, THETA, KE) **Precision:** Single **Availability:** PC.

**C10DB.** Airy function Bi(x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10d **Usage:** R = C10DB(X, KE) **Precision:** Single **Availability:** PC.

**C10DBD.** Complex Airy function Bi(z). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C10d **Usage:** CALL C10DBD(ZR, ZI, ID, KODE, BIR, BII, KE) **Precision:** Double **Availability:** PC.

**C10DBE.** Exponentially scaled Airy function Bi(x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C10d **Usage:** R = C10DBE(X) **Precision:** Single **Availability:** PC.

**C14RC.** Carlson's incomplete elliptic integral RC(x,y). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C14 **Usage:** R = C14RC(X, Y, KE) **Precision:** Single **Availability:** PC.

**C14RD.** Carlson's incomplete elliptic integral RD(x,y,z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C14 **Usage:** R = C14RD(X, Y, Z, KE) **Precision:** Single **Availability:** PC.

**C14RF.** Carlson's incomplete elliptic integral RF(x,y,z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C14 **Usage:** R = C14RF(X, Y, Z, KE) **Precision:** Single **Availability:** PC.

**C14RJ.** Carlson's incomplete elliptic integral RJ(x,y,z,p). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C14 **Usage:** R = C14RJ(X, Y, Z, P, KE) **Precision:** Single **Availability:** PC.

**C2CCUB.** Cube root of a real argument. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C2 **Usage:** R = C2CCUB(X) **Precision:** Single **Availability:** PC.

**C2CSQD.** Square root of a complex argument. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C2 **Usage:** CALL C2CSQD(X1R, X1I, X2R, X2I) **Precision:** Double **Availability:** PC.

**C2CUB.** Cube root of a real argument. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C2 **Usage:** R = C2CUB(X) **Precision:** Single **Availability:** PC.

**C4ACCO.** arccos(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACCO(Z) **Precision:** Single **Availability:** PC.

**C4ACCT.** cotan(x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACCT(X, KE) **Precision:** Single **Availability:** PC.

**C4ACSN.** arcsin(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACSN(Z) **Precision:** Single **Availability:** PC.

**C4ACT.** cotan(x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACT(X, KE) **Precision:** Single **Availability:** PC.

**C4ACT2.** arctan(z1/z2). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACT2(CSN, CCS, KE) **Precision:** Single **Availability:** PC.

**C4ACTI.** arctan(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACTI(Z, KE)

**Precision:** Single **Availability:** PC.

**C4ACTN.** tan(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4a **Usage:** R = C4ACTN(Z, KE) **Precision:** Single **Availability:** PC.

**C4BALR.** ln(1+x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4b **Usage:** R = C4BALR(X, KE) **Precision:** Single **Availability:** PC.

**C4BCLD.** ln(z). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C4b **Usage:** CALL C4BCLD(X1R, X1I, X2R, X2I, KE) **Precision:** Double **Availability:** PC.

**C4BCLR.** ln(1+x). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4b **Usage:** R = C4BCLR(X, KE) **Precision:** Single **Availability:** PC.

**C4BCXD.** exp(z). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C4b **Usage:** CALL C4BCXD(X1R, X1I, X2R, X2I) **Precision:** Double **Availability:** PC.

**C4BCXP.** (exp(z)-1)/z. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4b **Usage:** R = C4BCXP(X) **Precision:** Single **Availability:** PC.

**C4BEXP.** (exp(x)-1)/x. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4b **Usage:** R = C4BEXP(X) **Precision:** Single **Availability:** PC.

**C4CACH.** arccosh(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4c **Usage:** R = C4CACH(Z) **Precision:** Single **Availability:** PC.

**C4CASH.** arcsinh(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4c **Usage:** R = C4CASH(Z) **Precision:** Single **Availability:** PC.

**C4CATH.** arctanh(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4c **Usage:** R = C4CATH(Z, KE) **Precision:** Single **Availability:** PC.

**C4CCH.** cosh(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4c **Usage:** R = C4CCH(Z, KE) **Precision:** Single **Availability:** PC.

**C4CSH.** sinh(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4c **Usage:** R = C4CSH(Z, KE) **Precision:** Single **Availability:** PC.

**C4CSHD.** Complex sinh. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C4c **Usage:** CALL C4CSHD(ZR, ZI, CSHR, CSHI, CCHR, CCHI) **Precision:** Double **Availability:** PC.

**C4CTH.** tanh(z). *Proprietary Fortran function in the Scientific Desk.* **Classes:** C4c **Usage:** R = C4CTH(Z, KE) **Precision:** Single **Availability:** PC.

**C5.** Computes various logarithmic and exponential integrals. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C5 **Usage:** CALL C5(*11 parameters*) **Precision:** Double **Availability:** PC.

**C7ACLG.** Complex log of the gamma function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7a **Usage:** R = C7ACLG(Z, KE) **Precision:** Single **Availability:** PC.

**C7AG.** $\Gamma(x)$. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7a **Usage:** R = C7AG(X) **Precision:** Single (Double: C7AGD) **Availability:** PC.

**C7AGD.** *Double precision version of C7AG.*

**C7ALG.** $\ln(\Gamma(x))$. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7a **Usage:** R = C7ALG(X, KE) **Precision:** Single (Double: C7ALGD) **Availability:** PC.

**C7ALGD.** *Double precision version of C7ALG.*

**C7BCLB.** Log of the complex beta function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7b **Usage:** R = C7BCLB(A, B, KE) **Precision:** Single **Availability:** PC.

**C7BCPS.** Psi function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7c **Usage:** R = C7BCPS(X, KE) **Precision:** Single **Availability:** PC.

**C7BLB.** Log of the complete beta function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7b **Usage:** R = C7BLB(A, B, KE) **Precision:** Single **Availability:** PC.

**C7BPSD.** *Double precision version of C7BPSI.*

**C7BPSI.** Psi function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7c **Usage:** R = C7BPSI(X, KE) **Precision:** Single (Double: C7BPSD) **Availability:** PC.

**C7BTA.** Complete beta function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7b **Usage:** R = C7BTA(A, B, KE) **Precision:** Single **Availability:** PC.

**C7EG.** Incomplete gamma function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7e **Usage:** R = C7EG(A, X, KE) **Precision:** Single **Availability:** PC.

**C7EGC.** Complementary incomplete gamma function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C7e **Usage:** R = C7EGC(A, X, KE) **Precision:** Single **Availability:** PC.

**C8AE.** Error function. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C8a **Usage:** CALL C8AE(X, ERF, ERFC, DER, KE) **Precision:** Single (Double: C8AED) **Availability:** PC.

**C8AED.** *Double precision version of C8AE.*

**C8AEI.** Inverse of the error function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C8a **Usage:** R = C8AEI(P) **Precision:** Single **Availability:** PC.

**C8CD.** Dawson's integral. *Proprietary Fortran function in the Scientific Desk.* **Classes:** C8c **Usage:** R = C8CD(X, KE) **Precision:** Single (Double: C8CDD) **Availability:** PC.

**C8CDD.** *Double precision version of C8CD.*

**CABS.** Absolute value of a complex number. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** A4a **Usage:** C = CABS(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CACOS.** arccos(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CACOS(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CACOS.** arccos(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CACOS(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CACOSH.** arccosh(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** C = CACOSH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CACOSH.** arccosh(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CACOSH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CADD.** Add a scalar to each component of a vector, x = x + a, all complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL CADD (N, CA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CANCORR.** Performs canonical correlation and tests correlation hypotheses using an F approximation. Both standardized and unstandardized canonical coefficients and correlations between canonical variable and the original variables are produced. Options: canonical redundancy analysis, partial canonical correlation, weights, output data sets of scores on each canonical variable and canonical coefficients. *Proprietary stand-alone program using SAS command language.* **Classes:** L13c **Precision:** Single **Availability:** CAMVAX.

**CANDISC.** Performs a canonical discriminant analysis, computes Mahalanobis distances, and does both univariate and multivariate one-way analyses of variance. Tests zero correlations using an F approximation. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L12 **Precision:** Single **Availability:** CAMVAX.

**CARG.** Argument of a complex number. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** A4a **Usage:** R = CARG(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CARG.** Argument of a complex number. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** A4a **Usage:** C = CARG(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CASIN.** arcsin(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CASIN(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CASIN.** arcsin(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CASIN(ZINP) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CASINH.** arcsinh(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** C = CASINH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CASINH.** arcsinh(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CASINH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CATAN.** arctan(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CATAN(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CATAN.** arctan(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CATAN(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CATAN2.** arctan(z1/z2). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CATAN2(Z1, Z2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CATAN2.** arctan(z1/z2). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CATAN2(CSN, CCS) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CATANH.** arctanh(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** C = CATANH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CATANH.** arctanh(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CATANH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CATMOD.** Analyzes two-dimensional contingency tables by fitting linear models to functions of response frequencies using a maximum-likelihood estimation of parameters for log-linear models and the analysis of generalized logits, or using a weighted-least-squares estimation of parameters for general linear models. Options: weights, parameter testing. *Proprietary stand-alone program using SAS command language.* **Classes:** L9b, L9c **Precision:** Single **Availability:** CAMVAX.

**CAUCDF.** Computes the cumulative distribution function value for the Cauchy distribution with median 0 and 75% point 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1c **Usage:** CALL CAUCDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CAUCHY PROBABILITY PLOT.** Generates a probability plot for the Cauchy distribution with median 0 and 75% point 1. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2c **Usage:** CAUCHY PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CAUPDF.** Computes the probability density function value for the Cauchy distribution with median 0 and 75% point 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1c **Usage:** CALL CAUPDF(X, PDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CAUPLT.** Generates a Cauchy probability plot with median 0 and 75% point 1. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2c **Usage:** CALL CAUPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CAUPPF.** Computes the percent point function value for the Cauchy distribution with median 0 and 75% point 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2c **Usage:** CALL CAUPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CAURAN.** Generates a random sample of size N from the Cauchy distribution with median 0 and 75% point 1. *Fortran subroutine in DATAPAC.* **Classes:** L6a3 **Usage:** CALL CAURAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CAUSF.** Computes the sparsity function value for the Cauchy distribution with median 0 and 75% point 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2c **Usage:** CALL CAUSF(P, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CAXPY.** Compute a scalar times a vector plus a vector, y = ax + y, all complex. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a7 **Usage:** CALL CAXPY(N, CA, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CAXPY.** Compute a scalar times a vector plus a vector, y = ax + y, all complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a7 **Usage:** CALL CAXPY (N, CA, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CBABK2.** Forms eigenvectors of complex general matrix from eigenvectors of matrix output from CBAL. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL CBABK2(*8 parameters*) **Also see:** CBAL **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CBAL.** Balances a complex general matrix and isolates eigenvalues whenever possible. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1a **Usage:** CALL CBAL(NM, N, AR, AI, LOW, IGH, SCALE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CBETA.** Beta(a,b), for complex a,b (with positive real parts). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7b **Usage:** C = CBETA(Z1, Z2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CBETA.** Complex complete beta function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7b **Usage:** C = CBETA(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CBINS.** Sequence of modified Bessel functions $I_n(z)$. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10b2 **Usage:** CALL CBINS (Z, N, CBS) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CBJNS.** Sequence of Bessel functions $J_n(z)$. *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C10a2 **Usage:** CALL CBJNS (Z, N, CBS) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CBLKTR.** Solves certain complex block tridiagonal systems of linear equations arising from the discretization of separable elliptic partial differential equations. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b4b **Usage:** CALL CBLKTR(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CBNRHO.** Estimation of the bivariate normal correlation coefficient using a contingency table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L9b, L4b4 **Usage:** CALL CBNRHO (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CBRT.** Cube root of a real argument. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C2 **Usage:** R = CBRT(X) **Precision:** Single (Double: DCBRT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CBRT.** Cube root of a real argument. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C2 **Usage:** C = CBRT(X) **Precision:** Single (Double: DCBRT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCBCB.** Copy a complex band matrix stored in complex band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b8 **Usage:** CALL CCBCB (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCBCG.** Convert a complex matrix in band storage mode to a complex matrix in full storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CCBCG (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCBRT.** Cube root of a complex argument. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C2 **Usage:** C = CCBRT(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CCBRT.** Cube root of a complex argument. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C2 **Usage:** C = CCBRT(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCF.** Compute the sample cross-correlation function of two stationary time series. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10b2a **Usage:** CALL CCF (*14 parameters*) **Precision:** Single (Double:

DCCF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCF.** Computes and graphs cross-correlations between two time series. *Command in MINITAB proprietary interactive system.* **Classes:** L10b2a **Usage:** CCF [with up to K lags] between series in C and C **Precision:** Single **Availability:** 855NOS.

**CCF.** Compute and print a two-part cross correlation analysis of a pair of series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b2a **Usage:** CALL CCF (Y1, Y2, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CCFF.** Compute and print a two-part cross correlation analysis of a pair of series; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b2a **Usage:** CALL CCFF (*5 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CCFFS.** Compute and optionally print a two-part cross correlation analysis of a multivariate series using user-supplied control values; use FFT for computations; return cross covariance function. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b2a **Usage:** CALL CCFFS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CCFM.** Compute and print a two-part cross correlation analysis of a pair of series with missing observations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b2a **Usage:** CALL CCFM (Y1, YMISS1, Y2, YMISS2, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CCFMS.** Compute and optionally print a two-part cross correlation analysis of a multivariate series with missing observations using user- supplied control values; return cross covariance function. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b2a **Usage:** CALL CCFMS (*15 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CCFS.** Compute and optionally print a two-part cross correlation analysis of a multivariate series using user-supplied control values; return cross covariance function. *Portable Fortran subroutine in STARPAC.* **Classes:** L10b2a **Usage:** CALL CCFS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CCGCB.** Convert a complex matrix in full storage mode to a matrix in complex band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CCGCB (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCGCG.** Copy a complex general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b8 **Usage:** CALL CCGCG (N, A, LDA, B, LDB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCHDC.** Compute Cholesky decomposition of complex positive definite matrix with optional pivoting. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CCHDC(A, LDA, P, WORK, JPVT, JOB, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CCHDD.** Downdates Cholesky factorization of positive definite complex matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D7b **Usage:** CALL CCHDD(*12 parameters*) **Also see:** CCHDC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CCHEX.** Updates Cholesky factorization of positive definite complex matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D7b **Usage:** CALL CCHEX(*11 parameters*) **Also see:** CCHDC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CCHUD.** Updates Cholesky factorization of positive definite matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D7b **Usage:** CALL CCHUD(*11 parameters*) **Also see:** CCHDC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CCOPY.** Copy a vector X to a vector Y, both complex. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a5 **Usage:** CALL CCOPY(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CCOPY.** Copy a vector X to a vector Y, both complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a5 **Usage:** CALL CCOPY (N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCOS.** cos(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CCOS(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CCOS.** cos(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CCOS(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCOSH.** cosh(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** C = CCOSH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CCOSH.** cosh(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CCOSH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CCOT.** cot(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CCOT(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CCOT.** cotan(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CCOT(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDADD.** Sum of double precision complex numbers. Each is represented by a double precision array of two elements. *Proprietary Fortran subroutine in PORT.* **Classes:** A4 **Usage:** CALL CDADD (A, B, C) **Precision:** Double **Availability:** 855NOS, 205.

**CDCDOT.** Computes complex precision dot product and adds a scalar. Uses double precision accumulation. *Portable Fortran function in CMLIB (XBLAS sublibrary).* **Classes:** D1a4 **Usage:** C = CDCDOT(N, CB, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CDDIV.** Quotient of double precision complex numbers. Each is represented by a double precision array of two elements. *Proprietary Fortran subroutine in PORT.* **Classes:** A4 **Usage:** CALL CDDIV (A, B, C) **Precision:** Double **Availability:** 855NOS, 205.

**CDEXP.** exp(z) for complex double precision z. *Proprietary Fortran subroutine in PORT.* **Classes:** C4b **Usage:** CALL CDEXP (X, EXP) **Precision:** Double **Availability:** 855NOS, 205.

**CDF2P.** Print a plot of two sample cumulative distribution functions. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3b2, L4b1a1 **Usage:** CALL CDF2P (NOBS1, NOBS2, X) **Precision:** Single (Double: DCDF2P) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDFP.** Print a sample cumulative distribution function (CDF), a theoretical CDF, and confidence band information. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1a1 **Usage:** CALL CDFP (*6 parameters*) **Precision:** Single (Double: DCDFP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDGRD.** Approximate the gradient using central differences. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4f **Usage:** CALL CDGRD (*6 parameters*) **Precision:** Single (Double: DCDGRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDIST.** Compute a matrix of dissimilarities (or similarities) between the columns (or rows) of a matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L14d **Usage:** CALL CDIST (*11 parameters*) **Precision:** Single (Double: DCDIST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDLOG.** ln(z) for complex double precision z. *Proprietary Fortran subroutine in PORT.* **Classes:** C4b **Usage:** CALL CDLOG (X, LOG) **Precision:** Double **Availability:** 855NOS, 205.

**CDMUL.** Product of double precision complex numbers. Each is represented by a double precision array of two elements. *Proprietary Fortran subroutine in PORT.* **Classes:** A4 **Usage:** CALL CDMUL (A, B, C) **Precision:** Double **Availability:** 855NOS, 205.

**CDOTC.** Compute complex dot product using conjugated vector components. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a4 **Usage:** C = CDOTC(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CDOTC.** Compute the complex conjugate dot product, conjg(x)·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CDOTC(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDOTU.** Compute complex dot product using unconjugated vector components. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a4 **Usage:** C = CDOTU(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CDOTU.** Compute the complex dot product x·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CDOTU(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CDRIV1.** Numerical integration of complex initial value problems for ordinary differential equations, Gear stiff formulas. Easy to use. *Portable Fortran subroutine in CMLIB (CDRIV sublibrary).* **Classes:** I1a2, I1a1b **Usage:** CALL CDRIV1(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CDRIV2.** Numerical integration of complex initial value problems for ordinary differential equations, Gear stiff and Adams formulas, root finding. *Portable Fortran subroutine in CMLIB (CDRIV sublibrary).* **Classes:** I1a2, I1a1b **Usage:** CALL CDRIV2(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CDRIV3.** Numerical integration of complex initial value problems for ODEs, Gear and Adams formulas, implicit equations, sparse Jacobians, root finding. *Portable Fortran subroutine in CMLIB (CDRIV sublibrary).* **Classes:** I1a2, I1a1b **Usage:** CALL CDRIV3(*27 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CDSUB.** Subtract double precision complex numbers. Each is represented by a double precision array of two elements. *Proprietary Fortran subroutine in PORT.* **Classes:** A4 **Usage:** CALL CDSUB (A, B, C) **Precision:** Double **Availability:** 855NOS, 205.

**CEIL.** Smallest integer greater than or equal to x. Input and output are real. *Proprietary Fortran function in PORT.* **Classes:** C1 **Usage:** R = CEIL(X) **Precision:** Single (Double: DCEIL) **Availability:** 855NOS, 205.

**CENTER.** Centers data to mean 0, standard deviation 1. Optionally can select location and scale or minimum and maximum. *Command in MINITAB proprietary interactive system.* **Classes:** L2a **Usage:** CENTer C, ..., C put into C, ..., C [; subcommands LOCATION [K, ..., K]; SCALE [K, ..., K]; MINMAX [K, K].] **Precision:** Single **Availability:** 855NOS.

**CENTER.** Subtract the series mean from each observation of a series; return the centered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L2a **Usage:** CALL CENTER (Y, N, YC) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CERFE.** Complex scaled complementary error function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8a **Usage:** C = CERFE(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CEXP.** exp(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** C = CEXP(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CEXP.** exp(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** C = CEXP(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CEXPRL.** (exp(z)-1)/z. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** C = CEXPRL(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CEXPRL.** (exp(z)-1)/z. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** C = CEXPRL(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CFFT2D.** Two dimensional fast Fourier transform (FFT), forward or reverse, of a complex n-by-m matrix F. *Portable Fortran software in NMS library.* **Classes:** J1b **Usage:** CALL CFFT2D(N, F, LDF, FORWD) **Precision:** Single (Double: DCFFT2D) **Availability:** PC.

**CFFTB.** Backward complex discrete (fast) Fourier transform. Performs Fourier synthesis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a2 **Usage:** CALL CFFTB(N, C, WSAVE) **Also see:** CFFTF CFFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CFFTB.** Backward complex FFT of complex C(N). Returns result in C. *Portable Fortran software in NMS library.* **Classes:** J1a2 **Usage:** CALL CFFTB(N, C, W) **Also see:** CFFTI CFFTF **Precision:** Single (Double: DCFFTB) **Availability:** PC.

**CFFTF.** Forward complex discrete (fast) Fourier transform. Performs Fourier analysis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a2 **Usage:** CALL CFFTF(N, C, WSAVE) **Also see:** CFFTB CFFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CFFTF.** Forward complex FFT of complex C(N). Returns result in C. *Portable Fortran software in NMS library.* **Classes:** J1a2 **Usage:** CALL CFFTF(N, C, W) **Also see:** CFFTI CFFTB **Precision:** Single (Double: DCFFTF) **Availability:** PC.

**CFFTI.** Initialize array WSAVE for forward/back FFT. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Usage:** CALL CFFTI(N, WSAVE) **Also see:** CFFTF CFFTB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CG.** Computes the eigenvalues and, optionally, the eigenvectors of a complex general matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a4 **Usage:** CALL CG(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CGAMMA.** $\Gamma(z)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** C = CGAMMA(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CGAMMA.** Complex gamma function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** C = CGAMMA(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CGAMR.** $1/\Gamma(z)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** C = CGAMR(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CGAMR.** $1/\Gamma(z)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** C = CGAMR(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CGBCO.** Compute LU factorization of complex band matrix and estimate its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c2 **Usage:** CALL CGBCO(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGBDI.** Compute determinant of complex band matrix from its LU factors. (No provision for computing inverse directly.) *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D3c2 **Usage:** CALL CGBDI(ABD, LDA, N, ML, MU, IPVT, DET) **Also see:** CGBCO CGBFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGBFA.** Compute LU factorization of general complex band matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c2 **Usage:** CALL CGBFA(ABD, LDA, N, ML, MU, IPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGBSL.** Uses LU factorization of complex band matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c2 **Usage:** CALL CGBSL(ABD, LDA, N, ML, MU, IPVT, B, JOB) **Also see:** CGBCO CGBFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGECO.** Compute LU factorization of general complex matrix and estimate its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c1 **Usage:** CALL CGECO(A, LDA, N, IPVT, RCOND, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGEDI.** Compute determinant and/or inverse of general complex matrix from its LU factors. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c1, D3c1 **Usage:** CALL CGEDI(A, LDA, N, IPVT, DET, WORK, JOB) **Also see:** CGECO CGEFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGEEV.** Computes the eigenvalues and, optionally, the eigenvectors of a general complex matrix. *Portable Fortran subroutine in CMLIB (LICEPAK sublibrary).* **Classes:** D4a4 **Usage:** CALL CGEEV(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CGEFA.** Compute LU factorization of general complex matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c1 **Usage:** CALL CGEFA(A, LDA, N, IPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGEFS.** Factors and solves a general complex system of linear equations. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c1 **Usage:** CALL CGEFS(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGEIR.** Factors and solves a general complex system of linear equations and provides estimate of accuracy of the solution. (Needs n-by-n extra storage.) *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c1 **Usage:** CALL CGEIR(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGEL.** To solve a system of linear equations, i.e., the matrix equation AX = B, where A is a full matrix. *Fortran/meta subroutine in MAGEV.* **Classes:** D2c1 **Usage:** CALL CGEL(*11 parameters*) **Precision:** Single **Availability:** 205 (vectorized)

**CGESL.** Use LU factorization of general complex matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c1 **Usage:** CALL CGESL(A, LDA, N, IPVT, B, JOB) **Also see:** CGECO CGEFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CGTSL.** Solves systems with general complex tridiagonal matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c2a **Usage:** CALL CGTSL(N, C, D, E, B, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CH.** Computes the eigenvalues and, optionally, eigenvectors of a complex Hermitian matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a3 **Usage:** CALL CH(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHART.** Produces line printer vertical and horizontal bar charts (histograms), X-Y-Z block charts, pie charts, star charts, frequency and cumulative frequency plots. *Proprietary stand-alone program using SAS command language.* **Classes:** L3a1, L3a2, L3a4, L3a5, L3c, L3e1, L3e2, L3e4 **Precision:** Single **Availability:** CAMVAX.

**CHBCB.** Copy a complex Hermitian band matrix stored in band Hermitian storage mode to a complex band matrix stored in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CHBCB (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHFAC.** Compute an upper triangular factorization of a real symmetric nonnegative definite matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** D2b1b **Usage:** CALL CHFAC (*7 parameters*) **Precision:** Single (Double: DCHFAC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHFCG.** Extend a complex Hermitian matrix defined in its upper triangle to its lower triangle. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CHFCG (N, A, LDA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHFDV.** Evaluates a cubic polynomial and its first derivative at an array of points. The polynomial must be given in Hermite form. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E3a1, E3a2 **Usage:** CALL CHFDV(*12 parameters*) **Precision:** Single (Double: DCHFDV) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CHFEV.** Evaluates a cubic polynomial given in Hermite form at an array of points. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E3a1 **Usage:** CALL CHFEV(*11 parameters*) **Precision:** Single (Double: DCHFEV) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CHGRD.** Check a user-supplied gradient of a function. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4c **Usage:** CALL CHGRD (FCN, GRAD, N, X, INFO) **Precision:** Single (Double: DCHGRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHHES.** Check a user-supplied Hessian of an analytic function. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4c **Usage:** CALL CHHES (*6 parameters*) **Precision:** Single (Double: DCHHES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHI.** Hyperbolic cosine integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = CHI(X) **Precision:** Single (Double: DCHI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHI-SQUARED PPCC PLOT.** Generates a probability plot correlation coefficient plot for the chi-squared distribution (plot of probability plot correlation coefficient vs. degrees of freedom parameter $\nu$ for $\nu$ ranging from 1 to 100 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3c **Usage:** CHI-SQUARED PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CHI-SQUARED PROBABILITY PLOT.** Generates a probability plot for the chi-squared distribution with integer degrees of freedom $\nu$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2c **Usage:** CHI-SQUARED PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CHICO.** Computes factorization of complex Hermitian indefinite matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CHICO(A, LDA, N, KPVT, RCOND, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHIDF.** Chi-squared distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1c, C7e **Usage:** R = CHIDF(CHSQ, DF) **Precision:** Single (Double: DCHIDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHIDI.** Uses factorization of complex Hermitian indefinite matrix to compute its inertia, determinant, and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a, D3d1a **Usage:** CALL CHIDI(*8 parameters*) **Also see:** CHICO CHIFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHIEV.** Computes the eigenvalues and, optionally, the eigenvectors of a complex Hermitian matrix. *Portable Fortran subroutine in CMLIB (LICEPAK sublibrary).* **Classes:** D4a3 **Usage:** CALL CHIEV(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHIFA.** Computes factorization of complex Hermitian indefinite matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CHIFA(A, LDA, N, KPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHIGF.** Perform a chi-squared goodness of fit test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1c **Usage:** CALL CHIGF (*14 parameters*) **Precision:** Single (Double: DCHIGF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHIIN.** Inverse of the chi-squared distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a2c, C7e **Usage:** R = CHIIN(P, DF) **Precision:** Single (Double: DCHIIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHISL.** Uses factorization of complex Hermitian indefinite matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CHISL(A, LDA, N, KPVT, B) **Also see:** CHICO CHIFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHISQUARE.** Performs chi-square test for association (non-independence) on a two-way table and prints standard results. *Command in MINITAB proprietary interactive system.* **Classes:** L9b **Usage:** CHISquare test on table stored in columns C, ..., C **Precision:** Single **Availability:** 855NOS.

**CHJAC.** Check a user-supplied Jacobian of a system of equations with M functions in N unknowns. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4c **Usage:** CALL CHJAC (*7 parameters*) **Precision:** Single (Double: DCHJAC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CHKDER.** Checks gradients of M nonlinear functions in N variables evaluated at a point X for consistency with the functions themselves. A companion subprogram to subprograms SNLS1E and SNLS1. This subprogram can also be used to check the coding of the Jacobian matrix calculation. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Usage:** CALL CHKDER(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CHPCO.** Computes factorization of complex Hermitian indefinite matrix stored in packed form and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CHPCO(AP, N, KPVT, RCOND, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHPDI.** Uses factorization of complex Hermitian indefinite matrix stored in packed form to compute its inertia, determinant, and inverse. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a, D3d1a **Usage:** CALL CHPDI(*7 parameters*) **Also see:** CHPCO CHPFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHPFA.** Computes factorization of complex Hermitian indefinite matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CHPFA(AP, N, KPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHPSL.** Uses factorization of complex Hermitian indefinite matrix stored in packed form to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CHPSL(AP, N, KPVT, B)

**Also see:** CHPCO CHPFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CHSCDF.** Computes the cumulative distribution function value for the chi-squared distribution with degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L5a1c **Usage:** CALL CHSCDF(X, NU, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CHSPLT.** Generates a chi-squared probability plot with integer degrees of freedom parameter value NU. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2c **Usage:** CALL CHSPLT(X, N, NU) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CHSPPF.** Computes the percent point function value for the chi-squared distribution with integer degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L5a2c **Usage:** CALL CHSPPF(P, NU, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CHSRAN.** Generates a random sample of size N from the chi-squared distribution with integer degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L6a3 **Usage:** CALL CHSRAN(N, NU, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CHU.** Confluent hypergeometric function, U(a,b,x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C11 **Usage:** R = CHU(A, B, X) **Precision:** Single (Double: DCHU) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CI.** Cosine integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = CI(X) **Precision:** Single (Double: DCI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CIDMS.** Compute a confidence interval on a variance component estimated as proportional to the difference in two mean squares in a balanced complete experimental design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7d1 **Usage:** CALL CIDMS (*8 parameters*) **Precision:** Single (Double: DCIDMS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CIN.** Evaluate a function closely related to the cosine integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = CIN(X) **Precision:** Single (Double: DCIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CINH.** Evaluate a function closely related to the hyperbolic cosine integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = CINH(X) **Precision:** Single (Double: DCINH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CINVIT.** Computes eigenvectors of a complex upper Hessenberg matrix associated with specified eigenvalues using inverse iteration. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL CINVIT(*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CJYHBS.** Bessel and Struve functions $J_0(z)$, $J_1(z)$, $Y_0(z)$, $Y_1(z)$, $H_0(z)$, $H_1(z)$. *Portable Fortran subroutine in CMLIB (AMOSLIB sublibrary).* **Classes:** C10a2, C10e **Usage:** CALL CJYHBS(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CLBETA.** ln(Beta(a,b)), for complex a,b (with positive real parts). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7b **Usage:** C = CLBETA(Z1, Z2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CLBETA.** Complex logarithm of the complete beta function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7b **Usage:** C = CLBETA(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLIFE.** Life table analysis. *Proprietary Fortran subroutine in IMSL.* **Classes:** L15 **Usage:** CALL CLIFE (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLINK.** Perform a hierarchical cluster analysis given a distance matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L14a1a1 **Usage:** CALL CLINK (*8 parameters*) **Precision:** Single (Double: DCLINK) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLINQ.** Solves a complex system of linear equations. Coefficient matrix must be input as two real matrices. *Proprietary Fortran subroutine in PORT.* **Classes:** D2c1 **Usage:** CALL CLINQ (N, AR, AI, BR, BI, NB, XR, XI) **Precision:** Single (Double: DCLINQ) **Availability:** 855NOS, 205.

**CLNGAM.** $\ln(\Gamma(z))$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** C = CLNGAM(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CLNGAM.** $\ln(\Gamma(z))$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** C = CLNGAM(ZIN) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLNREL.** ln(1+z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** C = CLNREL(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CLNREL.** ln(1+z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** C = CLNREL(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLOG.** ln(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** C = CLOG(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CLOG.** ln(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** C = CLOG(Z) **Pre-**

**cision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLOG10.** log(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** C = CLOG10(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CLOG10.** log(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** C = CLOG10(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CLST2.** Finds the least squares solution of a complex linear algebraic system of equations AX=B. B may be a matrix. Uses real arithmetic. *Proprietary Fortran subroutine in PORT.* **Classes:** D9a1 **Usage:** CALL CLST2 (*11 parameters*) **Precision:** Single (Double: DCLST2) **Availability:** 855NOS, 205.

**CLUSTER.** Hierarchically clusters observations by one of eleven procedures (standard linkage methods, density linkage (including kth-nearest-neighbor and two-stage), and maximum-likelihood for mixtures of spherical multivariate normal distributions). Input data can be either coordinates or distances. Options: trimming input data, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L14a1a1 **Precision:** Single **Availability:** CAMVAX.

**CLUSTER.** Performs agglomerative hierarchical clustering. Options: clustering methods (single link, complete link, between- and within-groups average link, median, centroid, and Ward's methods), six proximity measures, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L14a1a1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**CMGNBN.** Solves certain complex block tridiagonal systems of linear equations arising from Helmholtz or Poisson equations in 2 dimensional Cartesian coordinates. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b4b **Usage:** CALL CMGNBN(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CMXMOV.** To move a matrix from one location to another. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b8 **Usage:** CALL CMXMOV(M, N, A, LA, B, BB) **Precision:** Single **Availability:** 205 (vectorized)

**CNBCO.** Factors a complex band matrix by Gaussian elimination and estimates its condition number. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c2 **Usage:** CALL CNBCO(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CNBDI.** Computes the determinant of a complex band matrix from previously computed factors. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D3c2 **Usage:** CALL CNBDI(ABE, LDA, N, ML, MU, IPVT, DET) **Also see:** CNBCO CNBFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CNBFA.** Factors a non-symmetric complex band matrix by elimination. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c2 **Usage:** CALL CNBFA(ABE, LDA, N, ML, MU, IPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CNBFS.** Factors and solves a general complex band matrix system of linear equations. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c2 **Usage:** CALL CNBFS(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CNBIR.** Factors and solves a general nonsymmetric complex band system of equations and estimates accuracy of the solution. (Requires N*(2*ML+MU) extra storage.) *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c2 **Usage:** CALL CNBIR(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CNBSL.** Solves the nonsymmetric complex band system of equations using factors previously computed. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2c2 **Usage:** CALL CNBSL(ABE, LDA, N, ML, MU, IPVT, B, JOB) **Also see:** CNBCO CNBFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CNCRD.** Calculate and test the significance of the Kendall coefficient of concordance. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b1b **Usage:** CALL CNCRD (*7 parameters*) **Precision:** Single (Double: DCNCRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CNUMB.** Compute cluster membership for a hierarchical cluster tree. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L14d **Usage:** CALL CNUMB (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CNVBDC.** Converts double precision vector into complex vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBDC (N, A, B) **Availability:** 855NOS, 205.

**CNVBDI.** Converts double precision vector into integer vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBDI (N, A, B) **Availability:** 855NOS, 205.

**CNVBDR.** Converts double precision vector into real vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBDR (N, A, B) **Availability:** 855NOS, 205.

**CNVBIC.** Converts integer vector into complex vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBIC (N, A, B) **Availability:** 855NOS, 205.

**CNVBID.** Converts integer vector to double precision vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBID (N, A, B) **Availability:** 855NOS, 205.

**CNVBIR.** Converts integer vector into real vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBIR (N, A, B) **Availability:** 855NOS, 205.

**CNVBRC.** Converts real vector into complex vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBRC (N, A, B) **Availability:** 855NOS, 205.

**CNVBRD.** Converts real vector into double precision vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBRD (N, A, B) **Availability:** 855NOS, 205.

**CNVBRI.** Converts real vector into integer vector (backward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVBRI (N, A, B) **Availability:** 855NOS, 205.

**CNVFDC.** Converts double precision vector into complex vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFDC (N, A, B) **Availability:** 855NOS, 205.

**CNVFDI.** Converts double precision vector into integer vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFDI (N, A, B) **Availability:** 855NOS, 205.

**CNVFDR.** Converts double precision vector into real vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFDR (N, A, B) **Availability:** 855NOS, 205.

**CNVFIC.** Converts integer vector into complex vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFIC (N, A, B) **Availability:** 855NOS, 205.

**CNVFID.** Converts integer vector into double precision vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFID (N, A, B) **Availability:** 855NOS, 205.

**CNVFIR.** Converts integer vector into real vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFIR (N, A, B) **Availability:** 855NOS, 205.

**CNVFRC.** Converts real vector into complex vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFRC (N, A, B) **Availability:** 855NOS, 205.

**CNVFRD.** Converts real vector into double precision vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFRD (N, A, B) **Availability:** 855NOS, 205.

**CNVFRI.** Converts real vector into integer vector (forward loop). *Proprietary Fortran subroutine in PORT.* **Classes:** A6a **Usage:** CALL CNVFRI (N, A, B) **Availability:** 855NOS, 205.

**CODE.** Codes the elements of the input vector X; 1.0 for minimum, 2.0 for next larger, etc. *Fortran subroutine in DATAPAC.* **Classes:** N6a1b **Usage:** CALL CODE(X, N, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**COMBAK.** Forms eigenvectors of complex general matrix from eigenvectors of upper Hessenberg matrix output from COMHES. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL COMBAK(*9 parameters*) **Also see:** COMHES **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COMHES.** Reduces complex general matrix to complex upper Hessenberg form using stabilized elementary similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b2 **Usage:** CALL COMHES(NM, N, LOW, IGH, AR, AI, INT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COMLR.** Computes eigenvalues of a complex upper Hessenberg matrix using the modified LR method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL COMLR(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COMLR2.** Computes eigenvalues and eigenvectors of complex upper Hessenberg matrix using modified LR method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL COMLR2(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COMPLEX DEMODULATION PLOT.** Performs complex demodulation and plots either an amplitude or a phase plot for equi-spaced data with a user-specified demodulation frequency. Option: recover estimate of demodulation frequency. *Command(s) in DATAPLOT interactive system.* **Classes:** L10a3b **Usage:** COMPLEX DEMODULATION <AMPLITUDE/PHASE> PLOT <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**COMQR.** Computes eigenvalues of complex upper Hessenberg matrix using the QR method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL COMQR(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COMQR2.** Computes eigenvalues and eigenvectors of complex upper Hessenberg matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL COMQR2(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CONDESCRIPTIVE.** Computes univariate summary statistics (mean, standard deviation, minimum, maximum) for continuous variables and computes standardized variables. Options: standard error of mean, variance, kurtosis, skewness, range, sum, z-scores, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1a1, L1c1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**CONFIDENCE LIMITS.** Generates confidence limits for the mean, based on normal theory, for a series of confidence levels. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a4n **Usage:** CONFIDENCE LIMITS FOR [THE] MEAN <VARIABLE NAME> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CONST.** Various mathematical and physical constants. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** C19 **Usage:** C = CONST(NAME) **Precision:** Single (Double: DCONST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CONTOUR PLOT.** Generate a contour plot with user specified contour levels. *Command(s) in DATAPLOT interactive system.* **Classes:** Q **Usage:** CONTOUR PLOT <VARIABLE 1> <VARIABLE 2> <VARIABLE 3> <VARIABLE 4> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**CONTROL CHART.** Plots a specified statistic (mean (default), standard deviation, or range) for replicated data as a function of time or replication group number. Output contains computed statistics, a typical value line and upper and lower control lines. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1d, L10a2b **Usage:** [MEAN/STANDARD DEVIATION/RANGE] CONTROL CHART <RESPONSE VARIABLE> <INDEPENDENT VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**COPY.** Copies the contents of the vector X into vector Y. *Fortran subroutine in DATAPAC.* **Classes:** D1a5 **Usage:** CALL COPY(X, N, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CORR.** Computes the sample correlation coefficient between the sets of data in the input vectors X and Y. *Fortran subroutine in DATAPAC.* **Classes:** L1b **Usage:** CALL CORR(X, Y, N, IWRITE, C) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CORR.** Computes correlation coefficients between variables, including Pearson product-moment and weighted product-moment correlations. Can also compute Spearman's rank-order correlation, Kendall's tau-b, and Hoeffding's measurement of dependence. Options: some univariate descriptive statistics, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L1c1b **Precision:** Single **Availability:** CAMVAX.

**CORR.** Compute and print a correlation analysis of a multivariate random sample. *Portable Fortran subroutine in STARPAC.* **Classes:** L4c1a **Usage:** CALL CORR (YM, N, M, IYM, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CORRELATION PLOT.** Plots autocorrelations vs. lag for equi-spaced univariate time series data or plots the cross-correlation coefficient vs. lag for equi-spaced bivariate time series data. *Command(s) in DATAPLOT interactive system.* **Classes:** L10a2a1, L10b2a **Usage:** <AUTO/CROSS-> CORRELATION PLOT <RESPONSE VARIABLE> [<INDEPENDENT VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**CORRELATION.** Calculates the Pearson product moment correlation coefficient between two or more pairs of vectors, handles missing values, and optionally saves results. *Command in MINITAB proprietary interactive system.* **Classes:** L1c1b **Usage:** CORRelation coefficients between data in columns C, ..., C [store in M] **Precision:** Single **Availability:** 855NOS.

**CORRS.** Compute and optionally print a correlation analysis of a multivariate random sample; return variance-covariance matrix. *Portable Fortran subroutine in STARPAC.* **Classes:** L4c1a **Usage:** CALL CORRS (*8 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**CORTB.** Forms eigenvectors of complex general matrix from eigenvectors of upper Hessenberg matrix output from CORTH. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL CORTB(*10 parameters*) **Also see:** CORTH **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CORTH.** Reduces complex general matrix to complex upper Hessenberg using unitary similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b2 **Usage:** CALL CORTH(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CORVC.** Compute the variance-covariance or correlation matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L1c1b **Usage:** CALL CORVC (*17 parameters*) **Precision:** Single (Double: DCORVC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**COS.** cos(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = COS(X) **Precision:** Single (Double: DCOS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COS.** cos(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = COS(X) **Precision:** Single (Double: DCOS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**COSDG.** cos(x), x in degrees. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = COSDG(X) **Precision:** Single (Double: DCOSDG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COSDG.** cos(x), x in degrees. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = COSDG(X) **Precision:** Single (Double: DCOSDG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**COSH.** cosh(x) . *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** R = COSH(X) **Precision:** Single (Double: DCOSH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COSH.** cosh(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = COSH(X) **Precision:** Single (Double: DCOSH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**COSH.** cosh(x). *Proprietary Fortran function in PORT.* **Classes:** C4c **Usage:** R = COSH(X) **Precision:** Single (Double: DCOSH) **Availability:** 855NOS, 205.

**COSQB.** Fast Fourier transform of quarter wave data. Computes a sequence from cosine series representation.

Fourier synthesis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a3 **Usage:** CALL COSQB(N, X, WSAVE) **Also see:** COSQF COSQI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COSQF.** Fast Fourier transform of quarter wave data. Fourier analysis. Computes coefficients in cosine series with odd wave numbers. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a3 **Usage:** CALL COSQF(N, X, WSAVE) **Also see:** COSQB COSQI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COSQI.** Initialize array WSAVE for SUBROUTINES COSQF and COSQB. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Usage:** CALL COSQI(N, WSAVE) **Also see:** COSQF COSQB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COST.** Discrete fast cosine transform of even sequence $X_i$. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a3 **Usage:** CALL COST(N, X, WSAVE) **Also see:** COSTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COSTI.** Initialize array WSAVE for SUBROUTINE COST. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Usage:** CALL COSTI(N, WSAVE) **Also see:** COST **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COT.** cot(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = COT(X) **Precision:** Single (Double: DCOT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**COT.** cotan(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = COT(X) **Precision:** Single (Double: DCOT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**COULFG.** A subroutine to compute the regular and irregular Coulomb wave functions F and G, and their derivatives dF/dx and dG/dx, for real values of the Sommerfeld parameter $\eta$ and the angular momentum L. A single call to COULFG computes the functions for fixed positive x, fixed real $\eta$, and a range of L-values in integer steps. *Portable Fortran software in COULOMB library.* **Classes:** C12, C10a3 **Usage:** CALL COULFG(*11 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**COUNT.** Computes the number of observations between XMIN and XMAX (inclusively) in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1d **Usage:** CALL COUNT(X, N, XMIN, XMAX, IWRITE, XCOUNT) **Precision:** Single **Availability:** 855NOS, 840NOS.

**CPBCO.** Uses Cholesky algorithm to compute factorization of complex positive definite band matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d2 **Usage:** CALL CPBCO(ABD, LDA, N, M, RCOND, Z, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPBDI.** Uses factorization of complex positive definite band matrix to compute determinant. (No provision for computing inverse.) *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D3d2 **Usage:** CALL CPBDI(ABD, LDA, N, M, DET) **Also see:** CPBCO CPBFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPBFA.** Uses Cholesky algorithm to compute factorization of complex positive definite band matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d2 **Usage:** CALL CPBFA(ABD, LDA, N, M, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPBSL.** Uses factorization of complex positive definite band matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d2 **Usage:** CALL CPBSL(ABD, LDA, N, M, B) **Also see:** CPBCO CPBFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPFFT.** Compute the cross periodogram of two stationary time series using a fast Fourier transform. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10b3a2 **Usage:** CALL CPFFT (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CPLOT.** Prints a scatter diagram which condenses as many as 10 lines of plot into one line and trims extreme x- and y-values. Option: form of output. *Command in MINITAB proprietary interactive system.* **Classes:** L3b3a **Usage:** CPLOt y in C vs x in C [; subcommands LINES = K; CHARACTERS = K; XBOUNDS are from K to K; YBOUNDS are from K to K.] **Precision:** Single **Availability:** 855NOS.

**CPOCO.** Uses Cholesky algorithm to compute factorization of complex positive definite matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CPOCO(A, LDA, N, RCOND, Z, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPODI.** Uses factorization of complex positive definite matrix to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b, D3d1b **Usage:** CALL CPODI(A, LDA, N, DET, JOB) **Also see:** CPOCO CPOFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPOFA.** Uses Cholesky algorithm to compute factorization of complex positive definite matrix. *Portable Fortran sub-*

*routine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CPOFA(A, LDA, N, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPOFS.** Factors and solves positive definite symmetric complex system of linear equations. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2d1b **Usage:** CALL CPOFS(A, LDA, N, V, ITASK, IND, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPOIR.** Solves positive definite Hermitian complex system of linear equations and estimates the accuracy of the solution. (Requires n-by-n extra storage.) *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2d1b **Usage:** CALL CPOIR(A, LDA, N, V, ITASK, IND, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPOLY.** Finds the zeros of a polynomial with complex coefficients. Uses an inconvenient representation of two real arrays to represent complex numbers. *Proprietary Fortran subroutine in PORT.* **Classes:** F1a2 **Usage:** CALL CPOLY (DEGREE, OPR, OPI, ZEROR, ZEROI) **Precision:** Single (Double: DCPOLY) **Availability:** 855NOS, 205.

**CPOSL.** Uses factorization of complex positive definite matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CPOSL(A, LDA, N, B) **Also see:** CPOCO CPOFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPPCO.** Uses Cholesky algorithm to factor complex positive definite matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CPPCO(AP, N, RCOND, Z, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPPDI.** Uses factorization of complex positive definite matrix stored in packed form to compute determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b, D3d1b **Usage:** CALL CPPDI(AP, N, DET, JOB) **Also see:** CPPCO CPPFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPPFA.** Uses Cholesky algorithm to factor complex positive definite matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CPPFA(AP, N, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPPSL.** Uses factorization of complex positive definite matrix stored in packed form to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1b **Usage:** CALL CPPSL(AP, N, B) **Also see:** CPPCO CPPFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPQR79.** Computes all the zeros of a general complex polynomial using eigenvalue methods, requiring n-by-n storage for nth degree polynomial. *Portable Fortran subroutine in CMLIB (CPQR79 sublibrary).* **Classes:** F1a2 **Usage:** CALL CPQR79(NDEG, COEFF, ROOT, IERR, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CPSI.** Psi (digamma) of complex argument. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7c **Usage:** C = CPSI(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CPSI.** Logarithmic derivative of the gamma function for a complex argument. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** C = CPSI(ZIN) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CPTSL.** Solves systems with complex positive definite tridiagonal matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d2a **Usage:** CALL CPTSL(N, D, E, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CPZERO.** Computes all the zeros of a polynomial with complex coefficients. Error bounds are also obtained. Uses Newton's method for systems. *Portable Fortran subroutine in CMLIB (CPZERO sublibrary).* **Classes:** F1a2 **Usage:** CALL CPZERO(N, A, R, T, IFLAG, S) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CQRDC.** Computes QR decomposition of general complex matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D5 **Usage:** CALL CQRDC(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CQRSL.** Applies the output of CQRDC to compute coordinate transformations, projections, and least squares solutions (general complex matrix). *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c1, D9a1 **Usage:** CALL CQRSL(*13 parameters*) **Also see:** CQRDC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CRBCB.** Convert a real matrix in band storage mode to a complex matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CRBCB (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CRBRB.** Copy a real band matrix stored in band storage mode. *Proprietary Fortran subroutine in IMSL*

*MATH/LIBRARY.* **Classes:** D1b8 **Usage:** CALL CRBRB (*9 parameters*) **Precision:** Single (Double: DCRBRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CRBRG.** Convert a real matrix in band storage mode to a matrix in full storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CRBRG (*7 parameters*) **Precision:** Single (Double: DCRBRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CRGCG.** Copy a real general matrix to a complex general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CRGCG (N, A, LDA, B, LDB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CRGRB.** Convert a real matrix in full storage mode to a matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CRGRB (*7 parameters*) **Precision:** Single (Double: DCRGRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CRGRG.** Copy a real general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b8 **Usage:** CALL CRGRG (N, A, LDA, B, LDB) **Precision:** Single (Double: DCRGRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CROSSTABS.** Produces tables that are the joint distribution of two or more variables that have a limited number of distinct values. Handles integer, continuous, and character data. Options: summary statistics, several measures of association, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1c1, L9 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**CROTG.** Construct Givens plane rotation of complex matrix. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1b10 **Usage:** CALL CROTG(CA, CB, CC, CS) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CRRCR.** Copy a real rectangular matrix to a complex rectangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CRRCR (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSAKM.** Compute the Akima cubic spline interpolant. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL CSAKM (*5 parameters*) **Precision:** Single (Double: DCSAKM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSBRB.** Copy a real symmetric band matrix stored in band symmetric storage mode to a real band matrix stored in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CSBRB (*8 parameters*) **Precision:** Single (Double: DCSBRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSCAL.** Multiply a vector by a scalar, y = ay, both complex. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a6 **Usage:** CALL CSCAL(N, CA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSCAL.** Multiply a vector by a scalar, y = ay, both complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a6 **Usage:** CALL CSCAL (N, CA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSCON.** Compute a cubic spline interpolant which is consistent with the concavity of the data. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL CSCON (*6 parameters*) **Precision:** Single (Double: DCSCON) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSDEC.** Compute the cubic spline interpolant with specified derivative endpoint conditions. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL CSDEC (*9 parameters*) **Precision:** Single (Double: DCSDEC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSDER.** Evaluate the derivative of a cubic spline. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a2, K6a2 **Usage:** R = CSDER(*5 parameters*) **Precision:** Single (Double: DCSDER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSET.** Set the components of a vector to a scalar, all complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a1 **Usage:** CALL CSET (N, CA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSEVL.** Evaluates a Chebyshev polynomial series. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C3a2 **Usage:** R = CSEVL(X, CS, N) **Also see:** INITS **Precision:** Single (Double: DCSEVL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CSEVL.** Evaluate a series of Chebyshev polynomials. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C3a2 **Usage:** R = CSEVL(X, CS, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSFRG.** Extend a real symmetric matrix defined in its upper triangle to its lower triangle. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b9 **Usage:** CALL CSFRG (N, A, LDA) **Precision:** Single (Double: DCSFRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSHER.** Compute a Hermite cubic spline interpolant. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL CSHER (*6 parameters*) **Precision:** Single (Double: DCSHER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSICO.** Computes factorization of complex symmetric indefinite matrix and estimates its condition. *Portable Fortran* *subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CSICO(A, LDA, N, KPVT, RCOND, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSIDI.** Uses factorization of complex symmetric indefinite matrix to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a, D3d1a **Usage:** CALL CSIDI(*8 parameters*) **Also see:** CSICO CSIFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSIFA.** Computes factorization of complex symmetric indefinite matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CSIFA(A, LDA, N, KPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSIN.** sin(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CSIN(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CSIN.** sin(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CSIN(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSINH.** sinh(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** C = CSINH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CSINH.** sinh(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CSINH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSINT.** Compute the cubic spline interpolant with the 'not-a-knot' condition. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL CSINT (*5 parameters*) **Precision:** Single (Double: DCSINT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSISL.** Uses factorization of complex symmetric indefinite matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CSISL(A, LDA, N, KPVT, B) **Also see:** CSICO CSIFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSITG.** Evaluate the integral of a cubic spline. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a3, K6a3 **Usage:** R = CSITG(A, B, NINTV, BREAK, CSCOEF) **Precision:** Single (Double: DCSITG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSNDF.** Evaluate the noncentral chi-squared distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1c **Usage:** R = CSNDF(CHSQ, DF, ALAM) **Precision:** Single (Double: DCSNDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSORT.** Sorts a character array in either increasing or decreasing order. Optionally another character array can be carried along. *Portable Fortran subroutine in CMLIB (SSORT sublibrary).* **Classes:** N6a2c **Usage:** CALL CSORT(X, Y, N, KFLAG, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CSPCO.** Computes factorization of complex symmetric indefinite matrix stored in packed form and computes its condition. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CSPCO(AP, N, KPVT, RCOND, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSPDI.** Uses factorization of complex symmetric indefinite matrix stored in packed form to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a, D3d1a **Usage:** CALL CSPDI(*7 parameters*) **Also see:** CSPCO CSPFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSPDI.** Finds a numerical approximation to the first derivative at requested points in given input data by using spline interpolation. *Proprietary Fortran subroutine in PORT.* **Classes:** H1 **Usage:** CALL CSPDI (X, Y, N, XX, YY, YYP, NN) **Precision:** Single (Double: DCSPDI) **Availability:** 855NOS, 205.

**CSPER.** Compute the cubic spline interpolant with periodic boundary conditions. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** CALL CSPER (*5 parameters*) **Precision:** Single (Double: DCSPER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSPFA.** Computes factorization of complex symmetric indefinite matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CSPFA(AP, N, KPVT, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSPFE.** Evaluates a cubic spline function which has already been fit to n input data pairs (x,y) by CSPFI. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a1, K6a1 **Usage:** CALL CSPFE (X, Y, YP, YPP, N, XX, YY, NN) **Precision:** Single (Double: DCSPFE) **Availability:** 855NOS, 205.

**CSPFI.** Fits a cubic spline function to n input data pairs (x,y) with various endpoint conditions. *Proprietary Fortran subroutine in PORT.* **Classes:** E1a **Usage:** CALL CSPFI (X, Y, N, B, YP, YPP) **Also see:** CSPFE **Precision:** Single (Double: DCSPFI) **Availability:** 855NOS, 205.

**CSPIN.** Interpolates at requested points in given input data using a spline approximation, not a least squares fit. *Proprietary Fortran subroutine in PORT.* **Classes:** E1a **Usage:** CALL CSPIN (X, Y, N, XX, YY, NN) **Precision:** Single (Double: DCSPIN) **Availability:** 855NOS, 205.

**CSPQU.** Finds the integral of a function defined by pairs (x,y) of input points. The x's can be unequally spaced. Uses spline interpolation. *Proprietary Fortran subroutine in PORT.* **Classes:** H2a1b2 **Usage:** CALL CSPQU (X, Y, N, XLOW, XHIGH, ANS) **Precision:** Single (Double: DCSPQU) **Availability:** 855NOS, 205.

**CSPSL.** Uses factorization of complex symmetric indefinite matrix stored in packed form to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2d1a **Usage:** CALL CSPSL(AP, N, KPVT, B) **Also see:** CSPCO CSPFA **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSQRT.** Square root of a complex argument. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C2 **Usage:** C = CSQRT(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CSQRT.** Square root of a complex argument. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C2 **Usage:** C = CSQRT(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSROT.** Applies Givens plane rotation to complex matrix. *Portable Fortran subroutine in CMLIB (XBLAS sublibrary).* **Classes:** D1b10 **Usage:** CALL CSROT(N, CX, INCX, CY, INCY, C, S) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSROT.** Apply a complex Givens plane rotation. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a8 **Usage:** CALL CSROT (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSROTM.** Apply a complex modified Givens plane rotation. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a8 **Usage:** CALL CSROTM (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSSCAL.** Multiply a complex vector by a single-precision scalar, y = ay. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a6 **Usage:** CALL CSSCAL(N, SA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSSCAL.** Multiply a complex vector by a single-precision scalar, y = ay. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a6 **Usage:** CALL CSSCAL (N, SA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSSCV.** Compute a smooth cubic spline approximation to noisy data using cross-validation to estimate the smoothing parameter. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K5 **Usage:** CALL CSSCV (*6 parameters*) **Precision:** Single (Double: DCSSCV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSSED.** Smooth one-dimensional data by error detection. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K5 **Usage:** CALL CSSED (*7 parameters*) **Precision:** Single (Double: DCSSED) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSSMH.** Compute a smooth cubic spline approximation to noisy data. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K5 **Usage:** CALL CSSMH (*7 parameters*) **Precision:** Single (Double: DCSSMH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSSWD.** Estimate the nonnormalized cross-spectral density of two stationary time series using a spectral window given the time series data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10b3a3 **Usage:** CALL CSSWD (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSSWP.** Estimate the nonnormalized cross-spectral density of two stationary time series using a spectral window given the spectral densities and cross periodogram. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10b3a3 **Usage:** CALL CSSWP (*14 parameters*) **Precision:** Single (Double: DCSSWP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSTAT.** Compute cell frequencies, cell means, and cell sums of squares for multivariate data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L1c1 **Usage:** CALL CSTAT (*14 parameters*) **Precision:** Single (Double: DCSTAT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSUB.** Subtract each component of a vector from a scalar, x = a − x, all complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL CSUB (N, CA, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSVAL.** Evaluate a cubic spline. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a1, K6a1 **Usage:** R = CSVAL(X, NINTV, BREAK, CSCOEF) **Precision:** Single (Double: DCSVAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSVCAL.** Multiply a complex vector by a single-precision scalar and store the result in another complex vector, y = ax. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a6 **Usage:** CALL CSVCAL (N, SA, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CSVDC.** Computes singular value decomposition of general complex matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D6 **Usage:** CALL CSVDC(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CSWAP.** Interchange vectors X and Y, both complex. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a5 **Usage:** CALL CSWAP(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CSWAP.** Interchange vectors X and Y, both complex. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a5 **Usage:** CALL CSWAP (N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTABLE.** Prints a coded two-way table, each cell of which is coded with one character for features MAXIMUM, MINIMUM, or EXTREME, and codes for values between the hinges, between hinges and inner fences, between inner and outer fences, and beyond the outer fences. *Command in MINITAB proprietary interactive system.* **Classes:** L1b **Usage:** CTABle data in C, row levels in C, columns levels in C [; subcommand MAXIMUM or MINIMUM or EXTREME.] **Precision:** Single **Availability:** 855NOS.

**CTAN.** tan(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** C = CTAN(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CTAN.** tan(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CTAN(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTANH.** tanh(z). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** C = CTANH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**CTANH.** tanh(z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** C = CTANH(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTCHI.** Perform a chi-squared analysis of a two-way contingency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L9b **Usage:** CALL CTCHI (*13 parameters*) **Precision:** Single (Double: DCTCHI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTIME.** Return CPU time used in seconds. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** S3 **Usage:** C = CTIME () **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTIME.** Return CPU time used in seconds. *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** S3 **Usage:** R = CTIME () **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTLLF.** Log-linear fit of contingency table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L9c **Usage:** CALL CTLLF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTPR.** Compute exact probabilities for contingency tables. *Proprietary Fortran subroutine in IMSL.* **Classes:** L9b **Usage:** CALL CTPR (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTPRB.** Compute exact probabilities in a two-way contingency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L9b **Usage:** CALL CTPRB (*7 parameters*) **Precision:** Single (Double: DCTPRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTRBYC.** Analysis of a contingency table. *Proprietary Fortran subroutine in IMSL.* **Classes:** L9b **Usage:** CALL CTRBYC (A, IRC, JRC, IR, IC, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTRCO.** Estimates condition of complex triangular matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c3 **Usage:** CALL CTRCO(T, LDT, N, RCOND, Z, JOB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CTRDI.** Computes determinant and/or inverse of complex triangular matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c3, D3c3 **Usage:** CALL CTRDI(T, LDT, N, DET, JOB, INFO) **Also see:** CTRCO **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CTRHO.** Estimate the bivariate normal correlation coefficient using a contingency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b4 **Usage:** CALL CTRHO (*12 parameters*) **Precision:** Single (Double: DCTRHO) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTRSL.** Solves systems with complex triangular matrix. *Portable Fortran subroutine in CMLIB (LINPAKC sublibrary).* **Classes:** D2c3 **Usage:** CALL CTRSL(T, LDT, N, B, JOB, INFO) **Also see:** CTRCO **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**CTRST.** Compute contrast estimates and sums of squares. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7a1 **Usage:** CALL CTRST (*8 parameters*) **Precision:** Single (Double: DCTRST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CTTWO.** Perform a chi-squared analysis of a 2-by-2 contingency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L9a **Usage:** CALL CTTWO (*11 parameters*) **Precision:** Single (Double: DCTTWO) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CUNIT.** Convert X in units XUNITS to Y in units YUNITS. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** C19 **Usage:** CALL CUNIT (X, XUNITS, Y, YUNITS) **Precision:** Single (Double: DCUNIT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**CV64T60.** To convert one or several Cyber 200 64-bit numbers to Cyber 170 60-bit format. The data type of the input numbers may be either all floating-point, all integer, or a mixture thereof. The data types will be preserved in the translation. *Fortran/meta function in MAGEV.* **Classes:** A6 **Usage:** I = CV64T60(N, A, B) **Precision:** Single **Availability:** 205 (vectorized)

**CVCAL.** Multiply a vector by a scalar and store the result in another vector, y = ax, all complex. *Proprietary Fortran* *subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a6 **Usage:** CALL CVCAL (N, CA, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CVF32TI2.** To convert one or several half precision (32-bit floating-point) numbers to 16-bit two's complement integers, i.e., to the IBM data type INTEGER*2. *Fortran/meta function in MAGEV.* **Classes:** A6 **Usage:** R = CVF32TI2(N, A, B, IOPT, HMAX) **Precision:** Half **Availability:** 205 (vectorized)

**CVF32TR4.** To convert one or several half precision (32-bit floating-point) numbers to IBM 32-bit floating-point format, also known as REAL*4. *Fortran/meta subroutine in MAGEV.* **Classes:** A6 **Usage:** CALL CVF32TR4 (N, A, B) **Precision:** Half **Availability:** 205 (vectorized)

**CVF64VR4.** To convert one or several Cyber 200 64-bit floating-point numbers to VAX 32-bit floating-point format. *Fortran/meta function in MAGEV.* **Classes:** A6 **Usage:** I = CVF64VR4(N, A, B, IXBYTB) **Precision:** Single **Availability:** 205 (vectorized)

**CVI2TF32.** To convert one or several 16-bit two's complement integers, corresponding to IBM INTEGER*2 format, to Cyber 200 half precision format, i.e., into 32-bit floating-point numbers. *Fortran/meta subroutine in MAGEV.* **Classes:** A6 **Usage:** CALL CVI2TF32(N, A, B) **Precision:** Half **Availability:** 205 (vectorized)

**CVR4TF32.** To convert IBM 32-bit floating-point numbers, also known as REAL*4, to Cyber 200 half precision format, i.e., into 32-bit floating-point numbers. *Fortran/meta function in MAGEV.* **Classes:** A6 **Usage:** R = CVR4TF32(N, A, B) **Precision:** Half **Availability:** 205 (vectorized)

**CVTSI.** Convert a character string containing an integer number into the corresponding integer form. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N3 **Usage:** CALL CVTSI (STRING, NUMBER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CVVR4F64.** To convert one or several VAX 32-bit floating-point numbers to Cyber 200 64-bit floating-point format. *Fortran/meta subroutine in MAGEV.* **Classes:** A6 **Usage:** CALL CVVR4F64(N, A, IXBYTA, B) **Precision:** Single **Availability:** 205 (vectorized)

**CWPL.** Weierstrass P-function with primitive half-periods 1/2 [AMS55 (18.1)]. The corresponding invariants are g2=4L, where L=2.62205... is the Lemniscate Constant [AMS55 (18.14.7)] and g3=0. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C15 **Usage:** R = CWPL(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CWPLD.** First derivative of CWPL. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C15 **Usage:** R = CWPLD(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CWPQ.** Weierstrass P-function in the equianharmonic case for complex argument with unit period parallelogram. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C15 **Usage:** R = CWPQ(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CWPQD.** First derivative of CWPQ. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C15 **Usage:** R = CWPQD(Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CZCDOT.** Compute the sum of a complex scalar plus a complex conjugate dot product, a + conjg(x)·y, using a double-precision accumulator. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CZCDOT(N, CA, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CZDOTA.** Compute the sum of a complex scalar, a complex dot product and the double-complex accumulator, which is set to the result ACC = ACC + a + x·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CZDOTA(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CZDOTC.** Compute the complex conjugate dot product, conjg(x)·y, using a double-precision accumulator. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CZDOTC(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CZDOTI.** Compute the sum of a complex scalar plus a complex dot product using a double-complex accumulator, which is set to the result ACC = a + x·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CZDOTI(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CZDOTU.** Compute the complex dot product x·y using a double-precision accumulator. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CZDOTU(N, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**CZUDOT.** Compute the sum of a complex scalar plus a complex dot product, a + x·y, using a double-precision accumulator. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** C = CZUDOT(N, CA, CX, INCX, CY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

# D

**D.** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real square (general) matrix. Orthogonal transformations are used. Real and imaginary components of output values are returned in separate real arrays. Estimates of the number of correct significant digits in the eigenvalue estimates can also be obtained. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4a2 **Usage:** CALL D(*10 parameters*) **Precision:** Single **Availability:** PC.

**D01AHF.** Computes a definite integral over a finite range to a specified relative accuracy using a method described by Patterson. *Proprietary Fortran function in NAG.* **Classes:** H2a1a1 **Usage:** R = D01AHF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01AJF.** General-purpose integrator which calculates an approximation to the integral of a function F(x) over a finite interval (A,B). *Proprietary Fortran subroutine in NAG.* **Classes:** H2a1a1 **Usage:** CALL D01AJF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01AKF.** Adaptive integrator, especially suited to oscillating, non-singular integrands, which calculates an approximation to the integral of a function F(x) over a finite interval (A,B). *Proprietary Fortran subroutine in NAG.* **Classes:** H2a2a1 **Usage:** CALL D01AKF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01ALF.** Is a general purpose integrator which calculates an approximation to the integral of a function F(x) over a finite interval (A,B), where the integrand may have local singular behaviour at a finite number of points within the integration interval. *Proprietary Fortran subroutine in NAG.* **Classes:** H2a2a1 **Usage:** CALL D01ALF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01AMF.** Calculates an approximation to the integral of a function F(x) over an infinite or semi-infinite interval (A,B). *Proprietary Fortran subroutine in NAG.* **Classes:** H2a3a1, H2a4a1 **Usage:** CALL D01AMF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01ANF.** Calculates an approximation to the cosine or the sine transform of a function G over (A,B), i.e., the integral of G(x)cos($\omega$x) or G(x)sin($\omega$x) over (A,B) (for a user-specified value of omega). *Proprietary Fortran subroutine in NAG.* **Classes:** H2a2a1 **Usage:** CALL D01ANF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01APF.** Adaptive integrator which calculates an approximation to the integral of a function G(x)W(x) over (A,B) where the weight function W has end-point singularities of algebraico-logarithmic type. (See input parameter KEY.) *Proprietary Fortran subroutine in NAG.* **Classes:** H2a2a1 **Usage:** CALL D01APF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01AQF.** Calculates an approximation to the Hilbert transform of a function G(x) over (A,B), i.e., the integral of G(x)/(x-C) over (A,B), for user-specified values of A,B,C. *Proprietary Fortran subroutine in NAG.* **Classes:** H2a2a1, J4 **Usage:** CALL D01AQF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01ARF.** Computes definite and indefinite integrals over a finite range to a specified relative or absolute accuracy, using a method described by Patterson. *Proprietary Fortran subroutine in NAG.* **Classes:** H2a1a1 **Usage:** CALL D01ARF(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01BAF.** Computes an estimate of the definite integral of a function of known analytical form, using a Gaussian quadrature formula with a specified number of abscissae. Formulae are provided for a finite interval (Gauss-Legendre), a semi-infinite interval (Gauss-Laguerre, Gauss-rational), and an infinite interval (Gauss-Hermite). *Proprietary Fortran function in NAG.* **Classes:** H2a1a2, H2a3a2, H2a4a2 **Usage:** R = D01BAF(D01XXX, A, B, N, FUN, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01BBF.** Returns the weights and abscissae appropriate to a Gaussian quadrature formula with a specified number of abscissae. The formulae provided are Gauss-Legendre, Gauss-rational, Gauss-Laguerre and Gauss-Hermite. *Proprietary Fortran subroutine in NAG.* **Classes:** H2c **Usage:** CALL D01BBF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01BCF.** Returns the weights (normal or adjusted) and abscissae for a Gaussian integration rule with a specified number of abscissae. Six different types of Gauss rule are allowed. *Proprietary Fortran subroutine in NAG.* **Classes:** H2c **Usage:** CALL D01BCF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01BDF.** Calculates an approximation to the integral of a function over a finite interval (A,B). It is non-adaptive and as such is recommended for the integration of smooth functions. These exclude integrands with singularities, derivative singularities or high peaks on (A,B), or which oscillate too strongly on (A,B). *Proprietary Fortran subroutine in NAG.* **Classes:** H2a1a1 **Usage:** CALL D01BDF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01DAF.** Attempts to evaluate a double integral to a specified absolute accuracy by repeated applications of the method described by Patterson. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b1a1 **Usage:** CALL D01DAF

(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01EAF.** Computes approximations to the integrals of a vector of similar functions, each defined over the same multi-dimensional hyper-rectangular region. The routine uses an adaptive subdivision strategy, and also computes absolute error estimates. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b1a1 **Usage:** CALL D01EAF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01FBF.** Computes an estimate of a multidimensional integral (from 1 to 20 dimensions), given the analytic form of the integrand and suitable Gaussian weights and abscissae. *Proprietary Fortran function in NAG.* **Classes:** H2b1a2 **Usage:** R = D01FBF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01FCF.** Attempts to evaluate a multidimensional integral (up to 15 dimensions), with constant and finite limits, to a specified relative accuracy, using an adaptive subdivision strategy. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b1a1 **Usage:** CALL D01FCF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01FDF.** Calculates an approximation to a definite integral in up to 30 dimensions, using the method of Sag and Szekeres. The region of integration is an n-sphere, or by built-in transformation via the unit n-cube, any product region. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b1a2 **Usage:** CALL D01FDF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01GAF.** Integrates a function which is specified numerically at four or more points, over the whole of its specified range, using third-order finite-difference formulae with error estimates, according to a method due to Gill and Miller. *Proprietary Fortran subroutine in NAG.* **Classes:** H2a1b2 **Usage:** CALL D01GAF (X, Y, N, ANS, ER, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01GBF.** Returns an approximation to the integral of a function over a hyper-rectangular region, using a Monte-Carlo method. An approximate relative error estimate is also returned. This routine is suitable for low accuracy work. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b1a1 **Usage:** CALL D01GBF(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01GCF.** Calculates an approximation to a definite integral in up to 20 dimensions, using the Korobov-Conroy number theoretic method. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b1a2 **Usage:** CALL D01GCF(*10 parameters*) **Also see:** D01GYF D01GZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01JAF.** Attempts to evaluate an integral over an n-dimensional sphere (n=2, 3, or 4), to a user specified absolute or relative accuracy, by means of a modified Sag-Szekeres method. The routine can handle singularities on the surface or at the centre of the sphere, and returns an error estimate. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b2a1 **Usage:** CALL D01JAF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D01PAF.** Returns a sequence of approximations to the integral of a function over a multi-dimensional simplex, together with an error estimate for the last approximation. *Proprietary Fortran subroutine in NAG.* **Classes:** H2b2a2 **Usage:** CALL D01PAF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02AGF.** Solves the two-point boundary-value problem for a system of ordinary differential equations, using initial value techniques and Newton iteration; it generalizes subroutine D02HAF to include the case where parameters other than boundary values are to be determined. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b2, I1b3 **Usage:** CALL D02AGF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02BAF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02BAF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02BBF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, and returns the solution at points specified by the user. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02BBF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02BDF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, and computes a global error estimate check. A stiffness check is also available. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02BDF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02BGF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, until a specified component attains a given value. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02BGF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02BHF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method, until a user-specified function of the solution is zero. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02BHF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02CAF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1b **Usage:** CALL D02CAF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02CBF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method, and returns the solution at points specified by the user. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1b **Usage:** CALL D02CBF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02CGF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method, until a specified component attains a given value. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1b **Usage:** CALL D02CGF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02CHF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method, until a user-specified function of the solution is zero. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1b **Usage:** CALL D02CHF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02EAF.** Integrates a stiff system of first-order differential equations over a range with suitable initial conditions, using a variable-order variable-step method implementing the Backward Differentiation Formulae. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02EAF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02EBF.** Integrates a stiff system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order, variable-step method implementing the Backward Differentiation Formulae, and returns the solution at points specified by the user. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02EBF (X, XEND, N, Y, TOL, IRELAB, FCN, MPED, PEDERV, OUTPUT, W, IW, IFAIL **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02EGF.** Integrates a stiff system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step method implementing the Backward Differentiation Formulae, until a specified component attains a given value. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02EGF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02EHF.** Integrates a stiff-system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order, variable-step method implementing the Backward Differentiation Formulae, until a user specified function of the solution is zero. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02EHF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02EJF.** Integrates a stiff system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order, variable-step method implementing the Backward Differentiation Formulae, until a user-specified function, if supplied, of the solution is zero, and returns the solution at points specified by the user, if desired. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02EJF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02GAF.** Solves the two-point boundary-value problem with assigned boundary values for a system of ordinary differential equations, using a deferred correction technique and a Newton iteration. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b2 **Usage:** CALL D02GAF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02GBF.** Solves a general linear two-point boundary value problem for a system of ordinary differential equations using a deferred correction technique. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b1 **Usage:** CALL D02GBF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02HAF.** Solves the two-point boundary-value problem for a system of ordinary differential equations. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b2 **Usage:** CALL D02HAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02HBF.** Solves the two-point boundary-value problem for a system of ordinary differential equations, using initial value techniques (D02PAF) and Newton iteration; it generalizes subroutine D02HAF to include the case where parameters other than boundary values are to be determined. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b2, I1b3 **Usage:** CALL D02HBF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02JAF.** Solves a regular linear two-point boundary value problem for a single N(th) order ordinary differential equation by a Chebyshev series using collocation and least squares. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b1 **Usage:** CALL D02JAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02JBF.** Solves a regular linear two-point boundary value problem for a system of ordinary differential equations by Chebyshev series using collocation and least squares. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b1 **Usage:** CALL D02JBF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02KAF.** Finds a specified eigenvalue of a regular second-order Sturm-Liouville system defined on a finite range, using a Pruefer transformation and a shooting method. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b3 **Usage:** CALL D02KAF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02KDF.** Finds a specified eigenvalue of a regular or singular second-order Sturm-Liouville system on a finite or infinite range, using a Pruefer transformation and a shooting method. Provision is made for discontinuities in the coefficient functions or their derivatives. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b3 **Usage:** CALL

D02KDF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02KEF.** Finds a specified eigenvalue of a regular singular second-order Sturm-Liouville system on a finite or infinite range, using a Pruefer transformation and a shooting method. It also reports values of the eigenfunction and its derivatives. Provision is made for discontinuities in the coefficient functions or their derivatives. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b3 **Usage:** CALL D02KEF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NBF.** Forward communication routine for integrating stiff systems of explicit ordinary differential equations when the Jacobian is a full matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NBF (*21 parameters*) **Also see:** D02NSF D02NVF D02NWF D02NYF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NCF.** Forward communication routine for integrating stiff systems of explicit ordinary differential equations when the Jacobian is a banded matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NCF (*23 parameters*) **Also see:** D02NTF D02NVF D02NWF D02NYF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NDF.** Forward communication routine for integrating stiff systems of explicit ordinary differential equations when the Jacobian is a sparse matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NDF (*23 parameters*) **Also see:** D02NUF D02NVF D02NWF D02NYF D02NXF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NGF.** Forward communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations when the Jacobian is a full matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NGF (*22 parameters*) **Also see:** D02NSF D02NVF D02NWF D02NYF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NHF.** Forward communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations when the Jacobian is a banded matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NHF (*24 parameters*) **Also see:** D02NTF D02NVF D02NWF D02NYF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NJF.** Forward communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations when the Jacobian is a sparse matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NJF (*24 parameters*) **Also see:** D02NUF D02NVF D02NWF D02NYF D02NXF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NMF.** Reverse communication routine for integrating stiff systems of explicit ordinary differential equations. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NMF (*24 parameters*) **Also see:** D02NVF D02NRF D02NWF D02NSF D02NTF D02NUF D02NXF D02NYF D02NZF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NNF.** Reverse communication routine for integrating stiff systems of implicit ordinary differential equations coupled with algebraic equations. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02NNF (*25 parameters*) **Also see:** D02NRF D02NVF D02NWF D02NSF D02NTF D02NUF D02NXF D02NZF D02NYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NRF.** Enquiry routine for communicating with D02NMF or D02NNF when supplying columns of a sparse Jacobian matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NRF (J, IPLACE, INFORM) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NSF.** Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if full matrix linear algebra is required. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NSF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NTF.** Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if banded matrix linear algebra is required. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NTF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NUF.** Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if sparse matrix linear algebra is required. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NUF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NVF.** Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if Backward Differentiation Formulae (BDF) are to be used. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NVF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NWF.** Setup routine which must be called by the user, prior to an integrator in the D02N subchapter, if the BLEND formulae are to be used. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NWF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NXF.** Optional output routine which the user may call, on exit from an integrator in the D02N subchapter, if sparse matrix linear algebra has been selected. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NXF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NYF.** Diagnostic routine which the user may call either after any user specified exit or after a mid-integration error exit from any of the integrators in the D02N subchapter. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NYF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02NZF.** Setup routine which must be called, if optional inputs need resetting, prior to a continuation call to any of the integrators in the D02N subchapter. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02NZF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02PAF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a Runge-Kutta-Merson method. A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended only to experienced users. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02PAF (*14 parameters*) **Also see:** D02XAF D02XBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02QAF.** Integrates a system of first-order ordinary differential equations over a range with suitable initial conditions, using a variable-order variable-step Adams method. A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended only to experienced users. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1b **Usage:** CALL D02QAF (*14 parameters*) **Also see:** D02XGF D02XHF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02QBF.** Integrates a stiff system of first-order ordinary differential equations, over a range with suitable initial conditions, using a variable-order variable-step Gear method. A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended only to experienced users. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02QBF (*17 parameters*) **Also see:** D02XGF D02XHF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02QDF.** Integrates a stiff system of first order ordinary differential equations, over a range with suitable initial conditions, using a variable-order, variable-step method based on the Backward Differentiation Formulae (BDF). A variety of facilities for interrupting the calculation are provided. This routine is relatively complicated and is recommended to experienced users only. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a2 **Usage:** CALL D02QDF (*18 parameters*) **Also see:** D02QQF D02XJF D02XKF D02QDF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02QQF.** Sets up interrupts for use in D02QDF. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02QQF (COMM, CHK, N, W, IW, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02RAF.** Solves the two-point boundary-value problem with general boundary conditions for a system of ordinary differential equations, using a deferred correction technique and Newton iteration. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b2 **Usage:** CALL D02RAF (*24 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02SAF.** Solves a two-point boundary-value problem for a system of first order ordinary differential equations with boundary conditions, combined with additional algebraic equations. It uses initial value techniques and a modified Newton iteration in a shooting and matching method. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b2 **Usage:** CALL D02SAF (*23 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02TGF.** Solves a system of linear ordinary differential equations by least-squares fitting of a series of Chebyshev polynomials using collocation. *Proprietary Fortran subroutine in NAG.* **Classes:** I1b1 **Usage:** CALL D02TGF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02XAF.** Interpolates the system of first-order ordinary differential equations from information provided by the Runge-Kutta-Merson routine D02PAF. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02XAF (*9 parameters*) **Also see:** D02PAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02XBF.** Interpolates one component of the solution of a system of first-order ordinary differential equations from information provided by the Runge-Kutta-Merson routine D02PAF. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02XBF (*10 parameters*) **Also see:** D02PAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02XGF.** Interpolates the solution of a system of first-order ordinary differential equations from information provided by the Adams routine D02QAF or the Gear routine D02QBF. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02XGF (*9 parameters*) **Also see:** D02QAF D02QBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02XHF.** Interpolates one component of the solution of a system of first-order ordinary differential equations from information provided by the Adams routine D02QAF or the Gear routine D02QBF. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02XHF (*10 parameters*) **Also see:** D02QAF D02QBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02XJF.** Interpolates components of the solution of a system of first-order ordinary differential equations from information provided by the integrators in the D02N subchapter (or by the routine D02QDF). *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02XJF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02XKF.** Interpolates components of the solution of a system of first-order ordinary differential equations from information provided by the integrators in the D02N subchapter (or by the routine D02QDF). It provides C1 interpolation suitable for general use. *Proprietary Fortran subroutine in NAG.* **Classes:** I1c **Usage:** CALL D02XKF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02YAF.** Integrates a system of first-order ordinary differential equations over one step, using Merson's Runge-Kutta method. *Proprietary Fortran subroutine in NAG.* **Classes:** I1a1a **Usage:** CALL D02YAF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D02ZAF.** Calculates the weighted norm of the local error estimate from inside a MONITR routine called from an integrator in the D02N subchapter. *Proprietary Fortran function in NAG.* **Classes:** I1c **Usage:** R = D02ZAF(NEQ, V, W, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03EAF.** Solves Laplaces's equation in two dimensions for an arbitrary domain bounded internally or externally by one or more closed contours, given the value of either the unknown function or its normal derivative (into the domain) at each point of the boundary. *Proprietary Fortran subroutine in NAG.* **Classes:** I2b1a1b **Usage:** CALL D03EAF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03EBF.** Uses the Strongly Implicit Procedure to calculate the solution to a system of simultaneous algebraic equations of five point molecule form on a two-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example (r, theta), can be used, being equivalent to a rectangular box.) *Proprietary Fortran subroutine in NAG.* **Classes:** I2b4b **Usage:** CALL D03EBF (*25 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03ECF.** Uses the Strongly Implicit Procedure to calculate the solution to a system of simultaneous algebraic equations of seven point molecule form on a three-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example, can be used if it is equivalent to a rectangular box.) *Proprietary Fortran subroutine in NAG.* **Classes:** I2b4b **Usage:** CALL D03ECF (N1, N2, N3, N1M, N2M, A, B, C, D, E, F, G, Q, T, APARAM, ITMAX, ITCOUN, ITUSED, NDIR, IXN, IYN, IZN, CONRES, CONCHN, RESIDS, CHNGS, WRKSP1, **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03EDF.** Solves 7-diagonal systems of linear equations which arise from the discretisation of an elliptic partial differential equation on a rectangular region. This routine uses a multigrid technique. *Proprietary Fortran subroutine in NAG.* **Classes:** I2b4b **Usage:** CALL D03EDF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03MAF.** Places a triangular mesh over a given 2-dimensional region. The region may have any shape, including one with holes. *Proprietary Fortran subroutine in NAG.* **Classes:** I2b4a, P **Usage:** CALL D03MAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03PAF.** Integrates a single linear or nonlinear parabolic partial differential equation in one space variable, using the method of lines and Gear's method. *Proprietary Fortran subroutine in NAG.* **Classes:** I2a1a **Usage:** CALL D03PAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03PBF.** Integrates a system of linear or nonlinear parabolic partial differential equations in one space variable, using the method of lines and Gear's method. *Proprietary Fortran subroutine in NAG.* **Classes:** I2a1a **Usage:** CALL D03PBF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03PGF.** Integrates a system of nonlinear parabolic partial differential equations in one space variable, using the method of lines and Gear's method. This routine provides quite general facilities; for simpler versions see D03PAF (for a single equation), or D03PBF (for simple systems). *Proprietary Fortran subroutine in NAG.* **Classes:** I2a1a **Usage:** CALL D03PGF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03UAF.** Performs at each call one iteration of the Strongly Implicit Procedure. It is used to calculate on successive calls a sequence of approximate corrections to the current estimate of the solution when solving a system of simultaneous algebraic equations for which the iterative up-date matrix is of five point molecule form on a two-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example (r,theta), can be used, being equivalent to a rectangular box.) *Proprietary Fortran subroutine in NAG.* **Classes:** I2b4b **Usage:** CALL D03UAF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D03UBF.** Performs at each call one iteration of the Strongly Implicit Procedure. It is used to calculate on successive calls the sequence of approximate corrections to the solution when solving a system of simultaneous algebraic equations for which the iterative up-date matrix is of seven-point molecule form on a three-dimensional topologically-rectangular mesh. (Topological means that a polar grid, for example, can be used if it is equivalent to a rectangular box.) *Proprietary Fortran subroutine in NAG.* **Classes:** I2b4b **Usage:** CALL D03UBF (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D04AAF.** Calculates a set of derivatives (up to order 14) of a function of one real variable at a point, together with a corresponding set of error estimates, using an extension of the Neville algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** H1 **Usage:** CALL D04AAF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D05AAF.** Solves a linear, non-singular Fredholm equation of the second kind with a split kernel. *Proprietary Fortran subroutine in NAG.* **Classes:** I3 **Usage:** CALL D05AAF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D05ABF.** Solves any linear non-singular Fredholm integral equation of the second kind with a smooth kernel. *Pro-*

*prietary Fortran subroutine in NAG.* **Classes:** I3 **Usage:** CALL D05ABF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**D1A3A.** Compute single precision sum of absolute values of components of vector. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1a3a **Usage:** R = D1A3A(N, SX, INCX) **Precision:** Single (Double: D1A3AD) **Availability:** PC.

**D1A3AD.** *Double precision version of D1A3A.*

**D1A3B.** Forms the square root of the sum of the squares of an array of real numbers, (Euclidean, $l_2$ norm). *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1a3b **Usage:** R = D1A3B(N, SX, INCX) **Precision:** Single (Double: D1A3BD) **Availability:** PC.

**D1A3BD.** *Double precision version of D1A3B.*

**D1A3C.** Finds the index of the real array element having maximum absolute value. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1a2, D1a3c **Usage:** I = D1A3C(N, SX, INCX) **Precision:** Single (Double: D1A3CD) **Availability:** PC.

**D1A3CD.** *Double precision version of D1A3C.*

**D1A41.** Forms the scalar (dot) product of two real arrays using double precision accumulation and multiplication. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1a4 **Usage:** R = D1A41(N, SX, INCX, SY, INCY) **Precision:** Single (Double: D1A41D) **Availability:** PC.

**D1A41D.** *Double precision version of D1A41.*

**D1A5.** Interchange two real vectors X and Y. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1a5 **Usage:** CALL D1A5(N, SX, INCX, SY, INCY) **Precision:** Single (Double: D1A5D) **Availability:** PC.

**D1A5C.** Copy a real vector X to a real vector Y. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1a5 **Usage:** CALL D1A5C(N, SX, INCX, SY, INCY) **Precision:** Single (Double: D1A5CD) **Availability:** PC.

**D1A5CD.** *Double precision version of D1A5C.*

**D1A5D.** *Double precision version of D1A5.*

**D1A6.** Compute a real constant times a real vector. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1a6 **Usage:** CALL D1A6(N, SA, SX, INCX) **Precision:** Single (Double: D1A6D) **Availability:** PC.

**D1A6D.** *Double precision version of D1A6.*

**D1A7.** Compute a real constant times a real vector plus a real vector. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1a7 **Usage:** CALL D1A7(N, SA, SX, INCX, SY, INCY) **Precision:** Single (Double: D1A7D) **Availability:** PC.

**D1A7D.** *Double precision version of D1A7.*

**D1A8.** Applies a plane rotation to a 2-by-N matrix whose first and second row are given in the real arrays X and Y. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1a8 **Usage:** CALL D1A8(N, X, INCX, Y, INCY, C, S) **Precision:** Single (Double: D1A8D) **Availability:** PC.

**D1A8D.** *Double precision version of D1A8.*

**D1A8G.** Performs a Givens plane rotation. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1a8 **Usage:** CALL D1A8G(A, B, C, S) **Precision:** Single (Double: D1A8GD) **Availability:** PC.

**D1A8GD.** *Double precision version of D1A8G.*

**D1B2.** Finds the element of a real rectangular matrix with the largest absolute value, the infinity norm. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1b2 **Usage:** R = D1B2(A, M, N, LDA) **Precision:** Single (Double: D1B2D) **Availability:** PC.

**D1B21.** Finds the 1-norm of a real matrix, the maximum of the sums of the absolute values of the column elements. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1b2 **Usage:** R = D1B21(A, LDA, M, N) **Precision:** Single (Double: D1B21D) **Availability:** PC.

**D1B21D.** *Double precision version of D1B21.*

**D1B22.** Finds the 1-norm of a real symmetric matrix stored in packed form, the maximum of the sums of the absolute values of the column elements. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1b2 **Usage:** R = D1B22(A, N) **Precision:** Single (Double: D1B22D) **Availability:** PC.

**D1B22D.** *Double precision version of D1B22.*

**D1B2D.** *Double precision version of D1B2.*

**D1B2S.** Finds the element with maximum absolute value, the infinity norm, of a real symmetric matrix stored in packed form. *Proprietary Fortran function in the Scientific Desk.* **Classes:** D1b2 **Usage:** R = D1B2S(A, N) **Precision:** Single (Double: D1B2SD) **Availability:** PC.

**D1B2SD.** *Double precision version of D1B2S.*

**D1B3.** Transposes a real rectangular matrix. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b3 **Usage:** CALL D1B3(A, M, N, LDA, B, LDB) **Precision:** Single (Double: D1B3D) **Availability:** PC.

**D1B3D.** *Double precision version of D1B3.*

**D1B5.** Adds the full storage mode n-by-n matrix A to the symmetric n-by-n matrix B stored in packed form and puts the result in the full matrix C. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b5 **Usage:** CALL D1B5(A, N, LDA, B, C, LDC) **Precision:** Single (Double: D1B5D) **Availability:** PC.

**D1B5D.** *Double precision version of D1B5.*

**D1B61.** Multiplies an m-by-n real matrix A by an n-by-n symmetric real matrix B stored in packed form and stores the result in the real m-by-n matrix C. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b6 **Usage:** CALL D1B61(A, LDA, M, N, B, C, LDC) **Precision:** Single (Double: D1B61D) **Availability:** PC.

**D1B61D.** *Double precision version of D1B61.*

**D1B62.** Multiplies an n-by-n real symmetric matrix A stored in packed form by an n-by-m real matrix B stores the result in the real n-by-m matrix C. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b6 **Usage:** CALL D1B62(A, N, B, M, LDB, C, LDC) **Precision:** Single (Double: D1B62D) **Availability:** PC.

**D1B62D.** *Double precision version of D1B62.*

**D1B63.** Forms $A^T$ times A, where A is a real n-by-m matrix. The result is stored in either packed or full form. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b6 **Usage:** CALL D1B63(A, LDA, N, M, LDC, IOPT, B, C) **Precision:** Single (Double: D1B63D) **Availability:** PC.

**D1B63D.** *Double precision version of D1B63.*

**D1B64.** Forms $B=A^T$ times A, where A and B are real symmetric matrices stored in packed form. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b6 **Usage:** CALL D1B64(A, N, B) **Precision:** Single (Double: D1B64D) **Availability:** PC.

**D1B64D.** *Double precision version of D1B64.*

**D1B65.** Multiplies two symmetric real matrices stored in packed form. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b6 **Usage:** CALL D1B65(A, N, B, C, LDC) **Precision:** Single (Double: D1B65D) **Availability:** PC.

**D1B65D.** *Double precision version of D1B65.*

**D1B6F.** General matrix multiply. Multiplies real matrix A(L,M) by real matrix B(M,N) and stores the result in C(L,N). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b6 **Usage:** CALL D1B6F(A, LDA, L, M, B, LDB, N, C, LDC) **Precision:** Single (Double: D1B6FD) **Availability:** PC.

**D1B6FD.** *Double precision version of D1B6F.*

**D1B9.** Converts a symmetric real matrix, A, stored in packed form to a full matrix, B. A and B may occupy the same memory locations. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D1b9 **Usage:** CALL D1B9(A, N, IOPT, LDB, B) **Precision:** Single (Double: D1B9D) **Availability:** PC.

**D1B9D.** *Double precision version of D1B9.*

**D1MACH.** *Double precision version of R1MACH.*

**D2A11.** Factors a real matrix by Gaussian elimination. Several other routines such as D2A12, D2A13, or D2A14, can make use of the factorization to solve a system of linear equations, compute inverse, or determinant. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a1 **Usage:** CALL D2A11(A, LDA, N, IPVT, KE) **Precision:** Single (Double: D2A11D) **Availability:** PC.

**D2A11D.** *Double precision version of D2A11.*

**D2A12.** Solves the real system of equations AX=B using the factors computed by D2A11. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a1 **Usage:** CALL D2A12(A, LDA, N, IPVT, B) **Also see:** D2A11 **Precision:** Single (Double: D2A12D) **Availability:** PC.

**D2A12D.** *Double precision version of D2A12.*

**D2A13.** Computes the determinant and/or inverse of a real matrix using the factors computed by D2A11. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a1, D3a1 **Usage:** CALL D2A13(A, LDA, N, IPVT, DET, WORK, JOB) **Also see:** D2A11 **Precision:** Single (Double: D2A13D) **Availability:** PC.

**D2A13D.** *Double precision version of D2A13.*

**D2A14.** Estimates the condition of a real matrix, given its factors from routine D2A11 and its 1-norm from routine D1B21. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a1 **Usage:** CALL D2A14(A, LDA, ANORM, N, IPVT, Z, RCOND) **Also see:** D2A11 D1B21 **Precision:** Single (Double: D2A14D) **Availability:** PC.

**D2A14D.** *Double precision version of D2A14.*

**D2A21.** Factors a real band matrix by elimination. Routine D2A22 can then be used to solve a banded system of linear equations. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a2 **Usage:** CALL D2A21(ABD, LDA, N, ML, MU, IPVT, KE) **Precision:** Single (Double: D2A21D) **Availability:** PC.

**D2A21D.** *Double precision version of D2A21.*

**D2A22.** Solves the real band system AX=B or $A^T X=B$ using the factors computed by D2A21. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a2 **Usage:** CALL D2A22(ABD, LDA, N, ML, MU, IPVT, B, JOB) **Also see:** D2A21 **Precision:** Single (Double: D2A22D) **Availability:** PC.

**D2A22D.** *Double precision version of D2A22.*

**D2A2A.** Solves a system of linear equations where the coefficient matrix is tridiagonal. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2a2a **Usage:** CALL D2A2A(N, C, D, E, B, KE) **Precision:** Single (Double: D2A2AD) **Availability:** PC.

**D2A2AD.** *Double precision version of D2A2A.*

**D2B11.** Factors a real symmetric positive definite matrix stored in packed form. Then routines such as D2B12, D2B13, or D2B14 can be used to solve a system of linear equations, or find the determinant, condition number, or inverse of the matrix. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1b **Usage:** CALL D2B11(AP, N, KE) **Precision:** Single (Double: D2B11D) **Availability:** PC.

**D2B11D.** *Double precision version of D2B11.*

**D2B12.** Solves the real symmetric positive definite system Ax=B using the factorization obtained from D2B11. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1b **Usage:** CALL D2B12(AP, N, B) **Also see:** D2B11 **Precision:** Single (Double: D2B12D) **Availability:** PC.

**D2B12D.** *Double precision version of D2B12.*

**D2B13.** Finds the determinant and/or the inverse of a real symmetric positive definite matrix using the factorization obtained from D2B11. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1b, D3b1b **Usage:** CALL D2B13(AP, N, DET, JOB) **Also see:** D2B11 **Precision:** Single (Double: D2B13D) **Availability:** PC.

**D2B13D.** *Double precision version of D2B13.*

**D2B14.** Finds the condition number of a real symmetric positive definite matrix A stored in packed form using the factorization obtained from D2B11 and the 1-norm of A obtained from D1B22. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1b **Usage:** CALL D2B14(AP, N, ANORM, RCOND, Z) **Also see:** D2B11 D1B22 **Precision:** Single (Double: D2B14D) **Availability:** PC.

**D2B14D.** *Double precision version of D2B14.*

**D2B1A.** Factors a real symmetric matrix stored in packed form by elimination with symmetric pivoting. The factors can then be used by routine D2B1A2 to solve a symmetric system of linear equations, routine D2B1A3 can be used to form the inverse or compute the determinant of the matrix, or routine D2B1A4 can be used to compute the condition number of the matrix. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1a **Usage:** CALL D2B1A(AP, N, KPVT, KE) **Precision:** Single **Availability:** PC.

**D2B1A2.** Solves the real symmetric system AX=B using the factors computed by D2B1A1. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1a **Usage:** CALL D2B1A2(AP, N, KPVT, B) **Also see:** D2B1A1 **Precision:** Single **Availability:** PC.

**D2B1A3.** Computes the determinant, inertia, or inverse of a real symmetric matrix using the factors from D2B1A1. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1a, D3b1a **Usage:** CALL D2B1A3(*7 parameters*) **Also see:** D2B1A1 **Precision:** Single **Availability:** PC.

**D2B1A4.** Estimates the condition of a real symmetric matrix A stored in packed form from the factors computed by D2B1A1. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b1a **Usage:** CALL D2B1A4(AP, N, KPVT, ANORM, RCOND, Z) **Also see:** D2B1A1 **Precision:** Single **Availability:** PC.

**D2B2S.** Solves a system of linear equations where the coefficeint matrix is symmetric positive definite tridiagonal. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D2b2a **Usage:** CALL D2B2S(N, D, E, B) **Precision:** Single (Double: D2B2SD) **Availability:** PC.

**D2B2SD.** *Double precision version of D2B2S.*

**D4A11.** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real symmetric packed matrix. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4a1 **Usage:** CALL D4A11(*10 parameters*) **Precision:** Single (Double: D4A11D) **Availability:** PC.

**D4A11D.** *Double precision version of D4A11.*

**D4A21.** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real square (general) matrix. Elementary transformations are used. Real and imaginary components of output values are returned in separate real arrays. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4a2 **Usage:** CALL D4A21(*10 parameters*) **Precision:** Single (Double: D4A21D) **Availability:** PC.

**D4A21D.** *Double precision version of D4A21.*

**D4A22D.** Calls a sequence of subroutines to find the eigenvalues and eigenvectors (if desired) of a real square (general) matrix. Orthogonal transformations are used. Real and imaginary components of output values are returned in separate real arrays. Estimates of the number of correct significant digits in the eigenvalue estimates can also be obtained. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4a2 **Usage:** CALL D4A22D(*10 parameters*) **Precision:** Double **Availability:** PC.

**D4C1A.** Balances a real matrix and isolates eigenvalues whenever possible. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c1a **Usage:** CALL D4C1A(LDA, N, A, LOW, IGH, SCALE) **Precision:** Single (Double: D4C1AD) **Availability:** PC.

**D4C1AD.** *Double precision version of D4C1A.*

**D4C1B.** Given a real general matrix, this subroutine reduces a submatrix situated in rows and columns LOW through IGH to upper Hessenberg form by stabilized elementary similarity transformations. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c1b2 **Usage:** CALL D4C1B(LDA, N, LOW, IGH, A, INT) **Precision:** Single (Double: D4C1BD) **Availability:** PC.

**D4C1BD.** *Double precision version of D4C1B.*

**D4C1C.** Reduces a real general matrix to upper Hessenberg form using orthogonal transformations. *Proprietary*

*Fortran subroutine in the Scientific Desk.* **Classes:** D4c1b2 **Usage:** CALL D4C1C(NM, N, LOW, IGH, A, ORT) **Precision:** Single (Double: D4C1CD) **Availability:** PC.

**D4C1CD.** *Double precision version of D4C1C.*

**D4C1D.** Reduces a real general symmetric matrix, stored as a one-dimensional array, to a symmetric tridiagonal matrix using orthogonal transformations. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c1b1 **Usage:** CALL D4C1D(N, NV, A, D, E, E2) **Also see:** D4C23 **Precision:** Single (Double: D4C1DD) **Availability:** PC.

**D4C1DD.** *Double precision version of D4C1D.*

**D4C23.** Accumulates the stabilized elementary similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by D4C1B. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c1b2 **Usage:** CALL D4C23(LDZ, N, LOW, IGH, A, INT, Z) **Also see:** D4C1B **Precision:** Single (Double: D4C23D) **Availability:** PC.

**D4C23D.** *Double precision version of D4C23.*

**D4C24.** Accumulates the orthogonal similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by D4C1C. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c1b2 **Usage:** CALL D4C24(NM, N, LOW, IGH, A, ORT, Z) **Also see:** D4C1C **Precision:** Single (Double: D4C24D) **Availability:** PC.

**D4C24D.** *Double precision version of D4C24.*

**D4C2A.** Finds the eigenvalues and eigenvectors of a symmetric tridiagonal matrix by the QL method. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4a5 **Usage:** CALL D4C2A(LDZ, N, D, E, Z, KE) **Precision:** Single (Double: D4C2AD) **Availability:** PC.

**D4C2AD.** *Double precision version of D4C2A.*

**D4C2B.** Finds the eigenvalues of a real upper Hessenberg matrix by the QR method. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c2b **Usage:** CALL D4C2B(LDH, N, LOW, IGH, H, WR, WI, KE) **Precision:** Single (Double: D4C2BD) **Availability:** PC.

**D4C2BD.** *Double precision version of D4C2B.*

**D4C2C.** Finds the eigenvalues and eigenvectors of a real upper Hessenberg matrix by the QR method. The eigenvectors of a real general matrix can also be found if D4C1B and D4C23 have been used to reduce this general matrix to upper Hessenberg form and to accumulate the similarity transformations. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c2b, D4a2 **Usage:** CALL D4C2C(*9 parameters*) **Precision:** Single (Double: D4C2CD) **Availability:** PC.

**D4C2CD.** *Double precision version of D4C2C.*

**D4C4.** Forms the eigenvectors of a real symmetric matrix by back transforming those of the corresponding symmetric tridiagonal matrix determined by D4C1D. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c4 **Usage:** CALL D4C4(LDZ, N, NV, A, M, Z) **Also see:** D4C1D **Precision:** Single (Double: D4C4D) **Availability:** PC.

**D4C4B.** Forms the eigenvectors of a real general matrix by back transforming those of the corresponding balanced matrix determined by D4C1A. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D4c4 **Usage:** CALL D4C4B(LDZ, N, LOW, IGH, SCALE, M, Z) **Also see:** D4C1A **Precision:** Single (Double: D4C4BD) **Availability:** PC.

**D4C4BD.** *Double precision version of D4C4B.*

**D4C4D.** *Double precision version of D4C4.*

**D4GEL.** To solve a linear system of equations, where the coefficient matrix is a D4-ordered block matrix. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a4 **Usage:** CALL D4GEL(*16 parameters*) **Precision:** Single **Availability:** 205 (vectorized)

**D6SVD.** Reduces a real n-by-p matrix X by orthogonal transformations U and V to diagonal form. The diagonal elements $S_i$ are the singular values of X. The columns of U are the corresponding left singular vectors, and the columns of V are the right singular vectors. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D6 **Usage:** CALL D6SVD(*13 parameters*) **Precision:** Single (Double: D6SVDD) **Availability:** PC.

**D6SVDD.** *Double precision version of D6SVD.*

**D9GLS.** Solves overdetermined linear least squares problems by QR factorizations using Householder transformations. It computes the least squares solution(s) to the problem AX=B where A is an m-by-n matrix, $m \geq n$ and B is the m-by-n matrix of right hand sides. User input bounds on the uncertainty in the elements of A are used to detect numerical rank deficiency. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D9a1 **Usage:** CALL D9GLS(*21 parameters*) **Precision:** Single (Double: D9GLSD) **Availability:** PC.

**D9GLSD.** *Double precision version of D9GLS.*

**D9GLU.** Solves underdetermined linear system of equations by LQ factorization using Householder transformations. It computes the minimal length solution(s) to the problem AX=B where A is an m-by-n matrix, $m \leq n$ and B is the m-by-n matrix of right hand sides. User input bounds on the uncertainty in the elements of A are used to detect numerical rank deficiency. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D9a1, D9a4 **Usage:**

CALL D9GLU(*21 parameters*) **Precision:** Single **Availability:** PC.
**DABALD.** *Double precision version of ABALD.*
**DABIBD.** *Double precision version of ABIBD.*
**DACF.** *Double precision version of ACF.*
**DACOS.** arccos(x), for double precision x. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = DACOS(D) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DACOS.** *Double precision version of ACOS.*
**DACOSH.** *Double precision version of ACOSH.*
**DACTBL.** *Double precision version of ACTBL.*
**DADD.** *Double precision version of SADD.*
**DAI.** *Double precision version of AI.*
**DAID.** *Double precision version of AID.*
**DAIDE.** *Double precision version of AIDE.*
**DAIE.** *Double precision version of AIE.*
**DALATN.** *Double precision version of ALATN.*
**DANEST.** *Double precision version of ANEST.*
**DANWAY.** *Double precision version of ANWAY.*
**DAONEC.** *Double precision version of AONEC.*
**DAONEW.** *Double precision version of AONEW.*
**DARBRB.** *Double precision version of ARBRB.*
**DARCOS.** *Double precision version of ARCOS.*
**DARMME.** *Double precision version of ARMME.*
**DARSIN.** *Double precision version of ARSIN.*
**DASIN.** arcsin(x), for double precision x. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = DASIN(D) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DASIN.** *Double precision version of ASIN.*
**DASINH.** *Double precision version of ASINH.*
**DASUM.** *Double precision version of SASUM.*
**DATAN.** arctan(x), for double precision x. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = DATAN(X) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DATAN.** *Double precision version of ATAN.*
**DATAN2.** arctan(x/y), for double precision x, y. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = DATAN2(X, Y) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DATAN2.** *Double precision version of ATAN2.*
**DATANH.** *Double precision version of ATANH.*
**DATWOB.** *Double precision version of ATWOB.*
**DAWS.** Dawson's integral. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C8c **Usage:** R = DAWS(X) **Precision:** Single (Double: DDAWS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DAWS.** Dawson's integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8c **Usage:** R = DAWS(X) **Precision:** Single (Double: DDAWS) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DAXPY.** *Double precision version of SAXPY.*
**DB2INK.** *Double precision version of B2INK.*
**DB2VAL.** *Double precision version of B2VAL.*
**DB3INK.** *Double precision version of B3INK.*
**DB3VAL.** *Double precision version of B3VAL.*
**DBCEVL.** Bicubic spline mixed partial derivative evaluator. *Proprietary Fortran subroutine in IMSL.* **Classes:** E3a2, K6a2 **Usage:** CALL DBCEVL (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DBCLSF.** *Double precision version of BCLSF.*
**DBCLSJ.** *Double precision version of BCLSJ.*
**DBCOAH.** *Double precision version of BCOAH.*
**DBCODH.** *Double precision version of BCODH.*
**DBCONF.** *Double precision version of BCONF.*
**DBCONG.** *Double precision version of BCONG.*
**DBCPOL.** *Double precision version of BCPOL.*
**DBCQDU.** Bicubic spline quadrature. *Proprietary Fortran subroutine in IMSL.* **Classes:** H2b1b2 **Usage:** CALL DBCQDU (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DBCTR.** *Double precision version of BCTR.*
**DBEI0.** *Double precision version of BEI0.*
**DBEI1.** *Double precision version of BEI1.*

**DBEIP0.** *Double precision version of BEIP0.*
**DBER0.** *Double precision version of BER0.*
**DBER1.** *Double precision version of BER1.*
**DBERP0.** *Double precision version of BERP0.*
**DBESCI.** *Double precision version of BESCI.*
**DBESCJ.** *Double precision version of BESCJ.*
**DBESI0.** *Double precision version of BESI0.*
**DBESI1.** *Double precision version of BESI1.*
**DBESJ.** *Double precision version of BESJ.*
**DBESJ0.** *Double precision version of BESJ0.*
**DBESJ1.** *Double precision version of BESJ1.*
**DBESK0.** *Double precision version of BESK0.*
**DBESK1.** *Double precision version of BESK1.*
**DBESKS.** *Double precision version of BESKS.*
**DBESRI.** *Double precision version of BESRI.*
**DBESRJ.** *Double precision version of BESRJ.*
**DBESY0.** *Double precision version of BESY0.*
**DBESY1.** *Double precision version of BESY1.*
**DBETA.** *Double precision version of BETA.*
**DBETAI.** *Double precision version of BETAI.*
**DBETDF.** *Double precision version of BETDF.*
**DBETIN.** *Double precision version of BETIN.*
**DBFQAD.** *Double precision version of BFQAD.*
**DBHAKV.** *Double precision version of BHAKV.*
**DBI.** *Double precision version of BI.*
**DBID.** *Double precision version of BID.*
**DBIDE.** *Double precision version of BIDE.*
**DBIE.** *Double precision version of BIE.*
**DBINDF.** *Double precision version of BINDF.*
**DBINES.** *Double precision version of BINES.*
**DBINOM.** *Double precision version of BINOM.*
**DBINPR.** *Double precision version of BINPR.*
**DBINT4.** *Double precision version of BINT4.*
**DBINTK.** *Double precision version of BINTK.*
**DBLIN.** Numerical integration of a function of two variables. *Proprietary Fortran function in IMSL.* **Classes:** H2b1a1 **Usage:** R = DBLIN(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DBLINF.** *Double precision version of BLINF.*
**DBNRDF.** *Double precision version of BNRDF.*
**DBOCLS.** Solves the general linearly constrained linear least squares problems. *Portable Fortran subroutine in CMLIB (BOCLS sublibrary).* **Classes:** D9b1 **Usage:** CALL DBOCLS(*15 parameters*) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DBOLS.** Solves linear least squares problems with simple bounds on the variables. *Portable Fortran subroutine in CMLIB (BOCLS sublibrary).* **Classes:** D9b1 **Usage:** CALL DBOLS(*17 parameters*) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DBOXP.** *Double precision version of BOXP.*
**DBQUAD.** *Double precision version of BQUAD.*
**DBS2DR.** *Double precision version of BS2DR.*
**DBS2IG.** *Double precision version of BS2IG.*
**DBS2IN.** *Double precision version of BS2IN.*
**DBS2VL.** *Double precision version of BS2VL.*
**DBS3DR.** *Double precision version of BS3DR.*
**DBS3IG.** *Double precision version of BS3IG.*
**DBS3IN.** *Double precision version of BS3IN.*
**DBS3VL.** *Double precision version of BS3VL.*
**DBSCAT.** *Double precision version of BSCAT.*
**DBSCPP.** *Double precision version of BSCPP.*
**DBSDER.** *Double precision version of BSDER.*
**DBSI0.** *Double precision version of BSI0.*
**DBSI0E.** *Double precision version of BESI0E.*
**DBSI0E.** *Double precision version of BSI0E.*

**DBSI1.** *Double precision version of BSI1.*
**DBSI1E.** *Double precision version of BESI1E.*
**DBSI1E.** *Double precision version of BSI1E.*
**DBSIES.** *Double precision version of BSIES.*
**DBSINS.** *Double precision version of BSINS.*
**DBSINT.** *Double precision version of BSINT.*
**DBSIS.** *Double precision version of BSIS.*
**DBSITG.** *Double precision version of BSITG.*
**DBSJ0.** *Double precision version of BSJ0.*
**DBSJ1.** *Double precision version of BSJ1.*
**DBSJNS.** *Double precision version of BSJNS.*
**DBSJS.** *Double precision version of BSJS.*
**DBSK0.** *Double precision version of BSK0.*
**DBSK0E.** *Double precision version of BESK0E.*
**DBSK0E.** *Double precision version of BSK0E.*
**DBSK1.** *Double precision version of BSK1.*
**DBSK1E.** *Double precision version of BESK1E.*
**DBSK1E.** *Double precision version of BSK1E.*
**DBSKES.** *Double precision version of BESKES.*
**DBSKES.** *Double precision version of BSKES.*
**DBSKS.** *Double precision version of BSKS.*
**DBSLS2.** *Double precision version of BSLS2.*
**DBSLSQ.** *Double precision version of BSLSQ.*
**DBSNAK.** *Double precision version of BSNAK.*
**DBSOPK.** *Double precision version of BSOPK.*
**DBSPBS.** *Double precision version of BSPBS.*
**DBSPDR.** *Double precision version of BSPDR.*
**DBSPEV.** *Double precision version of BSPEV.*
**DBSPL1.** *Double precision version of BSPL1.*
**DBSPLD.** *Double precision version of BSPLD.*
**DBSPLI.** *Double precision version of BSPLI.*
**DBSPLN.** *Double precision version of BSPLN.*
**DBSPPP.** *Double precision version of BSPPP.*
**DBSPVD.** *Double precision version of BSPVD.*
**DBSPVN.** *Double precision version of BSPVN.*
**DBSQAD.** *Double precision version of BSQAD.*
**DBSVAL.** *Double precision version of BSVAL.*
**DBSVLS.** *Double precision version of BSVLS.*
**DBSY0.** *Double precision version of BSY0.*
**DBSY1.** *Double precision version of BSY1.*
**DBSYS.** *Double precision version of BSYS.*
**DBURAM.** *Double precision version of BURAM.*
**DBURM1.** *Double precision version of BURM1.*
**DBVALU.** *Double precision version of BVALU.*
**DBVPFD.** *Double precision version of BVPFD.*
**DBVPMS.** *Double precision version of BVPMS.*
**DCADRE.** Numerical integration of a function using cautious adaptive Romberg extrapolation. *Proprietary Fortran function in IMSL.* **Classes:** H2a1a1 **Usage:** R = DCADRE(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DCBRT.** *Double precision version of CBRT.*
**DCCF.** *Double precision version of CCF.*
**DCDF2P.** *Double precision version of CDF2P.*
**DCDFP.** *Double precision version of CDFP.*
**DCDGRD.** *Double precision version of CDGRD.*
**DCDIST.** *Double precision version of CDIST.*
**DCDOT.** Computes a complex precision dot product using double precision accumulation. *Portable Fortran subroutine in CMLIB (XBLAS sublibrary).* **Classes:** D1a4 **Usage:** CALL DCDOT(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DCEIL.** *Double precision version of CEIL.*
**DCFFTB.** *Double precision version of CFFTB.*

**DCFFTF.** *Double precision version of CFFTF.*
**DCFT2D.** *Double precision version of CFFT2D.*
**DCHDC.** *Double precision version of SCHDC.*
**DCHDD.** *Double precision version of SCHDD.*
**DCHEX.** *Double precision version of SCHEX.*
**DCHFAC.** *Double precision version of CHFAC.*
**DCHFDV.** *Double precision version of CHFDV.*
**DCHFEV.** *Double precision version of CHFEV.*
**DCHGRD.** *Double precision version of CHGRD.*
**DCHHES.** *Double precision version of CHHES.*
**DCHI.** *Double precision version of CHI.*
**DCHIDF.** *Double precision version of CHIDF.*
**DCHIGF.** *Double precision version of CHIGF.*
**DCHIIN.** *Double precision version of CHIIN.*
**DCHJAC.** *Double precision version of CHJAC.*
**DCHU.** *Double precision version of CHU.*
**DCHUD.** *Double precision version of SCHUD.*
**DCI.** *Double precision version of CI.*
**DCIDMS.** *Double precision version of CIDMS.*
**DCIN.** *Double precision version of CIN.*
**DCINH.** *Double precision version of CINH.*
**DCKLS.** Check user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values. *Portable Fortran subroutine in STARPAC.* **Classes:** L8i **Usage:** CALL DCKLS (*9 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).
**DCKLSC.** Check user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8i **Usage:** CALL DCKLSC (*14 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).
**DCLINK.** *Double precision version of CLINK.*
**DCLINQ.** *Double precision version of CLINQ.*
**DCLST2.** *Double precision version of CLST2.*
**DCNCRD.** *Double precision version of CNCRD.*
**DCONST.** *Double precision version of CONST.*
**DCOPY.** *Double precision version of SCOPY.*
**DCORVC.** *Double precision version of CORVC.*
**DCOS.** *Double precision version of COS.*
**DCOSDG.** *Double precision version of COSDG.*
**DCOSH.** *Double precision version of COSH.*
**DCOT.** *Double precision version of COT.*
**DCPOLY.** *Double precision version of CPOLY.*
**DCRBRB.** *Double precision version of CRBRB.*
**DCRBRG.** *Double precision version of CRBRG.*
**DCRGRB.** *Double precision version of CRGRB.*
**DCRGRG.** *Double precision version of CRGRG.*
**DCSAKM.** *Double precision version of CSAKM.*
**DCSBRB.** *Double precision version of CSBRB.*
**DCSCON.** *Double precision version of CSCON.*
**DCSDEC.** *Double precision version of CSDEC.*
**DCSDER.** *Double precision version of CSDER.*
**DCSEVL.** *Double precision version of CSEVL.*
**DCSEVU.** Cubic spline first and second derivative evaluator. *Proprietary Fortran subroutine in IMSL.* **Classes:** E3a2, K6a2 **Usage:** CALL DCSEVU (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DCSFRG.** *Double precision version of CSFRG.*
**DCSHER.** *Double precision version of CSHER.*
**DCSINT.** *Double precision version of CSINT.*
**DCSITG.** *Double precision version of CSITG.*
**DCSNDF.** *Double precision version of CSNDF.*
**DCSPDI.** *Double precision version of CSPDI.*

**DCSPER.** *Double precision version of CSPER.*
**DCSPFE.** *Double precision version of CSPFE.*
**DCSPFI.** *Double precision version of CSPFI.*
**DCSPIN.** *Double precision version of CSPIN.*
**DCSPQU.** *Double precision version of CSPQU.*
**DCSQDU.** Cubic spline quadrature. *Proprietary Fortran subroutine in IMSL.* **Classes:** E3a3, H2a2b1, K6a3 **Usage:** CALL DCSQDU (X, Y, NX, C, IC, A, B, Q, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DCSSCV.** *Double precision version of CSSCV.*
**DCSSED.** *Double precision version of CSSED.*
**DCSSMH.** *Double precision version of CSSMH.*
**DCSSWP.** *Double precision version of CSSWP.*
**DCSTAT.** *Double precision version of CSTAT.*
**DCSVAL.** *Double precision version of CSVAL.*
**DCTCHI.** *Double precision version of CTCHI.*
**DCTPRB.** *Double precision version of CTPRB.*
**DCTRHO.** *Double precision version of CTRHO.*
**DCTRST.** *Double precision version of CTRST.*
**DCTTWO.** *Double precision version of CTTWO.*
**DCUBE.** Perform a triplets test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1d **Usage:** CALL DCUBE (*10 parameters*) **Precision:** Single (Double: DDCUBE) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DCUNIT.** *Double precision version of CUNIT.*
**DDASSL.** *Double precision version of SDASSL.*
**DDAWS.** *Double precision version of DAWS.*
**DDCUBE.** *Double precision version of DCUBE.*
**DDERIV.** *Double precision version of DERIV.*
**DDESKN.** *Double precision version of DESKN.*
**DDESPL.** *Double precision version of DESPL.*
**DDESPT.** *Double precision version of DESPT.*
**DDIFF.** *Double precision version of DIFF.*
**DDIRIC.** *Double precision version of DIRIC.*
**DDISL1.** *Double precision version of DISL1.*
**DDISL2.** *Double precision version of DISL2.*
**DDISLI.** *Double precision version of DISLI.*
**DDL2SF.** *Double precision version of DL2SF.*
**DDL2SW.** *Double precision version of DL2SW.*
**DDLPRS.** *Double precision version of DLPRS.*
**DDMSCR.** *Double precision version of DMSCR.*
**DDOT.** *Double precision version of SDOT.*
**DDRIV1.** *Double precision version of SDRIV1.*
**DDRIV2.** *Double precision version of SDRIV2.*
**DDRIV3.** *Double precision version of SDRIV3.*
**DDSCRM.** *Double precision version of DSCRM.*
**DDSQAR.** *Double precision version of DSQAR.*
**DE1.** *Double precision version of E1.*
**DEA.** *Double precision version of EA.*
**DEABM.** Solves a system of first order ordinary differential equations with arbitrary initial conditions by a predictor-corrector method. *Portable Fortran subroutine in CMLIB (DEPAC sublibrary).* **Classes:** I1a1b **Usage:** CALL DEABM(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DEBDF.** Solves a system of first order stiff ordinary differential equations with arbitrary initial conditions by Gear's method. *Portable Fortran subroutine in CMLIB (DEPAC sublibrary).* **Classes:** I1a2 **Usage:** CALL DEBDF(F, NEQ, T, Y, TOUT, INFO, RTOL, ATOL, IDID, RWORK, LRW, IWORK, LIW, RPAR, IPAR, JAC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DECOMP.** Decomposes a not-necessarily convex polygon into a sequence of convex polygons. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL DECOMP(*8 parameters*) **Precision:** Single **Availability:** 855NOS.
**DECOMP.** Decomposes a matrix using Gaussian elimination into a lower and upper triangular factorization and estimates the condition of the matrix. This routine is usually used in conjunction with SOLVE which takes the LU factorization produced by DECOMP to actually solve a system of linear equations. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** D2a1 **Precision:** Single **Availability:** 855NOS.

**DEEBSF.** *Double precision version of EEBSF.*
**DEEBSI.** *Double precision version of EEBSI.*
**DEESFF.** *Double precision version of EESFF.*
**DEESFI.** *Double precision version of EESFI.*
**DEFINE.** Defines a vector of constants by setting all of the elements in the single precision vector X equal to XNEW. *Fortran subroutine in DATAPAC.* **Classes:** D1a1 **Usage:** CALL DEFINE(X, N, XNEW) **Precision:** Single **Availability:** 855NOS, 840NOS.
**DEFINT.** Uses double exponential transformation of Mori to compute definite integral automatically to user specified accuracy. *Portable Fortran software in JCAM library.* **Classes:** H2a1a1 **Usage:** CALL DEFINT(FUNC, A, B, EPS, L, V) **Also see:** FIAB **Precision:** Double **Availability:** 855NOS (In source form only.)
**DEHINT.** Uses double exponential transformation of Mori to compute semi-infinite range integral automatically to user specified accuracy. *Portable Fortran software in JCAM library.* **Classes:** H2a3a1 **Usage:** CALL DEHINT(FUNC, A, EPS, V) **Also see:** HIAB **Precision:** Double **Availability:** 855NOS (In source form only.)
**DEI.** *Double precision version of EI.*
**DEIGEN.** *Double precision version of EIGEN.*
**DELE.** *Double precision version of ELE.*
**DELETE.** Deletes a boundary edge from a triangulation of a set of points in the plane. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL DELETE (*6 parameters*) **Also see:** TRMESH **Precision:** Single **Availability:** 855NOS.
**DELETE.** Deletes all observations in the vector X which are inside the interval [XMIN, XMAX]. *Fortran subroutine in DATAPAC.* **Classes:** L2c **Usage:** CALL DELETE(X, N, XMIN, XMAX, NEWN) **Precision:** Single **Availability:** 855NOS, 840NOS.
**DELK.** *Double precision version of ELK.*
**DELRC.** *Double precision version of ELRC.*
**DELRD.** *Double precision version of ELRD.*
**DELRF.** *Double precision version of ELRF.*
**DELRJ.** *Double precision version of ELRJ.*
**DEMOD.** Performs a complex demodulation on the data in the input vector X at the input demodulation frequency F. *Fortran subroutine in DATAPAC.* **Classes:** L10a3b **Usage:** CALL DEMOD(X, N, F) **Precision:** Single **Availability:** 855NOS, 840NOS.
**DEMOD.** Compute and plot the results of a complex demodulation of the input series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3b **Usage:** CALL DEMOD (Y, N, FD, FC, K, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).
**DEMODS.** Compute and optionally plot the results of a complex demodulation of the input series; return amplitude and phase functions of demodulated series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3b **Usage:** CALL DEMODS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).
**DENE.** *Double precision version of ENE.*
**DENOS.** *Double precision version of ENOS.*
**DEPISB.** *Double precision version of EPISB.*
**DEPISF.** *Double precision version of EPISF.*
**DERF.** *Double precision version of ERF.*
**DERFC.** *Double precision version of ERFC.*
**DERFCE.** *Double precision version of ERFCE.*
**DERFCI.** *Double precision version of ERFCI.*
**DERFI.** *Double precision version of ERFI.*
**DERIV.** Compute the first, second or third derivative of a user-supplied function. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** H1 **Usage:** R = DERIV(FCN, KORDER, X, BGSTEP, TOL) **Precision:** Single (Double: DDERIV) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DERKF.** Solves a system of first order ordinary differential equations with arbitrary initial conditions by a Runge-Kutta method. *Portable Fortran subroutine in CMLIB (DEPAC sublibrary).* **Classes:** I1a1a **Usage:** CALL DERKF(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DESKN.** Perform nonparametric probability density function estimation by the kernel method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1b2 **Usage:** CALL DESKN (*9 parameters*) **Precision:** Single (Double: DDESKN) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DESPL.** Perform nonparametric probability density function estimation by the penalized likelihood method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1b2 **Usage:** CALL DESPL (*11 parameters*) **Precision:** Single (Double: DDESPL) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DESPT.** Estimate a probability density function at specified points using linear or cubic interpolation. *Proprietary*

*Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1b2 **Usage:** CALL DESPT (*7 parameters*) **Precision:** Single (Double: DDESPT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**DEVASB.** *Double precision version of EVASB.*

**DEVASF.** *Double precision version of EVASF.*

**DEVBSB.** *Double precision version of EVBSB.*

**DEVBSF.** *Double precision version of EVBSF.*

**DEVCSB.** *Double precision version of EVCSB.*

**DEVCSF.** *Double precision version of EVCSF.*

**DEVESB.** *Double precision version of EVESB.*

**DEVESF.** *Double precision version of EVESF.*

**DEVFSB.** *Double precision version of EVFSB.*

**DEVFSF.** *Double precision version of EVFSF.*

**DEVLSB.** *Double precision version of EVLSB.*

**DEVLSF.** *Double precision version of EVLSF.*

**DEXCDF.** Computes the cumulative distribution function value for the double exponential (Laplace) distribution with mean 0. *Fortran subroutine in DATAPAC.* **Classes:** L5a1d **Usage:** CALL DEXCDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DEXINT.** *Double precision version of EXINT.*

**DEXP.** *Double precision version of EXP.*

**DEXPDF.** Computes the probability density function value for the double exponential (Laplace) distribution with mean 0. *Fortran subroutine in DATAPAC.* **Classes:** L5a1d **Usage:** CALL DEXPDF(X, PDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DEXPLT.** Generates a double exponential (Laplace) probability plot with mean 0 and standard deviation $\sqrt{2}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2d **Usage:** CALL DEXPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DEXPPF.** Computes the percent point function value for the double exponential (Laplace) distribution with mean 0. *Fortran subroutine in DATAPAC.* **Classes:** L5a2d **Usage:** CALL DEXPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DEXPRL.** *Double precision version of EXPREL.*

**DEXPRL.** *Double precision version of EXPRL.*

**DEXRAN.** Generates a random sample of size N from the double exponential (Laplace) distribution with mean 0 and standard deviation $\sqrt{2}$. *Fortran subroutine in DATAPAC.* **Classes:** L6a4 **Usage:** CALL DEXRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DEXSF.** Computes the sparsity function value for the double exponential (Laplace) distribution with mean 0 and standard deviation $\sqrt{2}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a2d **Usage:** CALL DEXSF(P, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DEZFTB.** *Double precision version of EZFFTB.*

**DEZFTF.** *Double precision version of EZFFTF.*

**DFAC.** *Double precision version of FAC.*

**DFACTR.** *Double precision version of FACTR.*

**DFCOEF.** *Double precision version of FCOEF.*

**DFCOST.** *Double precision version of FCOST.*

**DFDF.** *Double precision version of FDF.*

**DFDGRD.** *Double precision version of FDGRD.*

**DFDHES.** *Double precision version of FDHES.*

**DFDJAC.** *Double precision version of FDJAC.*

**DFDOBL.** *Double precision version of FDOBL.*

**DFEJER.** *Double precision version of FEJER.*

**DFFT.** *Double precision version of FFT.*

**DFFTC.** *Double precision version of FFTC.*

**DFFTCI.** *Double precision version of FFTCI.*

**DFFTR.** *Double precision version of FFTR.*

**DFFTRB.** *Double precision version of FFTRB.*

**DFFTRF.** *Double precision version of FFTRF.*

**DFFTRI.** *Double precision version of FFTRI.*

**DFHARR.** *Double precision version of FHARR.*

**DFIMAG.** *Double precision version of FIMAG.*

**DFIN.** *Double precision version of FIN.*

**DFLR.** *Double precision version of FLR.*

**DFMIN.** *Double precision version of FMIN.*

**DFNLSQ.** *Double precision version of FNLSQ.*
**DFOPCS.** *Double precision version of FOPCS.*
**DFPRMX.** *Double precision version of FPRMX.*
**DFPS2H.** *Double precision version of FPS2H.*
**DFPS3H.** *Double precision version of FPS3H.*
**DFQRUL.** *Double precision version of FQRUL.*
**DFRDMN.** *Double precision version of FRDMN.*
**DFREQ.** *Double precision version of FREQ.*
**DFRESI.** *Double precision version of FRESI.*
**DFROTA.** *Double precision version of FROTA.*
**DFRVAR.** *Double precision version of FRVAR.*
**DFSCOR.** *Double precision version of FSCOR.*
**DFSINT.** *Double precision version of FSINT.*
**DFZERO.** *Double precision version of FZERO.*
**DGAMDF.** *Double precision version of GAMDF.*
**DGAMI.** *Double precision version of GAMI.*
**DGAMIC.** *Double precision version of GAMIC.*
**DGAMIT.** *Double precision version of GAMIT.*
**DGAMMA.** *Double precision version of GAMMA.*
**DGAMR.** *Double precision version of GAMR.*
**DGAUSQ.** *Double precision version of GAUSQ.*
**DGBCO.** *Double precision version of SGBCO.*
**DGBDI.** *Double precision version of SGBDI.*
**DGBFA.** *Double precision version of SGBFA.*
**DGBSL.** *Double precision version of SGBSL.*
**DGCDF.** *Double precision version of GCDF.*
**DGCSCP.** *Double precision version of GCSCP.*
**DGDATA.** *Double precision version of GDATA.*
**DGDHES.** *Double precision version of GDHES.*
**DGEAR.** Differential equation solver – variable order Adams predictor-corrector method or Gear's method. *Proprietary Fortran subroutine in IMSL.* **Classes:** I1a2, I1a1b **Usage:** CALL DGEAR (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.
**DGECO.** *Double precision version of SGECO.*
**DGEDI.** *Double precision version of SGEDI.*
**DGEFA.** *Double precision version of SGEFA.*
**DGEFS.** *Double precision version of SGEFS.*
**DGESL.** *Double precision version of SGESL.*
**DGGUES.** *Double precision version of GGUES.*
**DGPISP.** *Double precision version of GPISP.*
**DGQ0IN.** *Double precision version of GQ0IN.*
**DGQM11.** *Double precision version of GQM11.*
**DGQRCF.** *Double precision version of GQRCF.*
**DGQRUL.** *Double precision version of GQRUL.*
**DGRPES.** *Double precision version of GRPES.*
**DGTSL.** *Double precision version of SGTSL.*
**DGVCSP.** *Double precision version of GVCSP.*
**DGVLSP.** *Double precision version of GVLSP.*
**DHAZEZ.** *Double precision version of HAZEZ.*
**DHAZRD.** *Double precision version of HAZRD.*
**DHAZST.** *Double precision version of HAZST.*
**DHHSTP.** *Double precision version of HHSTP.*
**DHOUAP.** *Double precision version of SHOUAP.*
**DHOUTR.** *Double precision version of SHOUTR.*
**DHPROD.** *Double precision version of SHPROD.*
**DHRRRR.** *Double precision version of HRRRR.*
**DHUMSL.** *Double precision version of HUMSL.*
**DHYPDF.** *Double precision version of HYPDF.*
**DHYPOT.** *Double precision version of HYPOT.*
**DHYPPR.** *Double precision version of HYPPR.*
**DIF.** Perform first-difference filter operation; return differenced series. (No printed output.) *Portable Fortran subrou-*

*tine in STARPAC.* **Classes:** L10a1c1 **Usage:** CALL DIF (Y, N, YF, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**DIFC.** Perform user-specified difference filter operation; return differenced series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c3 **Usage:** CALL DIFC (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**DIFF.** A subroutine that computes the first, second or third derivative of a real function of a single real variable, where the function is defined by a Fortran FUNCTION subprogram. The method used is Neville's process of extrapolating from a sequence of interpolating polynomials with interpolating points distributed symmetrically about $x_0$ or, if this is not possible, to one side of $x_0$. *Portable Fortran software in DERIV library.* **Classes:** H1 **Usage:** CALL DIFF(*10 parameters*) **Precision:** Single **Availability:** 855NOS (In source form only.)

**DIFF.** Difference a time series. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a1c1 **Usage:** CALL DIFF (*10 parameters*) **Precision:** Single (Double: DDIFF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**DIFFERENCES.** Computes all differences X – Y for each value X in one vector and each value Y in a second vector (useful for nonparametric tests and confidence intervals). *Command in MINITAB proprietary interactive system.* **Classes:** L2a **Usage:** DIFFerences between values in C and C, put into C [put indices into C and C] **Precision:** Single **Availability:** 855NOS.

**DIFFERENCES.** Computes differences between observations at a specified lag in a time series. *Command in MINITAB proprietary interactive system.* **Classes:** L10a1c1 **Usage:** DIFFerences [of lag K] for data in C, put into C **Precision:** Single **Availability:** 855NOS.

**DIFM.** Perform first-difference filter operation on series with missing data; return differenced series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c1 **Usage:** CALL DIFM (Y, YMISS, N, YF, YFMISS, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**DIFMC.** Perform user-specified difference filter operation on series with missing data; return differenced series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c3 **Usage:** CALL DIFMC (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**DINCLD.** *Double precision version of INCLD.*

**DINT.** *Double precision version of AINT.*

**DINTRV.** *Double precision version of INTRV.*

**DIRIC.** Compute the Dirichlet kernel. *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** L10a3a6 **Usage:** R = DIRIC(M, RANGLE, EPS) **Precision:** Single (Double: DDIRIC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**DIRNSE.** *Double precision version of IRNSE.*

**DISCR2.** Discretizes the data in the vector X into NUMCLA classes (using midpoint). *Fortran subroutine in DATAPAC.* **Classes:** L2a **Usage:** CALL DISCR2(X, N, NUMCLA, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DISCR3.** Discretizes the data in the vector X into NUMCLA classes (using class numbers). *Fortran subroutine in DATAPAC.* **Classes:** L2a **Usage:** CALL DISCR3(X, N, NUMCLA, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DISCRE.** Discretizes the data of the vector X according to class width. *Fortran subroutine in DATAPAC.* **Classes:** L2a **Usage:** CALL DISCRE(X, N, XMIN, XDEL, XMAX, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**DISCRIM.** Computes linear or quadratic discriminant functions for classifying observations into two or more groups. The distribution within each group should be approximately multivariate normal. The classification criterion can be based on either the individual within-group covariance matrices or the pooled covariance matrix. Options: homogeneity of the within-group covariance test, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L12 **Precision:** Single **Availability:** CAMVAX.

**DISCRIMINANT.** Performs multiple discriminant analysis. Options: statistical analyses, classifications, five stepwise selection methods (including minimizing Wilks' lambda, maximizing Mahalanobis' distance, maximizing Rao's V), varimax rotation, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L12 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**DISL1.** Compute the 1-norm distance between two points. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D1a3a, D1a11 **Usage:** R = DISL1(N, X, INCX, Y, INCY) **Precision:** Single (Double: DDISL1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**DISL2.** Compute the Euclidean (2-norm) distance between two points. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D1a3b, D1a11 **Usage:** R = DISL2(N, X, INCX, Y, INCY) **Precision:** Single (Double: DDISL2) **Availability:** 855NOS, 855VE, 205, 840NOS.

**DISLI.** Compute the infinity norm distance between two points. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D1a3c, D1a11 **Usage:** R = DISLI(N, X, INCX, Y, INCY) **Precision:** Single (Double:

DDISLI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**DISSPLA.** A large proprietary Fortran subprogram library for producing publication-quality plots of two- and three-dimensional data. Capabilities include drawing of axes, grids, and labels, curve drawing, area shading, interpolation and smoothing, multiple plots per page, surface view plotting, contouring, scaling, projection and rotation, and elaborate character fonts. Many output devices are supported; device-independent output is also supported. **Classes:** Q **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS. *On 855NOS, 855VE:* Output on Zeta plotter, Tektronix devices, QMS printers.

**DIST.** Computes the Euclidean distance between two cases or two variables. *Portable Fortran function in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** SDISTAN = DIST(M, X, Y, N, P) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**DITTO.** Partitions categorical data into clusters by maximizing the matches between cases in a cluster and the cluster mode. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL DITTO(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**DIVPAG.** *Double precision version of IVPAG.*

**DIVPBS.** *Double precision version of IVPBS.*

**DIVPRK.** *Double precision version of IVPRK.*

**DJCGRC.** *Double precision version of JCGRC.*

**DKALMN.** *Double precision version of KALMN.*

**DKAPMR.** *Double precision version of KAPMR.*

**DKEI0.** *Double precision version of AKEI0.*

**DKEI1.** *Double precision version of AKEI1.*

**DKEIP0.** *Double precision version of AKEIP0.*

**DKENDL.** *Double precision version of KENDL.*

**DKENDP.** *Double precision version of KENDP.*

**DKER0.** *Double precision version of AKER0.*

**DKER1.** *Double precision version of AKER1.*

**DKERP0.** *Double precision version of AKERP0.*

**DKMEAN.** *Double precision version of KMEAN.*

**DKRSKL.** *Double precision version of KRSKL.*

**DKS1DF.** *Double precision version of AKS1DF.*

**DKS2DF.** *Double precision version of AKS2DF.*

**DKSONE.** *Double precision version of KSONE.*

**DKSTWO.** *Double precision version of KSTWO.*

**DKTRND.** *Double precision version of KTRND.*

**DL2SF.** Fits discrete data with a B-spline of order K, by least squares. *Proprietary Fortran subroutine in PORT.* **Classes:** K1a1a1 **Usage:** CALL DL2SF (X, Y, NXY, K, T, NT, A) **Also see:** DSPLNE DSPLNI DSPLN1 DSPLN2 DSPLND **Precision:** Single (Double: DDL2SF) **Availability:** 855NOS, 205.

**DL2SFF.** *Double precision version of L2SFF.*

**DL2SFH.** *Double precision version of L2SFH.*

**DL2SW.** Fits discrete data with a B-spline of order k, by weighted least squares. *Proprietary Fortran subroutine in PORT.* **Classes:** K1a1a1 **Usage:** CALL DL2SW (X, Y, NXY, W, K, T, NT, A) **Also see:** SPLNE SPLND SPLNI SPLN1 SPLN2 **Precision:** Single (Double: DDL2SW) **Availability:** 855NOS, 205.

**DLBETA.** *Double precision version of ALBETA.*

**DLCHRG.** *Double precision version of LCHRG.*

**DLDNCH.** *Double precision version of LDNCH.*

**DLETTR.** *Double precision version of LETTR.*

**DLFCDS.** *Double precision version of LFCDS.*

**DLFCQS.** *Double precision version of LFCQS.*

**DLFCRB.** *Double precision version of LFCRB.*

**DLFCRG.** *Double precision version of LFCRG.*

**DLFCRT.** *Double precision version of LFCRT.*

**DLFCSF.** *Double precision version of LFCSF.*

**DLFDDS.** *Double precision version of LFDDS.*

**DLFDQS.** *Double precision version of LFDQS.*

**DLFDRB.** *Double precision version of LFDRB.*

**DLFDRG.** *Double precision version of LFDRG.*

**DLFDRT.** *Double precision version of LFDRT.*

**DLFDSF.** *Double precision version of LFDSF.*

**DLFIDS.** *Double precision version of LFIDS.*

**DLFIQS.** *Double precision version of LFIQS.*

**DLFIRB.** *Double precision version of LFIRB.*
**DLFIRG.** *Double precision version of LFIRG.*
**DLFISF.** *Double precision version of LFISF.*
**DLFSDS.** *Double precision version of LFSDS.*
**DLFSQS.** *Double precision version of LFSQS.*
**DLFSRB.** *Double precision version of LFSRB.*
**DLFSRG.** *Double precision version of LFSRG.*
**DLFSSF.** *Double precision version of LFSSF.*
**DLFTDS.** *Double precision version of LFTDS.*
**DLFTQS.** *Double precision version of LFTQS.*
**DLFTRB.** *Double precision version of LFTRB.*
**DLFTRG.** *Double precision version of LFTRG.*
**DLFTSF.** *Double precision version of LFTSF.*
**DLGAMS.** *Double precision version of ALGAMS.*
**DLI.** *Double precision version of ALI.*
**DLINDS.** *Double precision version of LINDS.*
**DLINEQ.** *Double precision version of LINEQ.*
**DLINFS.** *Double precision version of LINFS.*
**DLINRG.** *Double precision version of LINRG.*
**DLINRT.** *Double precision version of LINRT.*
**DLNGAM.** *Double precision version of ALNGAM.*
**DLNREL.** *Double precision version of ALNREL.*
**DLOG.** *Double precision version of ALOG.*
**DLOG10.** *Double precision version of ALOG10.*
**DLPRS.** Solve a linear programming problem via the revised simplex algorithm. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2a1 **Usage:** CALL DLPRS (*13 parameters*) **Precision:** Single (Double: DDLPRS) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DLQERR.** *Double precision version of LQERR.*
**DLQRRR.** *Double precision version of LQRRR.*
**DLQRSL.** *Double precision version of LQRSL.*
**DLSADS.** *Double precision version of LSADS.*
**DLSAQS.** *Double precision version of LSAQS.*
**DLSARB.** *Double precision version of LSARB.*
**DLSARG.** *Double precision version of LSARG.*
**DLSASF.** *Double precision version of LSASF.*
**DLSBRR.** *Double precision version of LSBRR.*
**DLSGRR.** *Double precision version of LSGRR.*
**DLSLDS.** *Double precision version of LSLDS.*
**DLSLQS.** *Double precision version of LSLQS.*
**DLSLRB.** *Double precision version of LSLRB.*
**DLSLRG.** *Double precision version of LSLRG.*
**DLSLRT.** *Double precision version of LSLRT.*
**DLSLSF.** *Double precision version of LSLSF.*
**DLSLTO.** *Double precision version of LSLTO.*
**DLSQRR.** *Double precision version of LSQRR.*
**DLSTSQ.** *Double precision version of LSTSQ.*
**DLSVRR.** *Double precision version of LSVRR.*
**DLUMB.** *Double precision version of LUMB.*
**DLUMD.** *Double precision version of LUMD.*
**DLUPCH.** *Double precision version of LUPCH.*
**DLUPQR.** *Double precision version of LUPQR.*
**DMACH.** *Double precision version of AMACH.*
**DMAMME.** *Double precision version of MAMME.*
**DMCCF.** *Double precision version of MCCF.*
**DMEDPL.** *Double precision version of MEDPL.*
**DMILLR.** *Double precision version of AMILLR.*
**DMKFL.** *Double precision version of MKFL.*
**DMLIN.** Numerical integration of a function of several variables over a hyper-rectangle (Gaussian method). *Proprietary Fortran function in IMSL.* **Classes:** H2b1a1 **Usage:** R = DMLIN(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**DMLSE.** *Double precision version of MLSE.*
**DMNPB.** *Double precision version of MNPB.*
**DMOLCH.** *Double precision version of MOLCH.*
**DMRRRR.** *Double precision version of MRRRR.*
**DMSCR.** Use Fisher's linear discriminant analysis method to reduce the number of variables. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L12 **Usage:** CALL DMSCR (*13 parameters*) **Precision:** Single (Double: DDMSCR) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DMSDST.** *Double precision version of MSDST.*
**DMSIDV.** *Double precision version of MSIDV.*
**DMURBV.** *Double precision version of MURBV.*
**DMURRV.** *Double precision version of MURRV.*
**DMVIND.** *Double precision version of MVIND.*
**DMWFE.** *Double precision version of MWFE.*
**DMXTXF.** *Double precision version of MXTXF.*
**DMXTYF.** *Double precision version of MXTYF.*
**DMXYTF.** *Double precision version of MXYTF.*
**DNBCO.** *Double precision version of SNBCO.*
**DNBDI.** *Double precision version of SNBDI.*
**DNBFA.** *Double precision version of SNBFA.*
**DNBFS.** *Double precision version of SNBFS.*
**DNBSL.** *Double precision version of SNBSL.*
**DNCONF.** *Double precision version of NCONF.*
**DNCONG.** *Double precision version of NCONG.*
**DNCTRD.** *Double precision version of NCTRD.*
**DNEQNF.** *Double precision version of NEQNF.*
**DNEQNJ.** *Double precision version of NEQNJ.*
**DNL2S1.** *Double precision version of NL2S1.*
**DNL2SN.** *Double precision version of NL2SN.*
**DNNBRD.** *Double precision version of NNBRD.*
**DNORDF.** *Double precision version of ANORDF.*
**DNORIN.** *Double precision version of ANORIN.*
**DNR1RB.** *Double precision version of NR1RB.*
**DNR1RR.** *Double precision version of NR1RR.*
**DNR2RR.** *Double precision version of NR2RR.*
**DNRCES.** *Double precision version of NRCES.*
**DNRIRR.** *Double precision version of NRIRR.*
**DNRM2.** *Double precision version of SNRM2.*
**DNSBJF.** *Double precision version of NSBJF.*
**DNSLSE.** *Double precision version of NSLSE.*
**DNSPE.** *Double precision version of NSPE.*
**DNSQE.** *Double precision version of SNSQE.*
**DNTIES.** *Double precision version of NTIES.*
**DODEQ.** *Double precision version of ODEQ.*
**DODES.** *Double precision version of ODES.*
**DODES1.** *Double precision version of ODES1.*
**DODESE.** *Double precision version of ODESE.*
**DODESH.** *Double precision version of ODESH.*
**DODR.** *Double precision version of SODR.*
**DODRC.** *Double precision version of SODRC.*
**DOPOLY.** *Double precision version of OPOLY.*
**DORTHP.** *Double precision version of ORTHP.*
**DOT.** Creates a tree of clusters of cases for categorical data by minimum-mutation fits. (Changes of variable values between a cluster and its direct ancestor in the tree are minimized.) *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a3 **Usage:** CALL DOT(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.
**DOUBLE EXPONENTIAL PROBABILITY PLOT.** Generates a probability plot for the double exponential distribution with mean 0 and standard deviation $\sqrt{2}$. Alternate name: LAPLACE PROBABILITY PLOT. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2d **Usage:** DOUBLE EXPONENTIAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**DOWFRQ.** *Double precision version of OWFRQ.*
**DPACF.** *Double precision version of PACF.*
**DPAIRS.** *Double precision version of PAIRS.*
**DPBCO.** *Double precision version of SPBCO.*
**DPBDI.** *Double precision version of SPBDI.*
**DPBFA.** *Double precision version of SPBFA.*
**DPBSL.** *Double precision version of SPBSL.*
**DPCGRC.** *Double precision version of PCGRC.*
**DPCHEV.** *Double precision version of PCHEV.*
**DPCHEZ.** *Double precision version of PCHEZ.*
**DPCHFD.** *Double precision version of PCHFD.*
**DPCHFE.** *Double precision version of PCHFE.*
**DPCHIA.** *Double precision version of PCHIA.*
**DPCHIC.** *Double precision version of PCHIC.*
**DPCHID.** *Double precision version of PCHID.*
**DPCHIM.** *Double precision version of PCHIM.*
**DPCHMC.** *Double precision version of PCHMC.*
**DPCHQA.** *Double precision version of PCHQA.*
**DPCHSP.** *Double precision version of PCHSP.*
**DPDES.** Solve a system of partial differential equations of the form $u_t = f(x, t, u, u_x, u_{xx})$ using the method of lines with cubic Hermite polynomials. *Proprietary Fortran subroutine in IMSL.* **Classes:** I2a1a, I2a2 **Usage:** CALL DPDES (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DPERMA.** *Double precision version of PERMA.*
**DPERMU.** *Double precision version of PERMU.*
**DPFQAD.** *Double precision version of PFQAD.*
**DPLOTP.** *Double precision version of PLOTP.*
**DPOCH.** *Double precision version of POCH.*
**DPOCH1.** *Double precision version of POCH1.*
**DPOCO.** *Double precision version of SPOCO.*
**DPODI.** *Double precision version of SPODI.*
**DPOFA.** *Double precision version of SPOFA.*
**DPOFS.** *Double precision version of SPOFS.*
**DPOIDF.** *Double precision version of POIDF.*
**DPOIES.** *Double precision version of POIES.*
**DPOIPR.** *Double precision version of POIPR.*
**DPOLRG.** *Double precision version of POLRG.*
**DPOSL.** *Double precision version of SPOSL.*
**DPPCO.** *Double precision version of SPPCO.*
**DPPDER.** *Double precision version of PPDER.*
**DPPDI.** *Double precision version of SPPDI.*
**DPPFA.** *Double precision version of SPPFA.*
**DPPITG.** *Double precision version of PPITG.*
**DPPQAD.** *Double precision version of PPQAD.*
**DPPSL.** *Double precision version of SPPSL.*
**DPPVAL.** *Double precision version of PPVAL.*
**DPRDCT.** *Double precision version of SPRDCT.*
**DPRINC.** *Double precision version of PRINC.*
**DPROBP.** *Double precision version of PROBP.*
**DPRPFT.** *Double precision version of PRPFT.*
**DPSI.** *Double precision version of PSI.*
**DPSIFN.** *Double precision version of PSIFN.*
**DPTSL.** *Double precision version of SPTSL.*
**DPUMB.** *Double precision version of PUMB.*
**DPUMD.** *Double precision version of PUMD.*
**DQ1DA.** *Double precision version of Q1DA.*
**DQADD.** Add a double precision scalar to the accumulator in extended precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** A3c **Usage:** CALL DQADD (A, ACC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DQAG.** *Double precision version of QAG.*
**DQAGE.** *Double precision version of QAGE.*

**DQAGI.** *Double precision version of QAGI.*
**DQAGIE.** *Double precision version of QAGIE.*
**DQAGP.** *Double precision version of QAGP.*
**DQAGPE.** *Double precision version of QAGPE.*
**DQAGS.** *Double precision version of QAGS.*
**DQAGSE.** *Double precision version of QAGSE.*
**DQAND.** *Double precision version of QAND.*
**DQAWC.** *Double precision version of QAWC.*
**DQAWCE.** *Double precision version of QAWCE.*
**DQAWF.** *Double precision version of QAWF.*
**DQAWFE.** *Double precision version of QAWFE.*
**DQAWO.** *Double precision version of QAWO.*
**DQAWOE.** *Double precision version of QAWOE.*
**DQAWS.** *Double precision version of QAWS.*
**DQAWSE.** *Double precision version of QAWSE.*
**DQC25C.** *Double precision version of QC25C.*
**DQC25F.** *Double precision version of QC25F.*
**DQC25S.** *Double precision version of QC25S.*
**DQCOSB.** *Double precision version of QCOSB.*
**DQCOSF.** *Double precision version of QCOSF.*
**DQD2DR.** *Double precision version of QD2DR.*
**DQD2VL.** *Double precision version of QD2VL.*
**DQD3DR.** *Double precision version of QD3DR.*
**DQD3VL.** *Double precision version of QD3VL.*
**DQDAG.** *Double precision version of QDAG.*
**DQDAGI.** *Double precision version of QDAGI.*
**DQDAGP.** *Double precision version of QDAGP.*
**DQDAGS.** *Double precision version of QDAGS.*
**DQDAWC.** *Double precision version of QDAWC.*
**DQDAWF.** *Double precision version of QDAWF.*
**DQDAWO.** *Double precision version of QDAWO.*
**DQDAWS.** *Double precision version of QDAWS.*
**DQDDER.** *Double precision version of QDDER.*
**DQDDOT.** *Double precision version of SDSDOT.*
**DQDNG.** *Double precision version of QDNG.*
**DQDOTA.** *Double precision version of SDDOTA.*
**DQDOTI.** *Double precision version of SDDOTI.*
**DQDVAL.** *Double precision version of QDVAL.*
**DQINI.** Initialize an extended-precision accumulator with a double precision scalar. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** A3c **Usage:** CALL DQINI (D, ACC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DQK15.** *Double precision version of QK15.*
**DQK15I.** *Double precision version of QK15I.*
**DQK15W.** *Double precision version of QK15W.*
**DQK21.** *Double precision version of QK21.*
**DQK31.** *Double precision version of QK31.*
**DQK41.** *Double precision version of QK41.*
**DQK51.** *Double precision version of QK51.*
**DQK61.** *Double precision version of QK61.*
**DQMOMO.** *Double precision version of QMOMO.*
**DQMUL.** Multiply double precision scalars in extended precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** A3c **Usage:** CALL DQMUL (A, B, ACC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DQNG.** *Double precision version of QNG.*
**DQPROG.** *Double precision version of QPROG.*
**DQRANK.** *Double precision version of SQRANK.*
**DQRDC.** *Double precision version of SQRDC.*
**DQRLS.** *Double precision version of SQRLS.*
**DQRLSS.** *Double precision version of SQRLSS.*
**DQRSL.** *Double precision version of SQRSL.*

**DQSINB.** *Double precision version of QSINB.*
**DQSINF.** *Double precision version of QSINF.*
**DQSTO.** Store a double precision approximation to an extended-precision scalar. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** A3c **Usage:** CALL DQSTO (ACC, D) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DQTEST.** *Double precision version of QTEST.*
**DQUAD.** *Double precision version of QUAD.*
**DRANDOM.** Generates K pseudo-random numbers from a user-specified discrete distribution. *Command in MINITAB proprietary interactive system.* **Classes:** L6a7 **Usage:** DRANdom K observations, values in C, probabilities in C, put into C **Precision:** Single **Availability:** 855NOS.
**DRANKS.** *Double precision version of RANKS.*
**DRATCH.** *Double precision version of RATCH.*
**DRBEST.** *Double precision version of RBEST.*
**DRCASE.** *Double precision version of RCASE.*
**DRCASP.** *Double precision version of RCASP.*
**DRCOMP.** *Double precision version of RCOMP.*
**DRCOV.** *Double precision version of RCOV.*
**DRCOVB.** *Double precision version of RCOVB.*
**DRCURV.** *Double precision version of RCURV.*
**DREBS.** Solve an initial-value problem for ordinary differential equations using the Bulirsch-Stoer extrapolation method. *Proprietary Fortran subroutine in IMSL.* **Classes:** I1a1c **Usage:** CALL DREBS (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DRECCF.** *Double precision version of RECCF.*
**DRECQR.** *Double precision version of RECQR.*
**DRFORP.** *Double precision version of RFORP.*
**DRGIVN.** *Double precision version of RGIVN.*
**DRGLM.** *Double precision version of RGLM.*
**DRINCF.** *Double precision version of RINCF.*
**DRINPF.** *Double precision version of RINPF.*
**DRLAV.** *Double precision version of RLAV.*
**DRLEQU.** *Double precision version of RLEQU.*
**DRLINE.** *Double precision version of RLINE.*
**DRLMV.** *Double precision version of RLMV.*
**DRLTR.** *Double precision version of RLTR.*
**DRNARM.** *Double precision version of RNARM.*
**DRNBET.** *Double precision version of RNBET.*
**DRNCHI.** *Double precision version of RNCHI.*
**DRNCHY.** *Double precision version of RNCHY.*
**DRNCOR.** *Double precision version of RNCOR.*
**DRNEXP.** *Double precision version of RNEXP.*
**DRNEXT.** *Double precision version of RNEXT.*
**DRNGAM.** *Double precision version of RNGAM.*
**DRNGCS.** *Double precision version of RNGCS.*
**DRNGCT.** *Double precision version of RNGCT.*
**DRNGDA.** *Double precision version of RNGDA.*
**DRNGDS.** *Double precision version of RNGDS.*
**DRNGDT.** *Double precision version of RNGDT.*
**DRNGES.** *Double precision version of RNGES.*
**DRNKSM.** *Double precision version of RNKSM.*
**DRNLIN.** *Double precision version of RNLIN.*
**DRNLNL.** *Double precision version of RNLNL.*
**DRNMVN.** *Double precision version of RNMVN.*
**DRNNOA.** *Double precision version of RNNOA.*
**DRNNOF.** *Double precision version of RNNOF.*
**DRNNOR.** *Double precision version of RNNOR.*
**DRNNOS.** *Double precision version of RNNOS.*
**DRNNPP.** *Double precision version of RNNPP.*
**DRNSES.** *Double precision version of RNSES.*
**DRNSPH.** *Double precision version of RNSPH.*
**DRNSRS.** *Double precision version of RNSRS.*

**DRNSTA.** *Double precision version of RNSTA.*
**DRNTRI.** *Double precision version of RNTRI.*
**DRNUN.** *Double precision version of RNUN.*
**DRNUNF.** *Double precision version of RNUNF.*
**DRNUNO.** *Double precision version of RNUNO.*
**DRNVMS.** *Double precision version of RNVMS.*
**DRNWIB.** *Double precision version of RNWIB.*
**DRONE.** *Double precision version of RONE.*
**DRORDM.** *Double precision version of RORDM.*
**DROREX.** *Double precision version of ROREX.*
**DROT.** *Double precision version of SROT.*
**DROTG.** *Double precision version of SROTG.*
**DROTIN.** *Double precision version of ROTIN.*
**DROTM.** *Double precision version of SROTM.*
**DROTMG.** *Double precision version of SROTMG.*
**DRPOLY.** *Double precision version of RPOLY.*
**DRPOLY.** *Double precision version of RPLOY.*
**DRQUAD.** *Double precision version of RQUAD.*
**DRSTAP.** *Double precision version of RSTAP.*
**DRSTAT.** *Double precision version of RSTAT.*
**DRSTEP.** *Double precision version of RSTEP.*
**DRUNS.** *Double precision version of RUNS.*
**DRVTE.** Calculate first, second, or third derivative of a user-supplied function. *Proprietary Fortran function in IMSL.* **Classes:** H1 **Usage:** R = DRVTE(F, N, X, H, ERR, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DSCAL.** *Double precision version of SSCAL.*
**DSCOLR.** *Double precision version of SCOLR.*
**DSCRM.** Perform a linear or a quadratic discriminant function analysis among several known groups. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L12 **Usage:** CALL DSCRM (*30 parameters*) **Precision:** Single (Double: DDSCRM) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DSCTP.** *Double precision version of SCTP.*
**DSDOT.** Compute single precision dot product x·y using double precision accumulation. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a4 **Usage:** R = DSDOT(N, SX, INCX, SY, INCY) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.
**DSDOT.** Compute single precision dot product x·y using double precision accumulation. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** R = DSDOT(N, SX, INCX, SY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DSDPLC.** *Double precision version of SDPLC.*
**DSET.** *Double precision version of SSET.*
**DSHI.** *Double precision version of SHI.*
**DSI.** *Double precision version of SI.*
**DSICO.** *Double precision version of SSICO.*
**DSIDI.** *Double precision version of SSIDI.*
**DSIFA.** *Double precision version of SSIFA.*
**DSIGNT.** *Double precision version of SIGNT.*
**DSIN.** *Double precision version of SIN.*
**DSINDG.** *Double precision version of SINDG.*
**DSINH.** *Double precision version of SINH.*
**DSISL.** *Double precision version of SSISL.*
**DSMPPR.** *Double precision version of SMPPR.*
**DSMPPS.** *Double precision version of SMPPS.*
**DSMPRR.** *Double precision version of SMPRR.*
**DSMPRS.** *Double precision version of SMPRS.*
**DSMPSC.** *Double precision version of SMPSC.*
**DSMPSR.** *Double precision version of SMPSR.*
**DSMPSS.** *Double precision version of SMPSS.*
**DSMPST.** *Double precision version of SMPST.*
**DSMSNO.** *Double precision version of SMSNO.*
**DSNKMC.** *Double precision version of SNKMC.*
**DSNRNK.** *Double precision version of SNRNK.*

**DSPCO.** *Double precision version of SSPCO.*
**DSPDI.** *Double precision version of SSPDI.*
**DSPENC.** *Double precision version of SPENC.*
**DSPFA.** *Double precision version of SSPFA.*
**DSPLN1.** *Double precision version of SPLN1.*
**DSPLN2.** *Double precision version of SPLN2.*
**DSPLND.** *Double precision version of SPLND.*
**DSPLNE.** *Double precision version of SPLNE.*
**DSPLNI.** *Double precision version of SPLNI.*
**DSPSL.** *Double precision version of SSPSL.*
**DSPWF.** *Double precision version of SPWF.*
**DSQAR.** Perform a D-square test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1d **Usage:** CALL DSQAR (*9 parameters*) **Precision:** Single (Double: DDSQAR) **Availability:** 855NOS, 855VE, 205, 840NOS.
**DSQRT.** *Double precision version of SQRT.*
**DSRCH.** *Double precision version of SRCH.*
**DSSWP.** *Double precision version of SSWP.*
**DSTMLP.** *Double precision version of STMLP.*
**DSUB.** *Double precision version of SSUB.*
**DSUM.** *Double precision version of SSUM.*
**DSUMSL.** *Double precision version of SUMSL.*
**DSURF.** *Double precision version of SURF.*
**DSVDC.** *Double precision version of SSVDC.*
**DSVRBN.** *Double precision version of SVRBN.*
**DSVRBP.** *Double precision version of SVRBP.*
**DSVRGN.** *Double precision version of SVRGN.*
**DSVRGP.** *Double precision version of SVRGP.*
**DSWAP.** *Double precision version of SSWAP.*
**DTAN.** *Double precision version of TAN.*
**DTANH.** *Double precision version of TANH.*
**DTCHBP.** *Double precision version of TCHBP.*
**DTCSCP.** *Double precision version of TCSCP.*
**DTDF.** *Double precision version of TDF.*
**DTFPE.** *Double precision version of TFPE.*
**DTIME.** Get time of day. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** R **Usage:** CALL DTIME (IHOUR, MINUTE, ISEC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DTIN.** *Double precision version of TIN.*
**DTLS.** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution. From: "The Extended Classical Total Least Squares Algorithm", by S. Van Huffel, J. Comp. Appl. Math. 25, (1989) 111-119. *Portable Fortran software in JCAM library.* **Classes:** D9a1, D9a4 **Usage:** CALL DTLS(*15 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)
**DTNDF.** *Double precision version of TNDF.*
**DTPTB.** Solve a system of ordinary differential equations with boundary conditions at two points, using a multiple shooting method. *Proprietary Fortran subroutine in IMSL.* **Classes:** I1b2 **Usage:** CALL DTPTB (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DTRCO.** *Double precision version of STRCO.*
**DTRDI.** *Double precision version of STRDI.*
**DTREEP.** *Double precision version of TREEP.*
**DTRIA.** Computes an approximation to the double integral of f(u,v) over a triangle in the uv-plane by using an $n^2$ point, generalized Gauss-Legendre product rule of polynomial degree precision 2n-2. From: "Computation of Double Integrals over a Triangle", by F.G. Lether, Algorithm 007, J. Comp. Appl. Math. 2(1976), 219-224. *Portable Fortran software in JCAM library.* **Classes:** H2b2a2 **Usage:** CALL DTRIA(*11 parameters*) **Precision:** Single **Availability:** 855NOS.
**DTRIGP.** *Double precision version of TRIGP.*
**DTRNRR.** *Double precision version of TRNRR.*
**DTRSL.** *Double precision version of STRSL.*
**DTWFRQ.** *Double precision version of TWFRQ.*

**DTWODQ.** *Double precision version of TWODQ.*
**DTWOMV.** *Double precision version of TWOMV.*
**DUMB.** *Double precision version of UMB.*
**DUMCGF.** *Double precision version of UMCGF.*
**DUMCGG.** *Double precision version of UMCGG.*
**DUMD.** *Double precision version of UMD.*
**DUMIAH.** *Double precision version of UMIAH.*
**DUMIDH.** *Double precision version of UMIDH.*
**DUMINF.** *Double precision version of UMINF.*
**DUMING.** *Double precision version of UMING.*
**DUMKFL.** *Double precision version of UMKFL.*
**DUMPOL.** *Double precision version of UMPOL.*
**DUNLSF.** *Double precision version of UNLSF.*
**DUNLSJ.** *Double precision version of UNLSJ.*
**DUVMGS.** *Double precision version of UVMGS.*
**DUVMID.** *Double precision version of UVMID.*
**DUVMIF.** *Double precision version of UVMIF.*
**DUVSTA.** *Double precision version of UVSTA.*
**DVBTOD.** *Double precision version of VBTOD.*
**DVCAL.** *Double precision version of SVCAL.*
**DVCPR.** Solve a system of ordinary differential equations with boundary conditions at two points, using a variable order, variable step size finite difference method with deferred corrections. *Proprietary Fortran subroutine in IMSL.* **Classes:** I1b2 **Usage:** CALL DVCPR (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.
**DVDTOB.** *Double precision version of VDTOB.*
**DVERK.** Differential equation solver – Runge-Kutta-Verner fifth and sixth order method. *Proprietary Fortran subroutine in IMSL.* **Classes:** I1a1a **Usage:** CALL DVERK (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.
**DVHS2P.** *Double precision version of VHS2P.*
**DVHSTP.** *Double precision version of VHSTP.*
**DWRRRL.** *Double precision version of WRRRL.*
**DWRRRN.** *Double precision version of WRRRN.*
**DXYZ.** *Double precision version of SXYZ.*
**DZBREN.** *Double precision version of ZBREN.*
**DZERO.** *Double precision version of ZERO.*
**DZONE.** *Double precision version of ZONE.*
**DZONEJ.** *Double precision version of ZONEJ.*
**DZREAL.** *Double precision version of ZREAL.*

# E

**E01AAF.** Interpolates at a given point X from a table of function values $y_i$ evaluated at equidistant or non-equidistant points $x_i$ (i=1, 2, ..., N+1), using Aitken's technique of successive linear interpolations. *Proprietary Fortran subroutine in NAG.* **Classes:** E1b **Usage:** CALL E01AAF (A, B, C, N1, N2, N, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E01ABF.** Interpolates at a given point X from a table of function values evaluated at equidistant points, by Everett's formula. *Proprietary Fortran subroutine in NAG.* **Classes:** E1b **Usage:** CALL E01ABF (N, P, A, G, N1, N2, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E01ACF.** Interpolates at a given point (A,B) from a table of function values defined on a rectangular grid in the X-Y plane, by fitting bi-cubic spline functions. *Proprietary Fortran subroutine in NAG.* **Classes:** E2a **Usage:** CALL E01ACF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E01AEF.** Constructs the Chebyshev series representation of a polynomial interpolant to a set of data which may contain derivative values. *Proprietary Fortran subroutine in NAG.* **Classes:** E1b **Usage:** CALL E01AEF (*14 parameters*) **Also see:** E02AKF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E01BAF.** Determines a cubic-spline interpolant to a given set of data. *Proprietary Fortran subroutine in NAG.* **Classes:** E1a **Usage:** CALL E01BAF (*9 parameters*) **Also see:** E02BBF E02BCF E02BDF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E01RAF.** Produces, from a set of function values and corresponding abscissae, the coefficients of an interpolating rational function expressed in continued fraction form. *Proprietary Fortran subroutine in NAG.* **Classes:** E1c **Usage:** CALL E01RAF (N, X, F, M, A, U, IW, IFAIL) **Also see:** E01RBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E01RBF.** Evaluates continued fractions of the form produced by NAG FORTRAN Library routine E01RAF. *Proprietary Fortran subroutine in NAG.* **Classes:** E3a1 **Usage:** CALL E01RBF(M, A, U, X, F, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02ACF.** Calculates a minimax polynomial fit to a set of data points. *Proprietary Fortran subroutine in NAG.* **Classes:** K2 **Usage:** CALL E02ACF (X, Y, N, A, M1, REF) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02ADF.** Computes weighted least-squares polynomial approximations to an arbitrary set of data points. *Proprietary Fortran subroutine in NAG.* **Classes:** K1a1a2 **Usage:** CALL E02ADF (*11 parameters*) **Also see:** E02AEF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02AEF.** Evaluates a polynomial from its Chebyshev series representation. *Proprietary Fortran subroutine in NAG.* **Classes:** C3a2, E3a1, K6a1 **Usage:** CALL E02AEF (NPLUS1, A, XCAP, P, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02AFF.** Computes the coefficients of a polynomial, in its Chebyshev series form, which interpolates (passes exactly through) data at a special set of points. Least-squares polynomial approximations can also be obtained. *Proprietary Fortran subroutine in NAG.* **Classes:** K1a1a2, E1b **Usage:** CALL E02AFF (NPLUS1, F, A, IFAIL) **Also see:** E02AEF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02AGF.** Computes constrained weighted least-squares polynomial approximations in Chebyshev series form to an arbitrary set of data points. The values of the approximations and any number of their derivatives can be specified at selected points. *Proprietary Fortran subroutine in NAG.* **Classes:** K1a2a **Usage:** CALL E02AGF (*21 parameters*) **Also see:** E02AKF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02AHF.** Determines the coefficients in the Chebyshev series representation of the derivative of a polynomial given in Chebyshev series form. *Proprietary Fortran subroutine in NAG.* **Classes:** C3a2, E3a2, K6a2 **Usage:** CALL E02AHF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02AJF.** Determines the coefficients in the Chebyshev series representation of the indefinite integral of a polynomial given in Chebyshev series form. *Proprietary Fortran subroutine in NAG.* **Classes:** C3a2, E3a3, H2a2b1, K6a3 **Usage:** CALL E02AJF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02AKF.** Evaluates a polynomial from its Chebyshev series representation, allowing an arbitrary index increment for accessing the array of coefficients. *Proprietary Fortran subroutine in NAG.* **Classes:** C3a2, E3a1, K6a1 **Usage:** CALL E02AKF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02BAF.** Computes a weighted least-squares approximation to an arbitrary set of data points by a cubic spline with knots prescribed by the user. Cubic spline interpolation can also be carried out. *Proprietary Fortran subroutine in NAG.* **Classes:** K1a1a1, E1a **Usage:** CALL E02BAF (*11 parameters*) **Also see:** E02BBF E02BCF E02BDF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02BBF.** Evaluates a cubic spline from its B-spline representation. *Proprietary Fortran subroutine in NAG.* **Classes:** E3a1, K6a1 **Usage:** CALL E02BBF (NCAP7, K, C, X, S, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02BCF.** Evaluates a cubic spline and its first three derivatives from its B-spline representation. *Proprietary Fortran subroutine in NAG.* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL E02BCF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02BDF.** Computes the definite integral of a cubic spline from its B-spline representation. *Proprietary Fortran subroutine in NAG.* **Classes:** E3a3, H2a2b1, K6a3 **Usage:** CALL E02BDF (NCAP7, K, C, DEFINT, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02CAF.** Forms an approximation to the weighted, least-squares Chebyshev series surface fit to data arbitrarily distributed on lines parallel to one independent co-ordinate axis. *Proprietary Fortran subroutine in NAG.* **Classes:** K1a1b **Usage:** CALL E02CAF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02CBF.** Evaluates a bivariate polynomial from the rectangular array of coefficients in its double Chebyshev series representation. *Proprietary Fortran subroutine in NAG.* **Classes:** E3a1, K6a1 **Usage:** CALL E02CBF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02DAF.** Forms a minimal, weighted least-squares bicubic spline surface fit with prescribed knots to a given set of data points. *Proprietary Fortran subroutine in NAG.* **Classes:** K1a1b **Usage:** CALL E02DAF (*19 parameters*) **Also see:** E02DBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02DBF.** Calculates values of a bicubic spline from its B-spline representation. *Proprietary Fortran subroutine in NAG.* **Classes:** E3a1, K6a1 **Usage:** CALL E02DBF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02GAF.** Calculates an $l_1$ solution to an over-determined system of linear equations. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a3 **Usage:** CALL E02GAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02GBF.** Calculates an $l_1$ solution to an over-determined system of linear equations, possibly subject to linear inequality constraints. *Proprietary Fortran subroutine in NAG.* **Classes:** D9b3 **Usage:** CALL E02GBF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02GCF.** Calculates an $L_\infty$ solution to an over-determined system of linear equations. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a2 **Usage:** CALL E02GCF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02RAF.** Calculates the coefficients in a Pade approximant to a function from its user-supplied Maclaurin expansion. *Proprietary Fortran subroutine in NAG.* **Classes:** K4 **Usage:** CALL E02RAF (*9 parameters*) **Also see:** E02RBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02RBF.** Evaluates a rational function at a user-supplied point, give the numerator and denominator coefficients. *Proprietary Fortran subroutine in NAG.* **Classes:** K6a1 **Usage:** CALL E02RBF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E02ZAF.** Sorts two-dimensional data into rectangular panels. *Proprietary Fortran subroutine in NAG.* **Classes:** E3d, K6d **Usage:** CALL E02ZAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04ABF.** Searches for a minimum, in a given finite interval, of a continuous function of a single variable, using function values only. The method (based on quadratic interpolation) is intended for functions which have a continuous first derivative (although it will usually work if the derivative has occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G1a1a **Usage:** CALL E04ABF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04BBF.** Searches for a minimum, in a given finite interval, of a continuous function of a single variable, using function and first derivative values. The method (based on cubic interpolation) is intended for functions which have a continuous first derivative (although it will usually work if the derivative has occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G1a1b **Usage:** CALL E04BBF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04CCF.** Minimizes a general function F(X) of N independent variables X = (X(1),X(2),...,X(N))(T) by the Simplex method. Derivatives of the function need not be supplied. *Proprietary Fortran subroutine in NAG.* **Classes:** G1b2 **Usage:** CALL E04CCF(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04CGF.** Is an easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a function of N independent variables using function values only. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G1b1a **Usage:** CALL E04CGF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04DBF.** Minimizes a general function F(X) of N independent variables by the conjugate gradient method due to Fletcher and Reeves. Formulae to calculate the value of the function and its first derivatives must be supplied. *Proprietary Fortran subroutine in NAG.* **Classes:** G1b1b **Usage:** CALL E04DBF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04DEF.** Is an easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a function of N independent variables when first derivatives are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G1b1b **Usage:** CALL E04DEF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04DFF.** Is an easy-to-use modified-Newton algorithm for finding an unconstrained minimum of a function of N independent variables when first derivatives are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G1b1b **Usage:** CALL E04DFF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04DGF.** Minimizes an unconstrained nonlinear function of several variables using a pre-conditioned, limited memory quasi-Newton conjugate gradient method. The routine is intended for use on large scale problems. *Proprietary Fortran subroutine in NAG.* **Classes:** G1b1b **Usage:** CALL E04DGF (*11 parameters*) **Also see:** E04DJF E04DKF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04DJF.** Supplies optional parameters to NAG Fortran Library routine E04DGF from an external file. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f **Usage:** CALL E04DJF (IOPTNS, INFORM) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04DKF.** Supplies individual optional parameters to NAG Fortran Library routine E04DGF. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f **Usage:** CALL E04DKF (STRING) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04EBF.** Is an easy-to-use modified-Newton algorithm for finding an unconstrained minimum of a function of N independent variables when first and second derivatives are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G1b1c **Usage:** CALL E04EBF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04FCF.** A comprehensive algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables ($m \geq n$). No derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a1 **Usage:** CALL E04FCF (*24 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04FDF.** An easy-to-use algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables ($m \geq n$). No derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a1 **Usage:** CALL E04FDF(*9 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04GBF.** A comprehensive quasi-Newton algorithm for finding an unconstrained minimum of a sum of squares of M non-linear functions in N variables ($m \geq n$). First derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a2 **Usage:** CALL E04GBF (*25 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04GCF.** An easy-to-use quasi-Newton algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables ($m \geq n$). First derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a2 **Usage:** CALL E04GCF (*9 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04GDF.** A comprehensive modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of M non-linear functions in N variables ($m \geq n$). First derivatives are required. The routine is intended for functions which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a2 **Usage:** CALL E04GDF (*23 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04GEF.** An easy-to-use modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of M nonlinear functions in N variables ($m \geq n$). First derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a2 **Usage:** CALL E04GEF (*9 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04HBF.** Computes a sensible set of finite-difference intervals for input to a quasi-Newton minimization routine which does not require derivatives. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f **Usage:** CALL E04HBF (N, FUNCT, X, NF, DELTA, HESL, LH, HESD, F, G, IW, LIW, W, LW, IFAIL **Also see:** E04JBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04HCF.** Checks that a user-supplied routine for evaluating an objective function and its first derivative values are consistent with the function values calculated. *Proprietary Fortran subroutine in NAG.* **Classes:** G4c, F3 **Usage:** CALL E04HCF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04HDF.** Checks that a user-supplied routine for calculating second derivatives of an objective function is consistent with a user-supplied routine for calculating the corresponding first derivatives. *Proprietary Fortran subroutine in NAG.* **Classes:** G4c, F3 **Usage:** CALL E04HDF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04HEF.** A comprehensive modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of m non-linear functions in n variables (m $\geq$ n). First and second derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a3 **Usage:** CALL E04HEF (*25 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04HFF.** An easy-to-use modified Gauss-Newton algorithm for finding an unconstrained minimum of a sum of squares of m nonlinear functions in n variables (m $\geq$ n). First and second derivatives are required. *Proprietary Fortran subroutine in NAG.* **Classes:** K1b1a3 **Usage:** CALL E04HFF (*9 parameters*) **Also see:** E04YCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04JAF.** An easy-to-use quasi-Newton algorithm for finding a minimum of a function, subject to fixed upper and lower bounds of the independent variables, using function values only. *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a1 **Usage:** CALL E04JAF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04JBF.** A comprehensive quasi-Newton algorithm for finding a minimum of a function of several variables, optionally subject to fixed upper and/or lower bounds on the variables. The user does not need to supply any derivatives of the function. The routine is intended for functions which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a1, G1b1a **Usage:** CALL E04JBF (*28 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04KAF.** An easy-to-use quasi-Newton algorithm for finding a minimum of a function subject to fixed upper and lower bounds on the independent variables when first derivatives of F are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a2 **Usage:** CALL E04KAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**E04KBF.** A comprehensive quasi-Newton algorithm for finding a minimum of a function of several variables optionally subject to a fixed upper and/ or lower bounds on the variables. First derivatives are required. The routine is intended for functions which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a2 **Usage:** CALL E04KBF (*27 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04KCF.** An easy-to-use modified-Newton algorithm for finding a minimum of a function, subject to fixed upper and lower bounds on the independent variables, when first derivatives of F are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a2 **Usage:** CALL E04KCF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04KDF.** A comprehensive modified Newton algorithm for finding: – an unconstrained minimum of a function of several variables – a minimum of a function of several variables subject to fixed upper and/or lower bounds on the variables. *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a2 **Usage:** CALL E04KDF (*24 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04LAF.** An easy-to-use modified-Newton algorithm for finding a minimum of a function, subject to fixed upper and lower bounds on the independent variables, when first and second derivatives of F are available. It is intended for functions which are continuous and which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a3 **Usage:** CALL E04LAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04LBF.** A comprehensive modified Newton algorithm for finding a minimum of a function of several variables optionally subject to fixed upper and/ or lower bounds on the variables. First and second derivatives are required. The routine is intended for functions which have continuous first and second derivative (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a3 **Usage:** CALL E04LBF (*24 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04MBF.** An easy-to-use routine for solving linear programming problems, or for finding a feasible point for such problems. It is not intended for large sparse problems. *Proprietary Fortran subroutine in NAG.* **Classes:** G2a1, G4d **Usage:** CALL E04MBF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04NAF.** A comprehensive routine for solving quadratic programming (QP) or linear programming (LP) problems. It is not intended for large sparse problems. *Proprietary Fortran subroutine in NAG.* **Classes:** G2a1, G2e1, G2e2 **Usage:** CALL E04NAF (*29 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04NCF.** Solves linearly constrained linear least-squares problems and convex quadratic programming problems. E04NCF may also be used to solve linear programming problems and to find a feasible point with respect to a set of linear inequality constraints. E04NCF treats all matrices as dense and hence is not intended for large sparse problems. *Proprietary Fortran subroutine in NAG.* **Classes:** D9b1, G2a1, G2e1, G4d **Usage:** CALL E04NCF (*21 parameters*) **Also see:** E04NDF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04NDF.** Supplies optional parameters to NAG Fortran Library routine E04NCF from an external file. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f, K6d **Usage:** CALL E04NDF (IOPTNS, INFORM) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04NEF.** Supplies individual optional parameters to NAG Fortran Library routine E04NCF. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f, K6d **Usage:** CALL E04NEF (STRING) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04UAF.** Attempts to find a minimum of a function of several variables subject to fixed bounds on the variables and to general equality and/or inequality constraints. A sequential augmented Lagrangian method is used, the minimization subproblems involved being solved by a quasi-Newton method. No derivatives are required. The routine is intended for functions and constraints which have continuous first and second derivatives (although it will usually work even if the derivatives have occasional discontinuities). *Proprietary Fortran subroutine in NAG.* **Classes:** G2h3b1a **Usage:** CALL E04UAF (*28 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04UCF.** Minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. The user may provide subroutines that define the objective and constraint functions and as many of their first partial derivatives as possible. Unspecified derivatives are approximated by finite differences. All matrices are treated as dense, and hence E04UCF is not intended for large sparse problems. *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a1, G2h1a2, G2h2a1, G2h2a2, G2h3a1 **Usage:** CALL E04UCF (*27 parameters*) **Also see:** E04UDF E04UEF E04XAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04UDF.** Supplies optional parameters to NAG Fortran Library routine E04UCF from an external file. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f **Usage:** CALL E04UDF (IOPTNS, INFORM) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04UEF.** Supplies individual optional parameters to NAG Fortran Library routine E04UCF. *Proprietary Fortran subroutine in NAG.* **Classes:** G4f **Usage:** CALL E04UEF (STRING) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04VCF.** A comprehensive routine to minimize an arbitrary smooth function subject to constraints, including simple bounds, linear constraints and smooth nonlinear constraints. The user must provide subroutines that define the objective, constraints, and their gradients. All matrices are treated as dense. Uses a sequential quadratic programming algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** G2h3b1b, G2h2a2, G2h1a2 **Usage:** CALL E04VCF (*36 parameters*) **Also see:** E04ZDF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04VDF.** An easy-to-use routine designed to minimize an arbitrary smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. (E04VDF may also be used for unconstrained, bound-constrained and linearly constrained optimization.) The user must provide subroutines that define the objective and constraint functions and their gradients. All matrices are treated as dense, and hence E04VDF is not intended for large sparse problems. *Proprietary Fortran subroutine in NAG.* **Classes:** G2h1a2 **Usage:** CALL E04VDF (*27 parameters*) **Also see:** E04ZDF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04XAF.** Computes an approximation to the gradient vector and the Hessian matrix for use in conjunction with an optimization routine (such as E04UCF). *Proprietary Fortran subroutine in NAG.* **Classes:** G4f **Usage:** CALL E04XAF(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04YAF.** Checks that a user-supplied routine for evaluating a vector of functions and the matrix of their first derivatives produces derivative values which are consistent with the function values calculated. *Proprietary Fortran subroutine in NAG.* **Classes:** G4c **Usage:** CALL E04YAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04YBF.** Checks that a user-supplied routine for evaluating the second derivative term of the Hessian matrix of a sum of squares is consistent with a user-supplied routine for calculating the corresponding first derivatives. *Proprietary Fortran subroutine in NAG.* **Classes:** G4c **Usage:** CALL E04YBF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04YCF.** Returns estimates of elements of the variance-covariance matrix of the estimated regression coefficients for a nonlinear least squares problem. The estimates are derived from the Jacobian of the function f(x) at the solution. *Proprietary Fortran subroutine in NAG.* **Classes:** K6d **Usage:** CALL E04YCF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E04ZCF.** Checks that user-supplied routines for evaluating an objective function, constraint functions and their first derivatives produce derivative values which are consistent with the function and constraint values calculated. *Proprietary Fortran subroutine in NAG.* **Classes:** G4c **Usage:** CALL E04ZCF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**E1.** Integral from x to infinity of exp(-t)/t. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C5 **Usage:** R = E1(X) **Precision:** Single (Double: DE1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**E1.** Exponential integral for arguments greater than zero and the Cauchy principal value of the integral for arguments less than zero. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = E1(X) **Precision:** Single (Double: DE1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**E1ACS.** Piecewise cubic Hermite spline. This routine sets derivatives needed to determine the Hermite representation of the cubic spline interpolant to given data, with specified boundary conditions. The resulting piecewise cubic may be evaluated by E3VAL. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E1a **Usage:** CALL E1ACS(*10 parameters*) **Also see:** E3VAL **Precision:** Single (Double: E1ACSD) **Availability:** PC.

**E1ACSD.** *Double precision version of E1ACS.*

**E1AHE.** Evaluates a piecewise cubic Hermite function and its first derivative at an array of points. It may be used by itself for Hermite interpolation, or as an evaluator for E1AM or E1AIC. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a1, E3a2 **Usage:** CALL E1AHE(*12 parameters*) **Also see:** E1AIC E1AM **Precision:** Single **Availability:** PC.

**E1AM.** Piecewise cubic Hermite interpolation to monotonic data. This routine sets derivatives needed to determine a monotone piecewise cubic Hermite interpolant to given data. Boundary values are provided which are compatible with monotonicity. The interpolant will have an extremum at each point where monotonicity switches direction. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E1a **Usage:** CALL E1AM(N, X, F, D, INCFD, KE) **Also see:** E3VAL **Precision:** Single (Double: E1AMD) **Availability:** PC.

**E1AMD.** *Double precision version of E1AM.*

**E1AQ.** Controls the evaluation of an osculatory quadratic spline which is consistent with the shape of the data (shape preserving). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a1 **Usage:** CALL E1AQ(*9 parameters*) **Also see:** E3VAL **Precision:** Single (Double: E1AQD) **Availability:** PC.

**E1AQD.** *Double precision version of E1AQ.*

**E1VDV.** Evaluates a cubic polynomial given in Hermite form and its first derivative at an array of points where the interval is known. While designed for use by E1AHE, it may be useful directly as an evaluator for a piecewise cubic Hermite function in applications such as graphing, where the interval is known. If only function values are required, use E3V3V instead. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a1, E3a2 **Usage:** CALL E1VDV(*12 parameters*) **Also see:** E1AHE **Precision:** Single **Availability:** PC.

**E3CHK.** Piecewise cubic Hermite monotonicity checker. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3d **Usage:** CALL E3CHK(*8 parameters*) **Precision:** Single **Availability:** PC.

**E3HIN.** Evaluates the definite integral of a piecewise cubic Hermite function over an arbitrary interval. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a3, H2a2b1 **Usage:** CALL E3HIN(N, X, F, D, INCFD, SKIP, A, B, KE) **Precision:** Single **Availability:** PC.

**E3INT.** Evaluates the definite integral of a piecewise cubic Hermite function over an interval whose endpoints are data points. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a3, H2a2b1 **Usage:** CALL E3INT(*9 parameters*) **Precision:** Single **Availability:** PC.

**E3VAL.** Evaluates a piecewise cubic Hermite function at an array of points where the interval is unknown or the evaluation array spans more than one interval. This routine may be used by itself for Hermite interpolation or as an evaluator for E1AM. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a1 **Usage:** CALL E3VAL(*11 parameters*) **Also see:** E1AM **Precision:** Single (Double: E3VALD) **Availability:** PC.

**E3VALD.** *Double precision version of E3VAL.*

**E3VEV.** Evaluates a cubic polynomial given in Hermite form at an array of points where the interval is known in advance. While designed for use by E3VAL, this routine may be used by itself as an evaluator of a piecewise cubic Hermite function in applications, such as graphing, where the interval is known in advance. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** E3a1 **Usage:** CALL E3VEV(*11 parameters*) **Also see:** E3VAL **Precision:** Single (Double: E3VEVD) **Availability:** PC.

**E3VEVD.** *Double precision version of E3VEV.*

**EA.** Given a slowly convergent sequence, this routine extrapolates nonlinearly to a better estimate of its limit, thus improving the rate of convergence. Based on the Epsilon Algorithm of P. Wynn. An estimate of the absolute error is also given. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** A7 **Usage:** CALL EA (*7 parameters*) **Precision:** Single (Double: DEA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EBALAC.** Balance a complex general matrix and isolate eigenvalues whenever possible. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4c1a **Usage:** CALL EBALAC (AR, AI, N, IA, K, L, D) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EBALAF.** Balance a real matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4c1a **Usage:** CALL EBALAF (A, N, IA, D, K, L) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EDGE.** Given a triangulation of a set of points in the plane, forces an edge between two given points in the triangulation. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL EDGE (*9 parameters*) **Also see:** TRMESH **Precision:** Single **Availability:** 855NOS.

**EEBSF.** Estimates the error in a given B-spline fit to a function by refining the mesh. *Proprietary Fortran function in PORT.* **Classes:** E3d, K6d **Usage:** R = EEBSF(K, T1, N1, A1, T2, N2, A2) **Precision:** Single (Double: DEEBSF) **Availability:** 855NOS, 205.

**EEBSI.** Estimates the error in a given B-spline fit to a function by refining the mesh intervals selected by user. *Proprietary Fortran function in PORT.* **Classes:** E3d, K6d **Usage:** R = EEBSI(*11 parameters*) **Precision:** Single (Double: DEEBSI) **Availability:** 855NOS, 205.

**EESFF.** Finds the maximum absolute error in a given B-spline fit to a function. *Proprietary Fortran function in PORT.* **Classes:** E3d, K6d **Usage:** R = EESFF(K, T, N, A, F) **Precision:** Single (Double: DEESFF) **Availability:** 855NOS, 205.

**EESFI.** Finds the maximum absolute error in a given B-spline fit to a function on a set of user selected intervals. *Proprietary Fortran function in PORT.* **Classes:** E3d, K6d **Usage:** R = EESFI(K, T, N, A, F, X, NX, EEST) **Precision:** Single (Double: DEESFI) **Availability:** 855NOS, 205.

**EHESSC.** Reduction of a general complex matrix to complex upper Hessenberg form. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4c1b2 **Usage:** CALL EHESSC (AR, AI, K, L, N, IA, ID) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EHESSF.** Reduction of a nonsymmetric matrix to upper Hessenberg form by orthogonal transformations. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4c1b2 **Usage:** CALL EHESSF (A, K, L, N, IA, D) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EHOUSH.** Reduction of a complex Hermitian matrix to real symmetric tridiagonal form. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4c1b1 **Usage:** CALL EHOUSH (AR, AI, N, D, E, TAU) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EHOUSS.** Reduction of a symmetric matrix to symmetric tridiagonal form using a Householder reduction. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4c1b1 **Usage:** CALL EHOUSS (A, N, D, E, E2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**EI.** Integral from -x to infinity of -exp(-t)/t. Exponential integral for arguments greater than zero and the Cauchy principal value for arguments less than zero. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C5 **Usage:** R = EI(X) **Precision:** Single (Double: DEI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EI.** Integral from -x to infinity of -exp(-t)/t. Exponential integral for arguments greater than zero and the Cauchy principal value for arguments less than zero. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = EI(X) **Precision:** Single (Double: DEI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EIGBS.** Find some eigenvalues and (optionally) eigenvectors of a real symmetric band matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a6 **Usage:** CALL EIGBS (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EIGCC.** Eigenvalues and (optionally) eigenvectors of a complex general matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a4 **Usage:** CALL EIGCC (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EIGCH.** Eigenvalues and (optionally) eigenvectors of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a3 **Usage:** CALL EIGCH (A, N, JOBN, D, Z, IZ, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**EIGEN.** Finds all eigenvalues and eigenvectors of a real matrix. Output consists of pairs of real arrays. *Proprietary Fortran subroutine in PORT.* **Classes:** D4a2 **Usage:** CALL EIGEN (NM, N, A, WR, WI, Z) **Precision:** Single (Double: DEIGEN) **Availability:** 855NOS, 205.

**EIGRF.** Eigenvalues and (optionally) eigenvectors of a real general matrix in full storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a2 **Usage:** CALL EIGRF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EIGRS.** Computes the eigenvalues and, optionally, the eigenvectors of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a1 **Usage:** CALL EIGRS (A, N, JOBN, D, Z, IZ, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**EIGZC.** Eigenvalues and (optionally) eigenvectors of the system $Ax=\lambda Bx$ where A and B are complex matrices. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4b4 **Usage:** CALL EIGZC (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EIGZF.** Eigenvalues and (optionally) eigenvectors of the system $Ax=\lambda Bx$ where A and B are real matrices. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4b2 **Usage:** CALL EIGZF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EIGZS.** Eigenvalues and (optionally) eigenvectors of the system $Ax = \lambda Bx$ where A and B are real symmetric matrices and B is positive definite. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4b1 **Usage:** CALL EIGZS (A, B, N, IJOB, D, Z, IZ, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELE.** Complete elliptic integral E(m), see [AMS55 (17.3.3)]. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C14 **Usage:** R = ELE(X) **Precision:** Single (Double: DELE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELK.** Complete elliptic integral K(m), see [AMS55 (17.3.1)]. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C14 **Usage:** R = ELK(X) **Precision:** Single (Double: DELK) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELLPACK.** Solves linear elliptic partial differential equations in general domains in two dimensions and in boxes; a variety of boundary conditions are handled. Users write programs in the ELLPACK language (Fortran extension) which allows them to declare elliptic problems and to select from a large library of modules to solve them numerically. Results can be tabulated or plotted; the solution is also available as a Fortran function for post-processing. *Proprietary stand-alone program using ELLPACK command language.* **Classes:** I2b1a **Precision:** Single **Availability:** 855NOS.

**ELMBAK.** Forms eigenvectors of real general matrix from eigenvectors of upper Hessenberg matrix output from ELMHES. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL ELMBAK(NM, LOW, IGH, A, INT, M, Z) **Also see:** ELMHES **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ELMHES.** Reduces real general matrix to upper Hessenberg form using stabilized elementary similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b2 **Usage:** CALL ELMHES(NM, N, LOW, IGH, A, INT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**ELRC.** Carlson's incomplete elliptic integral RC(x,y). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C14 **Usage:** R = ELRC(X, Y) **Precision:** Single (Double: DELRC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELRD.** Carlson's incomplete elliptic integral RD(x,y,z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C14 **Usage:** R = ELRD(X, Y, Z) **Precision:** Single (Double: DELRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELRF.** Carlson's incomplete elliptic integral RF(x,y,z). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C14 **Usage:** R = ELRF(X, Y, Z) **Precision:** Single (Double: DELRF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELRJ.** Carlson's incomplete elliptic integral RJ(x,y,z,p). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C14 **Usage:** R = ELRJ(X, Y, Z, RHO) **Precision:** Single (Double: DELRJ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ELTRAN.** Accumulates the stabilized elementary similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by ELMHES. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL ELTRAN(NM, N, LOW, IGH, A, INT, Z) **Also see:** ELMHES **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**ENE.** Exponential integral of integer order for arguments greater than zero scaled by exp(x). *Proprietary Fortran subroutine in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** CALL ENE (X, N, F) **Precision:** Single (Double: DENE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ENOS.** Evaluate the expected value of a normal order statistic. *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** L5a2n **Usage:** R = ENOS(I, N) **Precision:** Single (Double: DENOS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ENTER.** Save current error recovery mode and storage allocation status for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3c **Usage:** CALL ENTER (IRNEW) **Precision:** Single **Availability:** 855NOS, 205.

**ENTSRC.** Saves current recovery mode status and sets a new one for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3a **Usage:** CALL ENTSRC (IROLD, IRNEW) **Precision:** Single **Availability:** 855NOS, 205.

**EPICG.** Compute the performance index for a complex eigensystem. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = EPICG(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EPIHF.** Compute the performance index for a complex Hermitian eigensystem. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = EPIHF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EPIRG.** Compute the performance index for a real eigensystem. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = EPIRG(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EPISB.** Compute the performance index for a real symmetric eigensystem in band symmetric storage mode. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = EPISB(*8 parameters*) **Precision:** Single (Double: DEPISB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EPISF.** Compute the performance index for a real symmetric eigensystem. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D4c **Usage:** R = EPISF(*7 parameters*) **Precision:** Single (Double: DEPISF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EPRINT.** Print the current error message if the program is in the error state for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3c **Usage:** CALL EPRINT **Precision:** Single **Availability:** 855NOS, 205.

**EQRT1S.** Smallest or largest m eigenvalues of a symmetric tridiagonal matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a5 **Usage:** CALL EQRT1S (D, E2, N, M, ISW, IER) **Precision:** Single **Availability:** 855NOS,

855VE, 205, 840NOS.

**EQRT2S.** Eigenvalues and (optionally) eigenvectors of a symmetric tridiagonal matrix using the QL method. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a5 **Usage:** CALL EQRT2S (D, E, N, Z, IZ, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**EQRT3S.** The smallest (or largest) eigenvalues of a tridiagonal matrix in algebraic value whose sum exceeds a given value. *Proprietary Fortran subroutine in IMSL.* **Classes:** D4a5 **Usage:** CALL EQRT3S (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERF.** Error function, = $(2/\sqrt{\pi})$ × the integral from 0 to x of exp(-t$^2$)dt. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C8a, L5a1e **Usage:** R = ERF(X) **Precision:** Single (Double: DERF) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ERF.** Error function, = $(2/\sqrt{\pi})$ × the integral from 0 to x of exp(-t$^2$)dt. *Proprietary Fortran function in IMSL.* **Classes:** C8a, L5a1e **Usage:** R = ERF(Y) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERF.** Error function, = $(2/\sqrt{\pi})$ × the integral from 0 to x of exp(-t$^2$)dt. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8a, L5a1e **Usage:** R = ERF(X) **Precision:** Single (Double: DERF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERF.** Error function, = $(2/\sqrt{\pi})$ × the integral from 0 to x of exp(-t$^2$)dt. *Portable Fortran software in NMS library.* **Classes:** C8a, L5a1e **Usage:** R = ERF(X) **Precision:** Single (Double: DERF) **Availability:** PC.

**ERFC.** Complementary error function, = $(2/\sqrt{\pi})$ × the integral from x to infinity of exp(-t$^2$)dt. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C8a, L5a1e **Usage:** R = ERFC(X) **Precision:** Single (Double: DERFC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ERFC.** Complementary error function, = $(2/\sqrt{\pi})$ × the integral from x to infinity of exp(-t$^2$)dt. *Proprietary Fortran function in IMSL.* **Classes:** C8a, L5a1e **Usage:** R = ERFC(Y) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERFC.** Complementary error function, = $(2/\sqrt{\pi})$ × the integral from x to infinity of exp(-t$^2$)dt. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8a, L5a1e **Usage:** R = ERFC(X) **Precision:** Single (Double: DERFC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERFC.** Double precison complementary error function, = $(2/\sqrt{\pi})$ × the integral from x to infinity of exp(-t$^2$)dt. *Portable Fortran software in NMS library.* **Classes:** C8a, L5a1e **Usage:** R = ERFC(X) **Precision:** Single (Double: DERFC) **Availability:** PC.

**ERFCE.** Exponentially scaled complementary error function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8a **Usage:** R = ERFCE(X) **Precision:** Single (Double: DERFCE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERFCI.** Inverse complementary error function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8a **Usage:** R = ERFCI(X) **Precision:** Single (Double: DERFCI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERFI.** Inverse error function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C8a **Usage:** R = ERFI(X) **Precision:** Single (Double: DERFI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ERRINT.** Error function and complementary error function. *Portable Fortran software in SPECFN library.* **Classes:** C8a **Usage:** CALL ERRINT (X, ERF, ERFC) **Precision:** Double **Availability:** 855NOS (In source form only.)

**ERROFF.** Turns off the error state for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3c **Usage:** CALL ERROFF **Precision:** Single **Availability:** 855NOS, 205.

**ERROR BAR PLOT.** Generates an $X_i$ vs $i$ plot or a Y vs X plot with user specified error limits. Separate positive and negative error limits can be specified. Error limits are specified for the vertical variable only, not for the horizontal variable. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a6, L3b3a **Usage:** ERROR BAR PLOT <Y VARIABLE> <POS ERROR> <NEG ERROR> <X VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**ERSET.** Set error handler default print and stop actions. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** R3c **Usage:** CALL ERSET (IERSVR, IPACT, ISACT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EV1CDF.** Computes the cumulative distribution function value for the extreme value type 1 distribution. *Fortran subroutine in DATAPAC.* **Classes:** L5a1e **Usage:** CALL EV1CDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EV1PLT.** Generates an extreme value type 1 probability plot with mean Euler's number 0.57721566 and standard deviation $\pi/\sqrt{6}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2e **Usage:** CALL EV1PLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EV1PPF.** Computes the percent point function value for the extreme value type 1 distribution with mean Euler's number 0.57721566. *Fortran subroutine in DATAPAC.* **Classes:** L5a2e **Usage:** CALL EV1PPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EV1RAN.** Generates a random sample of size N from the extreme value type 1 distribution with mean Euler's number 0.57721566. *Fortran subroutine in DATAPAC.* **Classes:** L6a5 **Usage:** CALL EV1RAN(N, ISTART, X) **Precision:**

Single **Availability:** 855NOS, 840NOS.

**EV2CDF.** Computes the cumulative distribution function value for the extreme value type 2 distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1e **Usage:** CALL EV2CDF(X, GAMMA, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EV2PLT.** Generates an extreme value type 2 probability plot with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2e **Usage:** CALL EV2PLT(X, N, GAMMA) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EV2PPF.** Computes the percent point function value for the extreme value type 2 distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a2e **Usage:** CALL EV2PPF(P, GAMMA, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EV2RAN.** Generates a random sample of size N from the extreme value type 2 distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L6a5 **Usage:** CALL EV2RAN(N, GAMMA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EVAHF.** Compute the largest or smallest eigenvalues of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a3 **Usage:** CALL EVAHF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVASB.** Compute the largest or smallest eigenvalues of a real symmetric matrix in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a6 **Usage:** CALL EVASB (*7 parameters*) **Precision:** Single (Double: DEVASB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVASF.** Compute the largest or smallest eigenvalues of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVASF (*6 parameters*) **Precision:** Single (Double: DEVASF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVBHF.** Compute the eigenvalues in a given range of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a3 **Usage:** CALL EVBHF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVBSB.** Compute the eigenvalues in a given range of a real symmetric matrix stored in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a6 **Usage:** CALL EVBSB (*9 parameters*) **Precision:** Single (Double: DEVBSB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVBSF.** Compute the eigenvalues in a given range of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVBSF (*8 parameters*) **Precision:** Single (Double: DEVBSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCCG.** Compute all of the eigenvalues and eigenvectors of a complex matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a4 **Usage:** CALL EVCCG (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCCH.** Compute all of the eigenvalues and eigenvectors of a complex upper Hessenberg matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4c2b **Usage:** CALL EVCCH (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCHF.** Compute all of the eigenvalues and eigenvectors of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a3 **Usage:** CALL EVCHF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCRG.** Compute all of the eigenvalues and eigenvectors of a real matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a2 **Usage:** CALL EVCRG (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCRH.** Compute all of the eigenvalues and eigenvectors of a real upper Hessenberg matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4c2b **Usage:** CALL EVCRH (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCSB.** Compute all of the eigenvalues and eigenvectors of a real symmetric matrix in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a6 **Usage:** CALL EVCSB (*7 parameters*) **Precision:** Single (Double: DEVCSB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVCSF.** Compute all of the eigenvalues and eigenvectors of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVCSF (*6 parameters*) **Precision:** Single (Double: DEVCSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVEHF.** Compute the largest or smallest eigenvalues and the corresponding eigenvectors of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a3 **Usage:** CALL EVEHF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVESB.** Compute the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a6 **Usage:** CALL EVESB (*9 parameters*) **Precision:** Single (Double: DEVESB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVESF.** Compute the largest or smallest eigenvalues and the corresponding eigenvectors of a real symmetric matrix.

*Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVESF (*8 parameters*) **Precision:** Single (Double: DEVESF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVFHF.** Compute the eigenvalues in a given range and the corresponding eigenvectors of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVFHF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVFSB.** Compute the eigenvalues in a given range and the corresponding eigenvectors of a real symmetric matrix stored in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a6 **Usage:** CALL EVFSB (*11 parameters*) **Precision:** Single (Double: DEVFSB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVFSF.** Compute the eigenvalues in a given range and the corresponding eigenvectors of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVFSF (*10 parameters*) **Precision:** Single (Double: DEVFSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLCG.** Compute all of the eigenvalues of a complex matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a4 **Usage:** CALL EVLCG (N, A, LDA, EVAL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLCH.** Compute all of the eigenvalues of a complex upper Hessenberg matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4c2b **Usage:** CALL EVLCH (N, A, LDA, EVAL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLHF.** Compute all of the eigenvalues of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a3 **Usage:** CALL EVLHF (N, A, LDA, EVAL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLRG.** Compute all of the eigenvalues of a real matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a2 **Usage:** CALL EVLRG (N, A, LDA, EVAL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLRH.** Compute all of the eigenvalues of a real upper Hessenberg matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4c2b **Usage:** CALL EVLRH (N, A, LDA, EVAL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLSB.** Compute all of the eigenvalues of a real symmetric matrix in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a6 **Usage:** CALL EVLSB (N, A, LDA, NCODA, EVAL) **Precision:** Single (Double: DEVLSB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EVLSF.** Compute all of the eigenvalues of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4a1 **Usage:** CALL EVLSF (N, A, LDA, EVAL) **Precision:** Single (Double: DEVLSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EXACT RATIONAL FIT.** Exactly fits a rational function (ratio of polynomials) to data and optionally applies that fit to a more general data set. Output includes exact fit coefficients of the specified rational function and, for the more general data set, standard deviations of the coefficients, predicted values, residuals, residual standard deviation, degrees of freedom, sum of absolute deviations, and lack-of-fit analysis if there is replication. *Command(s) in DATAPLOT interactive system.* **Classes:** E1c **Usage:** EXACT <DEGREE OF NUMERATOR>/<DEGREE OF DENOMINATOR> RATIONAL FIT <ABBREVIATED RESPONSE VARIABLE> <ABBREVIATED INDEPENDENT VARIABLE> [<RESPONSE VARIABLE> <INDEPENDENT VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**EXINT.** Computes sequences of exponential integrals E(n+k,x) k=0,...,m-1 or exp(x) times same to specified tolerance. *Portable Fortran subroutine in CMLIB (AMOSLIB sublibrary).* **Classes:** C5 **Usage:** CALL EXINT(X, N, KODE, TOL, EN, IERR) **Precision:** Single (Double: DEXINT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EXP.** exp(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** R = EXP(X) **Precision:** Single (Double: DEXP) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EXP.** exp(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** R = EXP(X) **Precision:** Single (Double: DEXP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EXPCDF.** Computes the cumulative distribution function value for the exponential distribution with mean 1 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1e **Usage:** CALL EXPCDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXPINT.** Exponential integrals and scaled exponential integrals. *Portable Fortran software in SPECFN library.* **Classes:** C5 **Usage:** CALL EXPINT (RN, X, ENX, EXPENX, IERR) **Precision:** Double **Availability:** 855NOS (In source form only.)

**EXPONENTIAL PROBABILITY PLOT.** Generates a probability plot for the exponential distribution with mean 1 and standard deviation 1. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2e **Usage:** EXPONENTIAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**EXPPDF.** Computes the probability density function value for the exponential distribution with mean 1 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1e **Usage:** CALL EXPPDF(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXPPLT.** Generates an exponential probability plot with mean 1 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2e **Usage:** CALL EXPPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXPPPF.** Computes the percent point function value for the exponential distribution with mean 1 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2e **Usage:** CALL EXPPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXPRAN.** Generates a random sample of size N from the exponential distribution with mean 1 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L6a5 **Usage:** CALL EXPRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXPREL.** (exp(x)-1)/x. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4b **Usage:** R = EXPREL(X) **Precision:** Single (Double: DEXPRL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EXPRL.** (exp(x)-1)/x. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4b **Usage:** R = EXPRL(X) **Precision:** Single (Double: DEXPRL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**EXPSF.** Computes the sparsity function value for the exponential distribution with mean 1 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2e **Usage:** CALL EXPSF(P, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXSAR.** Produces exact maximum likelihood estimates of the parameters of a scalar AR-model. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2c2 **Precision:** Single **Availability:** 855NOS.

**EXTREM.** Performs an extreme value analysis on the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a4e **Usage:** CALL EXTREM(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**EXTREME VALUE TYPE 1 PROBABILITY PLOT.** Generate a probability plot for the extreme value type 1 distribution with mean Euler's number 0.57721566 and standard deviation $\pi/\sqrt{6}$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2e **Usage:** EXTREME VALUE TYPE 1 PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**EXTREME VALUE TYPE 2 PPCC PLOT.** Generates a probability plot correlation coefficient plot for the extreme value type 2 distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging from 1 to 100 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3e **Usage:** EXTREME VALUE TYPE 2 PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**EXTREME VALUE TYPE 2 PROBABILITY PLOT.** Generates a probability plot for the extreme value type 2 distribution with tail parameter $\gamma$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2e **Usage:** EXTREME VALUE TYPE 2 PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**EXTRMD.** *Double precision version of EXTRMR.*

**EXTRMI.** Finds extremal points of an integer function defined on a mesh. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a2, G1a2, N5a **Usage:** CALL EXTRMI (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 205.

**EXTRMR.** Finds extremal points of a real function defined on a mesh. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a2, G1a2, N5a **Usage:** CALL EXTRMR (*7 parameters*) **Precision:** Single (Double: EXTRMD) **Availability:** 855NOS, 205.

**EZFFTB.** Backward real discrete (fast) Fourier transform. Performs Fourier synthesis. Easy to use version. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a1 **Usage:** CALL EZFFTB(N, R, AZERO, A, B, WSAVE) **Also see:** EZFFTF **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EZFFTB.** Backward FFT of N data points in AZERO, A(N/2), B(N/2). Returns data in R(N). *Portable Fortran software in NMS library.* **Classes:** J1a1 **Usage:** CALL EZFFTB(N, R, AZERO, A, B, W) **Also see:** EZFFTI EZFFTF **Precision:** Single (Double: DEZFTB) **Availability:** PC.

**EZFFTF.** Forward real discrete (fast) Fourier transform. Performs Fourier analysis. Easy to use version. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a1 **Usage:** CALL EZFFTF(N, R, AZERO, A, B, WSAVE) **Also see:** EZFFTB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**EZFFTF.** Forward FFT of N data points R(N). Returns A(N/2) and B(N/2) as cosine and sine coefficients, AZERO as mean. *Portable Fortran software in NMS library.* **Classes:** J1a1 **Usage:** CALL EZFFTF(N, R, AZERO, A, B, W) **Also see:** EZFFTI EZFFTB **Precision:** Single (Double: DEZFTF) **Availability:** PC.

# F

**4-PLOT.** Produces 4 plots: $X_i$ vs. $i$, lag plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a1, L3a6, L3a7, L4a1a2n **Usage:** SEE DATAPLOT MANUAL FOR COMMAND SYNTAX. **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**F PROBABILITY PLOT.** Generates a probability plot for the F-distribution with degrees of freedom parameters $\nu_1$ and $\nu_2$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2f **Usage:** F PROBABILITY PLOT [VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**F01AAF.** Calculates the approximate inverse of a real matrix by Crout's method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F01AAF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01ABF.** Calculates the accurate inverse of a real symmetric positive definite matrix by Cholesky's method and iterative refinement. (Simplified parameter list). *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F01ABF (A, IA, N, B, IB, Z, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01ACF.** Calculates the accurate inverse of a real symmetric positive definite matrix by Cholesky's method and iterative refinement. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F01ACF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01ADF.** Calculates the approximate inverse of a real symmetric positive definite matrix by Cholesky's method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F01ADF (N, A, IA, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AEF.** Reduces the generalised eigenproblem $Ax = \lambda Bx$ to the standard symmetric eigenproblem $Pz = \lambda z$, where A is a real symmetric matrix and B is a real symmetric positive definite matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1c **Usage:** CALL F01AEF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AFF.** Derives eigenvectors x of the eigenproblems $Ax = \lambda Bx$, $ABx = \lambda x$ and $yBA = \lambda y$, where y is the transpose of x, from the corresponding eigenvectors $z = Mx$, where M is the transpose of the lower triangular matrix L of the derived standard symmetric eigenproblems. Matrices A and B are real and symmetric. In addition B is positive definite. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01AFF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AGF.** Gives the Householder reduction of a real symmetric matrix to tridiagonal form for use with F02BEF, F02AVF and F02BFF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b1 **Usage:** CALL F01AGF (N, TOL, A, IA, D, E, E2) **Also see:** F02BEF F02AVF F02BFF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AHF.** Derives eigenvectors of a real symmetric matrix from eigenvectors of the tridiagonal form where the tridiagonal matrix was produced by F01AGF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01AHF (*8 parameters*) **Also see:** F01AGF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01AJF.** Gives the Householder reduction of a real symmetric matrix A to tridiagonal form for use in F02AMF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b1 **Usage:** CALL F01AJF (*8 parameters*) **Also see:** F02AMF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AKF.** Reduces a real unsymmetric matrix to upper Hessenberg form. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b2 **Usage:** CALL F01AKF (N, K, L, A, IA, INTGER) **Also see:** F01APF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01ALF.** Transforms eigenvectors of a Hessenberg matrix to those of a real unsymmetric matrix from which the Hessenberg matrix has previously been derived. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01ALF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AMF.** Reduces a complex unsymmetric matrix to complex upper Hessenberg form. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b2 **Usage:** CALL F01AMF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01ANF.** Transforms eigenvectors of a complex upper Hessenberg matrix to those of a complex unsymmetric matrix from which the Hessenberg matrix has previously been derived. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01ANF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01APF.** Forms the matrix of accumulated transformations from information left by F01AKF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01APF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01ATF.** Balances a real unsymmetric matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1a **Usage:**

CALL F01ATF (N, IB, A, IA, K, L, D) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01AUF.** Transforms eigenvectors of a balanced matrix to those of the original real unsymmetric matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01AUF (N, K, L, M, D, Z, IZ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01AVF.** Balances a complex matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1a **Usage:** CALL F01AVF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01AWF.** Transforms eigenvectors of a balanced matrix to those of the original complex matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01AWF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01AXF.** Reduces an m-by-n real matrix, m≥n, to upper triangular form for use in F04AMF and F04ANF. The routine uses Householder transformations with column pivoting. *Proprietary Fortran subroutine in NAG.* **Classes:** D5 **Usage:** CALL F01AXF (*9 parameters*) **Also see:** F94AMF F04ANF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01AYF.** Gives the Householder reduction of a real symmetric matrix A to tridiagonal form for use in F02BEF, F02AVF and F02BFF. The routine is similar to F01AGF but is more economical in storage. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b1 **Usage:** CALL F01AYF (N, TOL, A, IA, D, E, E2) **Also see:** F02BEF F02AVF F02BFF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01AZF.** Derives eigenvectors of a real symmetric matrix from eigenvectors of the tridiagonal form produced by F01AYF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01AZF (N, M1, M2, A, IA, Z, IZ) **Also see:** F01AYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BCF.** Gives the Householder reduction of a complex Hermitian matrix to tridiagonal form for use in F02AVF or F02AYF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b1 **Usage:** CALL F01BCF (*10 parameters*) **Also see:** F01AVF F02AYF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01BDF.** Reduces the eigenproblems $ABx = \lambda x$, $x^T BA = \lambda x^T$, $BAy = \lambda y$ and $y^T AB = \lambda y^T$ to the standard symmetric eigenproblem $Qz = \lambda z$, where A is a real symmetric matrix and B is a real symmetric positive definite matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1c **Usage:** CALL F01BDF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BEF.** Derives eigenvectors y of the problems $y^T AB = \lambda y^T$ and $BAy = \lambda y$ from the corresponding eigenvectors of the derived standard symmetric eigenproblem. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c4 **Usage:** CALL F01BEF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BLF.** Calculates the rank and pseudo-inverse of an m-by-n real matrix A, m≥n, rank(A)≤n, using Householder's factorisation. *Proprietary Fortran subroutine in NAG.* **Classes:** D9c **Usage:** CALL F01BLF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BNF.** Performs the Cholesky decomposition of a complex positive definite Hermitian matrix, given the lower triangle of the matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2d1b **Usage:** CALL F01BNF (N, A, IA, P, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BPF.** Determines the inverse of a complex positive definite Hermitian matrix, given the lower triangle of the matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2d1b **Usage:** CALL F01BPF (N, A, IA, V, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BQF.** Forms the Cholesky decomposition of a real symmetric matrix G whose lower triangle only is stored. If G is not positive definite, the routine forms the Cholesky decomposition of G + E where E is a diagonal matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F01BQF (N, EPS, RL, IRL, D, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BRF.** Decomposes a real sparse matrix. The routine either forms the LU-decomposition of a permutation of the entire matrix, or, optionally, first permutes the matrix to block lower triangular form and then only decomposes the diagonal blocks. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a4 **Usage:** CALL F01BRF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BSF.** Decomposes a real sparse matrix using the pivotal sequence previously obtained by F01BRF when a matrix of the same sparsity pattern was decomposed. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a4 **Usage:** CALL F01BSF (*16 parameters*) **Also see:** F01BRF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BTF.** Decomposes a real matrix into a product of triangular matrices LU by Gaussian elimination with partial pivoting. The block-column method used is designed for efficient working on paged virtual machines. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F01BTF (N, A, IA, P, DP, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BUF.** Decomposes a symmetric positive definite band matrix into the form $ULDL^T U^T$ where U is a unit upper triangular matrix, L is unit lower triangular and D is diagonal. It is specifically designed to precede F01BVF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F01BUF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BVF.** Transforms the generalised symmetric eigenproblem $Ax = \lambda Bx$ to the equivalent standard eigenproblem $Cy = \lambda y$ where A, B and C are symmetric band matrices and B is positive definite. B must have been decomposed by

F01BUF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1c **Usage:** CALL F01BVF (*13 parameters*) **Also see:** F01BUF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BWF.** Reduces a symmetric band matrix to tridiagonal form. This routine will normally be used in conjunction with F02AVF to find all the eigenvalues of A. For selected eigenvalues there is a choice between F02BMF and the combination of F01BWF and F02BFF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c1b1 **Usage:** CALL F01BWF (N, M1, A, IA, D, E) **Also see:** F02AVF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01BXF.** Performs the Cholesky factorization $U^T U$ of a real symmetric positive definite matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F01BXF (N, A, IA, P, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CAF.** Sets elements of an m-by-n matrix A to zero. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b1 **Usage:** CALL F01CAF (A, M, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CBF.** Sets the elements A(i,j) to one if i=j and zero otherwise, where $1 \leq i \leq m$ and $1 \leq j \leq n$. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b1 **Usage:** CALL F01CBF (A, M, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CDF.** Adds elements of the m-by-n matrices B and C and stores the results in elements of the matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F01CDF (A, B, C, M, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CEF.** Subtracts elements of the matrix C from elements of the matrix B and stores the results in elements of the matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F01CEF (A, B, C, M, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CFF.** Copies elements of the matrix B into different positions in the matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b8 **Usage:** CALL F01CFF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CGF.** Adds elements of the matrix B to elements in different positions in the matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F01CGF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CHF.** Subtracts elements of the matrix B from elements in a different position in the matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F01CHF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CKF.** Returns with the result of the multiplication of two matrices B and C in the matrix A, with the option to overwrite B or C. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b6 **Usage:** CALL F01CKF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01CLF.** Post-multiplies the matrix B with the transpose of the matrix C and places the result in the matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b6 **Usage:** CALL F01CLF (A, B, C, N, P, M, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F01CMF.** Copies elements of one matrix into a second matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b8 **Usage:** CALL F01CMF (A, LA, B, LB, M, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CNF.** Copies a vector of length M into a row of a matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5, D1b8 **Usage:** CALL F01CNF (V, M, A, LA, I) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CPF.** Copies the contents of a vector into a second vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F01CPF (A, B, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CQF.** Sets the elements of a vector to zero. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a1 **Usage:** CALL F01CQF (A, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CRF.** Re-orders the elements of a vector of length mn, containing an m-by-n matrix, A, so that the new vector contains the transpose matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b3 **Usage:** CALL F01CRF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01CSF.** Forms the product c = Ab where b is a vector and A is a symmetric matrix whose lower triangle is stored by rows in a one-dimensional array. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F01CSF (A, LA, B, N, C) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01DAF.** Returns the sum of an initial value and a scalar product, using basic precision arithmetic. *Proprietary Fortran function in NAG.* **Classes:** D1a4 **Usage:** R = F01DAF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01DBF.** Returns the sum of an initial value and a scalar product, using additional precision arithmetic. *Proprietary Fortran function in NAG.* **Classes:** D1a4 **Usage:** R = F01DBF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01DCF.** Computes the value of a complex scalar product and subtracts it from a complex initial value, using basic precision arithmetic. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a4 **Usage:** CALL F01DCF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01DDF.** Computes the value of a complex scalar product and subtracts it from a complex initial value, using additional precision arithmetic. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a4 **Usage:** CALL F01DDF

(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01DEF.** Returns the value of the scalar product of two arrays of length N, using basic precision arithmetic. *Proprietary Fortran function in NAG.* **Classes:** D1a4 **Usage:** R = F01DEF(A, B, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01LBF.** Decomposes a general real band matrix of order N, with M1 sub-diagonals and M2 super-diagonals, where M1+M2+1 is much less than N, into triangular matrices using Gaussian elimination with partial pivoting. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a2 **Usage:** CALL F01LBF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01LEF.** Factorises a real tridiagonal matrix into triangular factors, T-λI=PLU, using Gaussian elimination with partial pivoting. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a2a **Usage:** CALL F01LEF (*9 parameters*) **Also see:** F04LEF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01LZF.** Factorises a real upper triangular matrix A as $QCP^T$ where Q and P are orthogonal and C is upper bidiagonal. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a3 **Usage:** CALL F01LZF (*21 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01MAF.** Find an incomplete Cholesky factorisation of a sparse symmetric positive definite matrix A. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b4 **Usage:** CALL F01MAF (*15 parameters*) **Also see:** F04MAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01MCF.** Computes the Cholesky factorisation of a symmetric, positive-definite, variable-bandwidth matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b2 **Usage:** CALL F01MCF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01NAF.** Finds the LU factorisation of a complex band matrix by Gaussian elimination with partial pivoting. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c2 **Usage:** CALL F01NAF (*9 parameters*) **Also see:** F04NAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01QAF.** Factorises a real matrix A as QU where Q is an m-by-m orthogonal matrix, and U is an m-by-n matrix which is zero except for the elements on and above the leading diagonal. *Proprietary Fortran subroutine in NAG.* **Classes:** D5 **Usage:** CALL F01QAF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F01QBF.** Factorises a real m-by-n (m≤n) matrix A as UQ where Q is an n-by-n orthogonal matrix, and U is an m-by-n matrix which is zero except for the elements on and above the leading diagonal in the first m columns. *Proprietary Fortran subroutine in NAG.* **Classes:** D5 **Usage:** CALL F01QBF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AAF.** Calculates all the eigenvalues of a real symmetric matrix by Householder reduction and the QL algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a1 **Usage:** CALL F02AAF (A, IA, N, R, E, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02ABF.** Calculates all the eigenvalues and eigenvectors of a real symmetric matrix by Householder reduction and the QL algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a1 **Usage:** CALL F02ABF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02ADF.** Calculates all the eigenvalues of Ax=λBx, where A is a real symmetric matrix and B is a real symmetric positive definite matrix, using Householder reduction and the QL algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4b1 **Usage:** CALL F02ADF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AEF.** Calculates all the eigenvalues and eigenvectors of Ax=λBx, where A is a real symmetric matrix and B is a real symmetric positive definite matrix, using Householder reduction and the QL algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4b1 **Usage:** CALL F02AEF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AFF.** Calculates all the eigenvalues of a real unsymmetric matrix by reduction to Hessenberg form and the QR algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a2 **Usage:** CALL F02AFF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AGF.** Calculates all the eigenvalues and eigenvectors of a real unsymmetric matrix by reduction to Hessenberg form and the QR algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a2 **Usage:** CALL F02AGF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AJF.** Calculates all the eigenvalues of a complex matrix by reduction to upper Hessenberg form and the LR algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a4 **Usage:** CALL F02AJF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AKF.** Calculates all the eigenvalues and eigenvectors of a complex matrix by reduction to upper Hessenberg form and the LR algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a4 **Usage:** CALL F02AKF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AMF.** Calculates all the eigenvalues and eigenvectors of a real symmetric tridiagonal matrix or of a full real symmetric matrix that has been reduced to tridiagonal form using F01AJF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c2a **Usage:** CALL F02AMF (*7 parameters*) **Also see:** F01AJF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F02ANF.** Calculates all the eigenvalues of a complex upper Hessenberg matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c2b **Usage:** CALL F02ANF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02APF.** Calculates all the eigenvalues of a real upper Hessenberg matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c2b **Usage:** CALL F02APF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F02AQF.** Calculates all the eigenvalues and eigenvectors of a real upper Hessenberg matrix or a real unsymmetric matrix which has been reduced to upper Hessenberg form by F01AKF and F01APF and which may have been balanced using F01ATF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c2b **Usage:** CALL F02AQF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F02ARF.** Calculates all the eigenvalues and eigenvectors of either a complex upper Hessenberg matrix or a complex full matrix which has been reduced to upper Hessenberg form by F01AMF and may have been balanced using F01AVF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c2b **Usage:** CALL F02ARF (*16 parameters*) **Also see:** F01AMF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F02AVF.** Calculates all the eigenvalues of a real symmetric tridiagonal matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a5 **Usage:** CALL F02AVF (N, EPS, D, E, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AWF.** Calculates all the eigenvalues of a complex Hermitian matrix by reduction to real symmetric tridiagonal form and the QL algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a3 **Usage:** CALL F02AWF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AXF.** Calculates all the eigenvalues and eigenvectors of a complex Hermitian matrix by reduction to real symmetric tridiagonal form and the QL algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a3 **Usage:** CALL F02AXF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02AYF.** Calculates all the eigenvalues and eigenvectors of a complex Hermitian matrix which has been reduced to real symmetric tridiagonal form using F01BCF. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c2a **Usage:** CALL F02AYF (*9 parameters*) **Also see:** F01BCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BBF.** Calculates selected eigenvalues and eigenvectors of a real symmetric matrix by Householder reduction, by the method of bisection and by inverse iteration, where the selected eigenvalues lie between two given values. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a1 **Usage:** CALL F02BBF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BCF.** Calculates selected eigenvalues and eigenvectors of a real unsymmetric matrix by reduction to Hessenberg form, the QR algorithm and inverse iteration, where the moduli of the selected eigenvalues lie between two given values. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a2 **Usage:** CALL F02BCF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BDF.** Calculates selected eigenvalues and eigenvectors of a complex matrix by reduction to Hessenberg form, the LR algorithm and inverse iteration, where the moduli of the selected eigenvalues lie between two given values. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a4 **Usage:** CALL F02BDF (*24 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BEF.** Calculates selected eigenvalues and eigenvectors of a real symmetric tridiagonal matrix, where the selected eigenvalues lie between two given values. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a5 **Usage:** CALL F02BEF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BFF.** Calculates selected eigenvalues of a real symmetric tridiagonal matrix, where, if the eigenvalues are numbered in ascending order, the numbers of the first and last eigenvalues required are given. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a5 **Usage:** CALL F02BFF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BJF.** Calculates all the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax=\lambda Bx$ where A and B are real, square matrices, using the QZ algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4b2 **Usage:** CALL F02BJF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BKF.** Calculates selected eigenvectors of a real upper Hessenberg matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c3 **Usage:** CALL F02BKF (*14 parameters*) **Also see:** F02AKF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02BLF.** Calculates selected eigenvectors of a complex upper Hessenberg matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D4c3 **Usage:** CALL F02BLF (*20 parameters*) **Also see:** F02AKF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02FHF.** Finds the eigenvalues of the generalised band symmetric eigenvalue problem $Ax=\lambda Bx$, where A and B are symmetric band matrices and B is positive definite. *Proprietary Fortran subroutine in NAG.* **Classes:** D4b5 **Usage:** CALL F02FHF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02FJF.** Finds eigenvalues and eigenvectors of a real sparse symmetric or generalised symmetric eigenvalue problem. *Proprietary Fortran subroutine in NAG.* **Classes:** D4a7, D4b1 **Usage:** CALL F02FJF (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02GJF.** Calculates all the eigenvalues and, if required, all the eigenvectors of the complex generalized eigenproblem $Ax=\lambda Bx$ where A and B are complex, square matrices, using the QZ algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D4b4 **Usage:** CALL F02GJF (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02SDF.** Finds the eigenvector corresponding to a given real eigenvalue for the generalised problem $Ax=\lambda Bx$, or for the standard problem $Ax=\lambda x$, where A and B are real band matrices. *Proprietary Fortran subroutine in NAG.* **Classes:** D4b5 **Usage:** CALL F02SDF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02SZF.** Provides a singular value decomposition of a real n-by-n upper bidiagonal matrix A into $QDP^T$, where Q and P are n-by-n orthogonal matrices, and $D = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ with $\sigma_1 \geq \sigma_2 \geq \ldots, \sigma_n \geq 0$, these being the singular values of A. *Proprietary Fortran subroutine in NAG.* **Classes:** D6 **Usage:** CALL F02SZF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02WAF.** Finds the singular values and the right-hand singular vectors (principal components) of a real rectangular m-by-n matrix A, where $m \geq n$. *Proprietary Fortran subroutine in NAG.* **Classes:** D6 **Usage:** CALL F02WAF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02WBF.** Finds the singular values and right-hand singular vectors (principal components) of a real rectangular m-by-n matrix A, where $m \leq n$. *Proprietary Fortran subroutine in NAG.* **Classes:** D6 **Usage:** CALL F02WBF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02WCF.** Computes the singular values and left- and right-hand singular vectors of a real rectangular m-by-n matrix A, $A = QDP^T$, where $Q^TQ = P^TP = I_k$, $k = \min(m,n)$ and $D = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k)$ with $\sigma_1 \geq \sigma_2 \geq \ldots, \geq \sigma_k \geq 0$. *Proprietary Fortran subroutine in NAG.* **Classes:** D6 **Usage:** CALL F02WCF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F02WDF.** Returns the Householder QU factorisation of a real rectangular m-by-n ($m \geq n$) matrix A. Further, on request or if A is not of full rank, part or all of the singular value decomposition of A is returned. *Proprietary Fortran subroutine in NAG.* **Classes:** D6 **Usage:** CALL F02WDF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F03AAF.** Calculates the determinant of a real matrix using the factorisation method of Crout. *Proprietary Fortran subroutine in NAG.* **Classes:** D3a1 **Usage:** CALL F03AAF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F03ABF.** Calculates the determinant of a real symmetric positive definite matrix using Cholesky decomposition. *Proprietary Fortran subroutine in NAG.* **Classes:** D3b1b **Usage:** CALL F03ABF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F03ACF.** Calculates the determinant of a real symmetric positive definite band matrix using Cholesky decomposition. *Proprietary Fortran subroutine in NAG.* **Classes:** D3b2 **Usage:** CALL F03ACF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F03ADF.** Calculates the determinant of a complex matrix using the factorisation method of Crout. *Proprietary Fortran subroutine in NAG.* **Classes:** D3c1 **Usage:** CALL F03ADF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F03AEF.** Decomposes a real symmetric positive definite matrix A into triangular matrices $LL^T$ using Cholesky's method, and evaluates the determinant. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b, D3b1b **Usage:** CALL F03AEF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F03AFF.** Decomposes a real matrix into triangular factor matrices LU by Crout's method and evaluates the determinant. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1, D3a1 **Usage:** CALL F03AFF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F03AGF.** Decomposes a real symmetric positive definite band matrix into triangular matrices using Cholesky's method and evaluates the determinant. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b2, D3b2 **Usage:** CALL F03AGF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F03AHF.** Decomposes a complex matrix into triangular factor matrices LU by Crout's method and evaluates the determinant. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c1, D3c1 **Usage:** CALL F03AHF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F03AMF.** Finds the determinant of a complex positive definite Hermitian matrix A, following a Cholesky decomposition of A as given by F01BNF. *Proprietary Fortran subroutine in NAG.* **Classes:** D3d1b **Usage:** CALL F03AMF (N, TEN, P, D1, D2) **Also see:** F01BNF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AAF.** Calculates the approximate solution of a set of real linear equations with multiple right hand sides by Crout's factorisation method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04AAF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ABF.** Calculates the accurate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, by Cholesky's decomposition method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F04ABF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ACF.** Calculates the approximate solution of a set of real symmetric positive definite band equations with multiple right hand sides by Cholesky's decomposition method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b2

Usage: CALL F04ACF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ADF.** Calculates the approximate solution of a set of complex linear equations with multiple right hand sides by Crout's factorisation method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c1 **Usage:** CALL F04ADF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AEF.** Calculates the accurate solution of a set of real linear equations with multiple right hand sides by Crout's factorisation method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04AEF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AFF.** Calculates the accurate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices $A = LL^T$ using F03AEF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F04AFF (*14 parameters*) **Also see:** F03AEF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AGF.** Calculates the approximate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices $A = LL^T$ using F03AEF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F04AGF (*9 parameters*) **Also see:** F03AEF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F04AHF.** Calculates the accurate solution of a set of real linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F03AFF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04AHF (*16 parameters*) **Also see:** F03AFF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AJF.** Calculates the approximate solution of a set of real linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F03AFF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04AJF (N, IR, A, IA, P, B, IB) **Also see:** F03AFF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F04AKF.** Calculates the approximate solution of a set of complex linear equations with multiple right hand sides AX=B where A has been decomposed into triangular matrices using F03AHF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c1 **Usage:** CALL F04AKF (N, IR, A, IA, P, B, IB) **Also see:** F03AHF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F04ALF.** Calculates the approximate solution of a set of real symmetric positive definite band linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F03AGF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b2 **Usage:** CALL F04ALF (*10 parameters*) **Also see:** F03AGF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AMF.** Calculates the accurate least squares solution of a set of m linear equations in n unknowns, $m \geq n$ and rank=n with multiple right hand sides, AX=B. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a1 **Usage:** CALL F04AMF (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ANF.** Calculates the approximate least squares solution of a set of m linear equations in n unknowns, $m \geq n$ and rank=n with a single right hand side, Ax=b, where A has been decomposed into triangular matrices using F01AXF. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a1 **Usage:** CALL F04ANF (*9 parameters*) **Also see:** F01AXF **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 205:* vectorized.

**F04AQF.** Calculates the approximate solution of a set of real symmetric positive definite linear equations with a single right hand side Ax=b where A has been decomposed into $LDL^T$ using F01BQF. (Economical storage). *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F04AQF (N, M, RL, D, B, X) **Also see:** F01BQF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ARF.** Calculates the approximate solution of a set of real linear equations with a single right hand side, Ax=b, by Crout's factorisation method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04ARF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ASF.** Calculates the accurate solution of a set of real symmetric positive definite linear equations with a single right hand side, Ax=b, by Cholesky's decomposition method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F04ASF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04ATF.** Calculates the accurate solution of a set of real linear equations with a single right hand side, Ax=b, by Crout's factorisation method. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04ATF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AWF.** Calculates the approximate solutions of a set of complex linear equations with multiple right hand sides, AX=B, where A is positive definite Hermitian, following the Cholesky decomposition of A by F01BNF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2d1b **Usage:** CALL F04AWF (*9 parameters*) **Also see:** F01BNF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AXF.** Calculates the approximate solution of a set of real sparse linear equations with a single right hand side, $Ax = b$ or $A^T x = b$, where A has been decomposed by F01BRF or F01BSF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a4 **Usage:** CALL F04AXF (*10 parameters*) **Also see:** F01BRF F01BSF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AYF.** Calculates the approximate solution of a set of real linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F01BTF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a1 **Usage:** CALL F04AYF (*8 parameters*) **Also see:** F01BTF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04AZF.** Calculates the approximate solution of a set of real symmetric positive definite linear equations with multiple right hand sides, AX=B, where A has been decomposed into triangular matrices using F01BXF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b1b **Usage:** CALL F04AZF (*8 parameters*) **Also see:** F01BXF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04EAF.** Calculates the approximate solution of a set of real tridiagonal linear equations with a single right hand side, Tx = b. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a2a **Usage:** CALL F04EAF (N, D, DU, DL, B, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04FAF.** Calculates the approximate solution of a set of real symmetric positive definite tridiagonal linear equations. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b2a **Usage:** CALL F04FAF (JOB, N, D, E, B, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04JAF.** Finds the minimal solution of a linear least squares problem, Ax=b, where A is a real m-by-n ($m \geq n$) matrix and b is an m element vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a1 **Usage:** CALL F04JAF (*11 parameters*) **Also see:** F04YAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04JDF.** Finds the minimal solution of a linear least squares problem, Ax=b, where A is a real m-by-n ($m \leq n$) matrix and b is an m element vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a1 **Usage:** CALL F04JDF (*11 parameters*) **Also see:** F04YAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04JGF.** Finds the solution of a linear least squares problem, Ax=b, where A is a real m-by-n ($m \geq n$) matrix and b is an m element vector. If the matrix of observations is not of full rank, then the minimal least squares solution is returned. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a1 **Usage:** CALL F04JGF (*12 parameters*) **Also see:** F04YAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04LDF.** Calculates the approximate solution of a set of general real band linear equations with multiple right-hand sides AX=B where A has been decomposed into triangular matrices using F01LBF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a2 **Usage:** CALL F04LDF (*12 parameters*) **Also see:** F01LBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04LEF.** Solves a system of tridiagonal equations following the factorisation by F01LEF. This routine is intended for applications such as inverse iteration as well as straightforward linear equation applications. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a2a **Usage:** CALL F04LEF (*10 parameters*) **Also see:** F01LEF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04MAF.** Solves a sparse symmetric positive definite system of linear equations, Ax = b, using a pre-conditioned conjugate gradient method, following a factorization of A by F01MAF. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b4 **Usage:** CALL F04MAF (*15 parameters*) **Also see:** F01MAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04MBF.** Solves a system of sparse symmetric linear equations $(A-\lambda I)x=b$, using a Lanczos algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b4 **Usage:** CALL F04MBF (*22 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04MCF.** Computes the approximate solution of a system of real linear equations with multiple right hand sides, Ax = B, where A is a symmetric positive-definite variable-bandwidth matrix, which has previously been factorised by F01MCF. Related systems may also be solved. *Proprietary Fortran subroutine in NAG.* **Classes:** D2b2 **Usage:** CALL F04MCF (*12 parameters*) **Also see:** F01MCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04NAF.** Solves a system of complex band equations following factorisation by F01NAF. This routine is intended for applications such as inverse iteration as well as straightforward linear equation applications. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c2 **Usage:** CALL F04NAF (*10 parameters*) **Also see:** F01NAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04QAF.** Solves sparse unsymmetric equations, sparse linear least squares problems and sparse damped linear least squares problems using a Lanczos algorithm. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a4, D9a1 **Usage:** CALL F04QAF (*25 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F04YAF.** Returns elements of the estimated variance-covariance matrix of the sample regression coefficients for the solution of a linear least squares problem. *Proprietary Fortran subroutine in NAG.* **Classes:** D9a1 **Usage:** CALL F04YAF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F05AAF.** Applies the Schmidt orthogonalisation process to n vectors in m dimensional space, $n \leq m$. *Proprietary Fortran subroutine in NAG.* **Classes:** D5 **Usage:** CALL F05AAF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F05ABF.** Approximate 2-norm of a vector. *Proprietary Fortran function in NAG.* **Classes:** D1a3b **Usage:** R = F05ABF(X, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06AAF.** Generate real plane rotation. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06AAF( A, B, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SROTG.

**F06BAF.** Generate real plane rotation, storing tangent. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06BAF( A, B, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BCF.** Recover real cosine c and real sine s from given real tangent t, i.e., $c = 1/\sqrt{1+t^2}$ and s=tc. *Proprietary Fortran subroutine in NAG.* **Classes:** C4a **Usage:** CALL F06BCF( T, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BEF.** Generate real Jacobi plane rotation. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06BEF( JOB, X, Y, Z, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BHF.** Apply real similarity rotation to 2-by-2 symmetric matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06BHF( X, Y, Z, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BLF.** Quotient of real scalars, with overflow flag. *Proprietary Fortran function in NAG.* **Classes:** A3a **Usage:** R = F06BLF( A, B, FAIL ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BMF.** Compute Euclidean norm from scaled form. *Proprietary Fortran function in NAG.* **Classes:** D1a3b **Usage:** R = F06BMF( SCALE, SSQ ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BNF.** Compute square root of $(a^2 + b^2)$, real a and b. *Proprietary Fortran function in NAG.* **Classes:** D1a3b **Usage:** R = F06BNF( A, B ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06BPF.** Compute eigenvalue of 2-by-2 real symmetric matrix. *Proprietary Fortran function in NAG.* **Classes:** D4a1 **Usage:** R = F06BPF( A, B, C ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06CAF.** Generate complex plane rotation, storing tangent, real cosine. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06CAF( A, B, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06CBF.** Generate complex plane rotation, storing tangent, real sine. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06CBF( A, B, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06CCF.** Recover real cosine c and complex sine s from given complex tangent t, i.e., $c = 1/\sqrt{1+|t|^2}$ and s=tc. *Proprietary Fortran subroutine in NAG.* **Classes:** C4a **Usage:** CALL F06CCF( T, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06CDF.** Recover complex cosine c and real sine s from given complex tangent t, i.e., $c = (\text{sign}(\text{real}(t))|t|)/(t\sqrt{1+|t|^2})$ and s = tc. *Proprietary Fortran subroutine in NAG.* **Classes:** C4a **Usage:** CALL F06CDF( T, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06CHF.** Apply complex similarity rotation to 2-by-2 Hermitian matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06CHF( X, Y, Z, C, S ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06CLF.** Quotient of complex scalars, with overflow flag. *Proprietary Fortran function in NAG.* **Classes:** A4a **Usage:** C = F06CLF( A, B, FAIL ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06DBF.** Broadcast scalar into integer vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a1 **Usage:** CALL F06DBF( N, CONST, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06DFF.** Copy integer vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F06DFF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06EAF.** Dot product of two real vectors. *Proprietary Fortran function in NAG.* **Classes:** D1a4 **Usage:** R = F06EAF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SDOT.

**F06ECF.** Add scalar times real vector to real vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a7 **Usage:** CALL F06ECF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SAXPY.

**F06EDF.** Multiply real vector by scalar. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06EDF( N, ALPHA, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSCAL.

**F06EFF.** Copy real vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F06EFF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SCOPY.

**F06EGF.** Swap two real vectors. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F06EGF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSWAP.

**F06EJF.** Compute Euclidean norm of real vector. *Proprietary Fortran function in NAG.* **Classes:** D1a3b **Usage:** R = F06EJF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SNRM2.

**F06EKF.** Sum the absolute values of real vector elements. *Proprietary Fortran function in NAG.* **Classes:** D1a3a **Usage:** R = F06EKF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SASUM.

**F06EPF.** Apply real plane rotation. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06EPF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SROT.

**F06FAF.** Compute cosine of angle between two real vectors. *Proprietary Fortran function in NAG.* **Classes:** D1a11

**Usage:** R = F06FAF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FBF.** Broadcast scalar into real vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a1 **Usage:** CALL F06FBF( N, CONST, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FCF.** Multiply real vector by diagonal matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b6 **Usage:** CALL F06FCF( N, D, INCD, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FDF.** Multiply real vector by scalar, preserving input vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06FDF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FGF.** Negate real vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06FGF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FJF.** Update Euclidean norm of real vector in scaled form. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a3b **Usage:** CALL F06FJF( N, X, INCX, SCALE, SUMSQ ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FKF.** Compute weighted Euclidean norm of real vector. *Proprietary Fortran function in NAG.* **Classes:** D1a3b **Usage:** R = F06FKF( N, W, INCW, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FLF.** Elements of real vector with largest and smallest absolute value. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a2, D1a3c, N5a **Usage:** CALL F06FLF( N, X, INCX, XMAX, XMIN ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FPF.** Apply real symmetric plane rotation to two vectors. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06FPF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FQF.** Generate sequence of real plane rotations. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06FQF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FRF.** Generate real elementary reflection, NAG style. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a9 **Usage:** CALL F06FRF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FSF.** Generate real elementary reflection, LINPACK style. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a9 **Usage:** CALL F06FSF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FTF.** Apply real elementary reflection, NAG style. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a9 **Usage:** CALL F06FTF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06FUF.** Apply real elementary reflection, LINPACK style. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a9 **Usage:** CALL F06FUF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06GAF.** Dot product of two complex vectors, unconjugated. *Proprietary Fortran function in NAG.* **Classes:** D1a4 **Usage:** C = F06GAF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CDOTU.

**F06GBF.** Dot product of two complex vectors, conjugated. *Proprietary Fortran function in NAG.* **Classes:** D1a4 **Usage:** C = F06GBF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CDOTC.

**F06GCF.** Add scalar times complex vector to complex vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a7 **Usage:** CALL F06GCF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CAXPY.

**F06GDF.** Multiply complex vector by complex scalar. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06GDF( N, ALPHA, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CSCAL.

**F06GFF.** Copy complex vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F06GFF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CCOPY.

**F06GGF.** Swap two complex vectors. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F06GGF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CSWAP.

**F06HBF.** Broadcast scalar into complex vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a1 **Usage:** CALL F06HBF( N, CONST, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HCF.** Multiply complex vector by complex diagonal matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06HCF( N, D, INCD, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HDF.** Multiply complex vector by complex scalar, preserving input vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06HDF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HGF.** Negate complex vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06HGF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HPF.** Apply complex plane rotation. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06HPF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HQF.** Generate sequence of complex plane rotations. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06HQF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HRF.** Generate complex elementary reflection. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a9 **Usage:** CALL F06HRF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06HTF.** Apply complex elementary reflection. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a9 **Usage:** CALL F06HTF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06JDF.** Multiply complex vector by real scalar. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06JDF( N, ALPHA, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CSSCAL.

**F06JJF.** Compute Euclidean norm of complex vector. *Proprietary Fortran function in NAG.* **Classes:** D1a3b **Usage:** R = F06JJF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SCNRM2.

**F06JKF.** Sum the absolute values of complex vector elements. *Proprietary Fortran function in NAG.* **Classes:** D1a3a **Usage:** R = F06JKF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SCASUM.

**F06JLF.** Index, real vector element with largest absolute value. *Proprietary Fortran function in NAG.* **Classes:** D1a2, D1a3c, N5a **Usage:** I = F06JLF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is ISAMAX.

**F06JMF.** Index, complex vector element with largest absolute value. *Proprietary Fortran function in NAG.* **Classes:** D1a2, D1a3c, N5a **Usage:** I = F06JMF( N, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is ICAMAX.

**F06KCF.** Multiply complex vector by real diagonal matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06KCF( N, D, INCD, X, INCX ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06KDF.** Multiply complex vector by real scalar, preserving input vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a6 **Usage:** CALL F06KDF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06KFF.** Copy real vector to complex vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a5 **Usage:** CALL F06KFF( N, X, INCX, Y, INCY ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06KJF.** Update Euclidean norm of complex vector in scaled form. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a3b **Usage:** CALL F06KJF( N, X, INCX, SCALE, SUMSQ ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06KLF.** Last non-negligible element of real vector. *Proprietary Fortran function in NAG.* **Classes:** D1a2, D1a11, N5a **Usage:** I = F06KLF( N, X, INCX, TOL ) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06KPF.** Apply real plane rotation to two complex vectors. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a8 **Usage:** CALL F06KPF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**F06PAF.** Matrix-vector product, real general matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PAF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SGEMV.

**F06PBF.** Matrix-vector product, real general band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PBF(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SGBMV.

**F06PCF.** Matrix-vector product, real symmetric matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PCF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSYMV.

**F06PDF.** Matrix-vector product, real symmetric band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PDF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSBMV.

**F06PEF.** Matrix-vector product, real symmetric packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PEF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSPMV.

**F06PFF.** Matrix-vector product, real triangular matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PFF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is STRMV.

**F06PGF.** Matrix-vector product, real triangular band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PGF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is STBMV.

**F06PHF.** Matrix-vector product, real triangular packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06PHF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is STPMV.

**F06PJF.** Solve system of equations, real triangular matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a3 **Usage:** CALL F06PJF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is STRSV.

**F06PKF.** Solve system of equations, real triangular band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:**

D2a3 **Usage:** CALL F06PKF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is STBSV.

**F06PLF.** Solve system of equations, real tiiangular packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2a3 **Usage:** CALL F06PLF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is STPSV.

**F06PMF.** Rank-1 update, real general matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06PMF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SGER.

**F06PPF.** Rank-1 update, real symmetric matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06PPF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSYR.

**F06PQF.** Rank-1 update, real symmetric packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06PQF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSPR.

**F06PRF.** Rank-2 update, real symmetric matrix . *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06PRF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSYR2.

**F06PSF.** Rank-2 update, real symmetric packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06PSF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is SSPR2.

**F06SAF.** Matrix-vector product, complex general matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SAF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CGEMV.

**F06SBF.** Matrix-vector product, complex general band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SBF(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CGBMV.

**F06SCF.** Matrix-vector product, complex Hermitian matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SCF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHEMV.

**F06SDF.** Matrix-vector product, complex Hermitian band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SDF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHBMV.

**F06SEF.** Matrix-vector product, complex Hermitian packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SEF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHPMV.

**F06SFF.** Matrix-vector product, complex triangular matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SFF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CTRMV.

**F06SGF.** Matrix-vector product, complex triangular band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SGF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CTBMV.

**F06SHF.** Matrix-vector product, complex triangular packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b4 **Usage:** CALL F06SHF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CTPMV.

**F06SJF.** Solve system of equations, complex triangular matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c3 **Usage:** CALL F06SJF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CTRSV.

**F06SKF.** Solve system of equations, complex triangular band matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c3 **Usage:** CALL F06SKF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CTBSV.

**F06SLF.** Solve system of equations, complex triangular packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D2c3 **Usage:** CALL F06SLF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CTPSV.

**F06SMF.** Rank-1 update, complex general matrix, unconjugated vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06SMF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CGERU.

**F06SNF.** Rank-1 update, complex general matrix, conjugated vector. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06SNF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CGERC.

**F06SPF.** Rank-1 update, complex Hermitian matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Us-**

age: CALL F06SPF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHER.

**F06SQF.** Rank-1 update, complex Hermitian packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06SQF(*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHPR.

**F06SRF.** Rank-2 update, complex Hermitian matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06SRF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHER2.

**F06SSF.** Rank-2 update, complex Hermitian packed matrix. *Proprietary Fortran subroutine in NAG.* **Classes:** D1b5 **Usage:** CALL F06SSF(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205. *On 855NOS, 205, 855VE:* Alternate name is CHPR2.

**F1A1A.** Finds the zeros of a polynomial with real coefficients. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F1a1 **Usage:** CALL F1A1A(N, A, R, T, IFLG, S, KE) **Precision:** Single **Availability:** PC.

**F1A1B.** Finds the zeros of a polynomial with complex coefficients. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F1a2 **Usage:** CALL F1A1B(NZ, A, R, T, IFLG, S, KE) **Precision:** Single **Availability:** PC.

**F1A1E.** Evaluates a complex polynomial and its derivatives, with optional error bounds. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** C3b, F3 **Usage:** CALL F1A1E(N, M, A, Z, B, KBD) **Precision:** Single **Availability:** PC.

**F1A21.** Searches for a zero of a user-defined function between given input values until the width of the interval has collapsed to within a tolerance set by the user. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F1b **Usage:** CALL F1A21(F, B, C, RE, AE, KE) **Precision:** Single (Double: F1A21D) **Availability:** PC.

**F1A21D.** *Double precision version of F1A21.*

**F1A22.** Searches for a zero of a user-defined function between given input values until the width of the interval has collapsed to within a tolerance set by the user. User can specify if the routine is allowed to search outside the input interval in the case that there is no sign change in that interval. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F1b **Usage:** CALL F1A22 (*11 parameters*) **Precision:** Double **Availability:** PC.

**F2B.** Solves a system of N simultaneous nonlinear equations. Uses an iterative method which is a variation of Newton's method using Gaussian elimination in a manner similar to the Gauss-Seidel process. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F2 **Usage:** CALL F2B(*22 parameters*) **Precision:** Single (Double: F2BD) **Availability:** PC.

**F2BD.** *Double precision version of F2B.*

**F2BE.** Solves a system of N simultaneous nonlinear equations. (Easy to use.) Uses an iterative method which is a variation of Newton's method using Gaussian elimination in a manner similar to the Gauss-Seidel process. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F2 **Usage:** CALL F2BE(*11 parameters*) **Precision:** Single (Double: F2BED) **Availability:** PC.

**F2BED.** *Double precision version of F2BE.*

**F2BP.** Solves a system of N simultaneous nonlinear equations. Uses a modification of the Powell hybrid method. Jacobian can be provided by the user or approximated by the program. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F2 **Usage:** CALL F2BP(*27 parameters*) **Precision:** Single (Double: F2BPD) **Availability:** PC.

**F2BPD.** *Double precision version of F2BP.*

**F2BPE.** Solves a system of N simultaneous nonlinear equations. (Easy to use.) Uses a modification of the Powell hybrid method. Jacobian can be provided by the user or approximated by the program. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** F2 **Usage:** CALL F2BPE(*11 parameters*) **Precision:** Single (Double: F2BPED) **Availability:** PC.

**F2BPED.** *Double precision version of F2BPE.*

**FAC.** Factorial. Input is integer, output is real. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C1 **Usage:** I = FAC(N) **Precision:** Single (Double: DFAC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**FAC.** Factorial. Input is integer, output is real. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C1 **Usage:** R = FAC(N) **Precision:** Single (Double: DFAC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FACTOR.** Performs several types of common factor and component analysis for multivariate data, a correlation matrix, a covariance matrix, a factor pattern, or a matrix of scoring coefficients. A variety of methods are available for extracting factors, for prior communality estimation, and for rotation. Options: weights, factor scores. *Proprietary stand-alone program using SAS command language.* **Classes:** L13a **Precision:** Single **Availability:** CAMVAX.

**FACTOR.** Performs principal components analysis and factor analysis. Options: seven factor extraction techniques (principal axis factoring, least squares, maximum likelihood, etc.), rotation (varimax, equamax, quartimax, etc.), factor scores (regression, Bartlett, Anderson-Rubin), plot, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L13a, L13b **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**FACTR.** Extract initial factor loading estimates in factor analysis. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FACTR (*18 parameters*) **Precision:** Single (Double: DFACTR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FASTCLUS.** Performs a disjoint cluster analysis by minimizing the sum of squared distances from the cluster means. User specifies the maximum number of clusters and optionally, the minimum radius of the clusters. Designed for use with large data sets. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L14a1b **Precision:** Single **Availability:** CAMVAX.

**FC.** Fits piecewise polynomial to discrete data with equality and inequality constraints. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** K1a1a1, K1a2a, L8g **Usage:** CALL FC(*15 parameters*) **Also see:** CV **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**FCDF.** Computes the cumulative distribution function value for the F-distribution with degrees of freedom parameters NU1 and NU2. *Fortran subroutine in DATAPAC.* **Classes:** L5a1f **Usage:** CALL FCDF(X, NU1, NU2, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**FCOEF.** Compute a matrix of factor score coefficients for input to IMSL routine FSCOR. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FCOEF (*11 parameters*) **Precision:** Single (Double: DFCOEF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FCOST.** Discrete Fourier cosine transformation of an even sequence. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a3 **Usage:** CALL FCOST (N, SEQ, COEF) **Also see:** FCOSI **Precision:** Single (Double: DFCOST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FDF.** Evaluate the F distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1f **Usage:** R = FDF(F, DFN, DFD) **Precision:** Single (Double: DFDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FDGRD.** Approximate the gradient using forward differences. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4f **Usage:** CALL FDGRD (*7 parameters*) **Precision:** Single (Double: DFDGRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FDHES.** Approximate the Hessian using forward differences and function values. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4f **Usage:** CALL FDHES (*8 parameters*) **Precision:** Single (Double: DFDHES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FDJAC.** Approximate the Jacobian of M functions in N unknowns using forward differences. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4f **Usage:** CALL FDJAC (*9 parameters*) **Precision:** Single (Double: DFDJAC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FDOBL.** Compute a direct oblimin rotation of a factor loading matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FDOBL (*14 parameters*) **Precision:** Single (Double: DFDOBL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FEJER.** Compute the Fejer kernel. *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** L10a3a6 **Usage:** R = FEJER(M, RANGLE, EPS) **Precision:** Single (Double: DFEJER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFT.** FFT of complex data sequence (forward or inverse) – any number of points. Useful for multivariate transforms. Uses only real arithmetic. *Proprietary Fortran subroutine in PORT.* **Classes:** J1a2, J1b **Usage:** CALL FFT (A, B, NTOT, N, NSPAN, ISN) **Precision:** Single (Double: DFFT) **Availability:** 855NOS, 205.

**FFT1D.** One-dimensional fast Fourier transform of one or several real or complex vectors whose lengths can be expressed as $n = 2^p 3^q 5^r$, where p, q, and r are non-negative integers and $p \geq 2$. The user can choose among several options. *Fortran/meta subroutine in MAGEV.* **Classes:** J1a1, J1a2 **Usage:** CALL FFT1D(*10 parameters*) **Precision:** Single (Half: HFFT1D) **Availability:** 205 (vectorized)

**FFT2B.** Inverse Fourier transform of a complex periodic two-dimensional array. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1b **Usage:** CALL FFT2B (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFT2C.** Computes the fast Fourier transform of a complex valued sequence of length equal to a power of two. *Proprietary Fortran subroutine in IMSL.* **Classes:** J1a2 **Usage:** CALL FFT2C (A, M, IWK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFT2D.** Compute Fourier coefficients of a complex periodic two-dimensional array. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a2, J1b **Usage:** CALL FFT2D (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFT3D.** Fast Fourier transform of a complex valued 1, 2 or 3 dimensional array. *Proprietary Fortran subroutine in IMSL.* **Classes:** J1b **Usage:** CALL FFT3D (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTC.** Mixed radix fast Fourier transform of complex data. Two arrays used for complex data. *Proprietary Fortran subroutine in PORT.* **Classes:** J1a2 **Usage:** CALL FFTC (N, AR, AI) **Also see:** FFTCI **Precision:** Single (Double: DFFTC) **Availability:** 855NOS, 205.

**FFTCB.** Compute the complex periodic sequence from its Fourier coefficients. *Proprietary Fortran subroutine in*

*IMSL MATH/LIBRARY.* **Classes:** J1a2 **Usage:** CALL FFTCB (N, COEF, SEQ) **Also see:** FFTCI **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTCC.** Compute the fast Fourier transform of a complex valued sequence. *Proprietary Fortran subroutine in IMSL.* **Classes:** J1a2 **Usage:** CALL FFTCC (A, N, IWK, WK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTCF.** Compute the Fourier coefficients of a complex periodic sequence. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a2 **Usage:** CALL FFTCF (N, SEQ, COEF) **Also see:** FFTCI **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTCI.** Inverse fast Fourier transform, given the Fourier coefficients in the frequency domain. *Proprietary Fortran subroutine in PORT.* **Classes:** J1a2 **Usage:** CALL FFTCI (N, FR, FI) **Also see:** FFTC **Precision:** Single (Double: DFFTCI) **Availability:** 855NOS, 205.

**FFTLEN.** Compute the minimum extended series length for using the Singleton FFT; return the extended series length. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** J1a1 **Usage:** CALL FFTLEN (N, NDIV, NFFT) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**FFTR.** Mixed radix fast Fourier transform to find the transform of 2N real data points. *Proprietary Fortran subroutine in PORT.* **Classes:** J1a1 **Usage:** CALL FFTR (NNP2, A, B) **Also see:** FFTRI **Precision:** Single (Double: DFFTR) **Availability:** 855NOS, 205.

**FFTR.** Compute the Fourier coefficients of an input series of <real> observations; return the Fourier coefficients. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** J1a1 **Usage:** CALL FFTR (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**FFTRB.** Compute the real periodic sequence from its Fourier coefficients. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** J1a1 **Usage:** CALL FFTRB (N, COEF, SEQ) **Also see:** FFTRI **Precision:** Single (Double: DFFTRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTRC.** Fast Fourier transform of a real valued sequence. *Proprietary Fortran subroutine in IMSL.* **Classes:** J1a1 **Usage:** CALL FFTRC (A, N, X, IWK, WK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTRF.** Compute the Fourier coefficients of a real periodic sequence. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** J1a1 **Usage:** CALL FFTRF (N, SEQ, COEF) **Also see:** FFTRI **Precision:** Single (Double: DFFTRF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FFTRI.** Finds the inverse Fourier transform using Fourier coefficients assumed to arise from real data in the time domain. *Proprietary Fortran subroutine in PORT.* **Classes:** J1a1 **Usage:** CALL FFTRI (NN, FR, FI) **Also see:** FFTR **Precision:** Single (Double: DFFTRI) **Availability:** 855NOS, 205.

**FFTSC.** Sine and cosine transforms of a real valued sequence. *Proprietary Fortran subroutine in IMSL.* **Classes:** J1a3 **Usage:** CALL FFTSC (A, N, ST, CT, IWK, WK, CWK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FHARR.** Compute an oblique rotation of an unrotated factor loading matrix using the Harris-Kaiser method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FHARR (*16 parameters*) **Precision:** Single (Double: DFHARR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FIGI.** Transforms certain real non-symmetric tridiagonal matrix to symmetric tridiagonal matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1c **Usage:** CALL FIGI(NM, N, T, D, E, E2, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**FIGI2.** Transforms certain real non-symmetric tridiagonal matrix to symmetric tridiagonal matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1c **Usage:** CALL FIGI2(NM, N, T, D, E, A, IERR) **Also see:** IMTQL2 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**FIMAG.** Compute the image transformation matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FIMAG (NF, T, LDT, TI, LDTI) **Precision:** Single (Double: DFIMAG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FIN.** Evaluate the inverse of the F distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a2f **Usage:** R = FIN(P, DFN, DFD) **Precision:** Single (Double: DFIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FISH.** Clusters a sequence of cases into subsequences by Fisher's method of exact optimization. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a2 **Usage:** CALL FISH(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**FIT.** Fits a nonlinear model (including linear and polynomial models) to data by least squares. Output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication. Options: user-set starting values, number of iterations, accuracy, weighted and robust fits. *Command(s) in DATAPLOT interactive system.* **Classes:** L8a1a1, L8b1b1, L8c1b1, L8e1b1 **Usage:** Enter HELP FIT in Dataplot **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**FLINV.** Inverse Laplace transform of a user supplied complex function. *Proprietary Fortran subroutine in IMSL.* **Classes:** J3 **Usage:** CALL FLINV (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FLR.** Largest integer less than or equal to x. Input and output are real. *Proprietary Fortran function in PORT.* **Classes:** C1 **Usage:** R = FLR(X) **Precision:** Single (Double: DFLR) **Availability:** 855NOS, 205.

**FMIN.** Finds minimum of REAL FUNCTION F(X) on interval [AX,BX] to accuracy EPS. Needs EXTERNAL statement for user selected function name. Returns FMIN as location of minimum with AX,BX bracketing this value. *Portable Fortran software in NMS library.* **Classes:** G1a1a **Usage:** R = FMIN(AX, BX, F, EPS) **Precision:** Single (Double: DFMIN) **Availability:** PC.

**FMIN.** Finds an approximate local minimum of a univariate user defined function. *Proprietary Fortran function in PORT.* **Classes:** G1a2 **Usage:** R = FMIN(F, X, A, B, T) **Precision:** Single (Double: DFMIN) **Availability:** 855NOS, 205.

**FNLSQ.** Least squares approximation with user-supplied basis functions. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1a1a3 **Usage:** CALL FNLSQ (*10 parameters*) **Precision:** Single (Double: DFNLSQ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FNM.** Calculates an approximation to the point where a user defined function attains a minimum on a given interval. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** G1a1a **Precision:** Single **Availability:** 855NOS.

**FOPCS.** Compute an orthogonal Procrustes rotation of a factor loading matrix using a target matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FOPCS (*10 parameters*) **Precision:** Single (Double: DFOPCS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FOURIE.** Performs a Fourier analysis of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L10a3a2 **Usage:** CALL FOURIE(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**FPRMX.** Compute an oblique Promax or Procrustes rotation of a factor loading matrix using a target matrix, including pivot and power vector options. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FPRMX (*18 parameters*) **Precision:** Single (Double: DFPRMX) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FPS2H.** Solve Poisson's or Helmholtz's equation on a two-dimensional rectangle using a fast Poisson solver based on the HODIE finite-difference scheme. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I2b1a1a **Usage:** CALL FPS2H (*13 parameters*) **Precision:** Single (Double: DFPS2H) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FPS3H.** Solve Poisson's or Helmholtz's equation on a three-dimensional box using a fast Poisson solver based on the HODIE finite-difference scheme. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I2b1a1a **Usage:** CALL FPS3H (*17 parameters*) **Precision:** Single (Double: DFPS3H) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FQRUL.** Compute a Fejer quadrature rule with various classical weight functions. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2c **Usage:** CALL FQRUL (*8 parameters*) **Precision:** Single (Double: DFQRUL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FRACTAL PLOT.** Generate a two-dimensional self-similar fractal figure via the random iteration algorithm with optional control of plot features. The user specifies the initial rotation angle, the x and y axis scale factors and translations, the secondary rotation angle, and probability weights for each map. *Command(s) in DATAPLOT interactive system.* **Classes:** Q **Usage:** FRACTAL PLOT <LIST OF ONE TO SEVEN VARIABLES> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**FRAN.** Generates a random sample of size N from the F-distribution with degrees of freedom parameters NU1 and NU2. *Fortran subroutine in DATAPAC.* **Classes:** L6a6 **Usage:** CALL FRAN(N, NU1, NU2, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**FRDMN.** Perform Friedman's test for a randomized complete block design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7b **Usage:** CALL FRDMN (*8 parameters*) **Precision:** Single (Double: DFRDMN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FREQ.** Computes the sample frequency and sample cumulative frequency for the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1d **Usage:** CALL FREQ(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**FREQ.** Tally multivariate observations into a multi-way frequency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L2b **Usage:** CALL FREQ (*13 parameters*) **Precision:** Single (Double: DFREQ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FREQ.** Builds frequency or crosstabulation tables for one-way to n-way categorical data. Can compute tests and measures of association for two-way tables and can do stratified analysis and compute statistics within as well as across strata for n-way tables. Options: missing values, weights, additional analysis. *Proprietary stand-alone program using SAS command language.* **Classes:** L9 **Precision:** Single **Availability:** CAMVAX.

**FREQUENCIES.** Produces table of frequency counts and percentages for values of individual variables. Options:

bar charts, histograms, percentiles, univariate summary statistics, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1a1d, L1c1d, L3a1, L3a2, L3a4 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**FREQUENCY PLOT.** Generates one of the following plots: frequency, relative frequency, cumulative frequency, relative cumulative frequency. Optional control of upper and lower limits and/or class width. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a2 **Usage:** [RELATIVE/CUMULATIVE/RELATIVE CUMULATIVE] FREQUENCY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Also see:** S **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**FRESI.** Compute communalities and the standardized factor residual correlation matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FRESI (*9 parameters*) **Precision:** Single (Double: DFRESI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FROTA.** Compute an orthogonal rotation of a factor loading matrix using a generalized orthomax criterion, including quartimax, varimax, and equamax rotations. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FROTA (*12 parameters*) **Precision:** Single (Double: DFROTA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FRVAR.** Compute the factor structures and the variance explained by each factor. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FRVAR (*10 parameters*) **Precision:** Single (Double: DFRVAR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FSADDFV.** Broadcast add. The full-word scalar A is added to each array-element in the full-word vector B. The result is stored into the full-word vector C. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a11 **Usage:** CALL FSADDFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HSADDHV) **Availability:** 205 (vectorized)

**FSCOR.** Compute a set of factor scores given the factor score coefficient matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13a **Usage:** CALL FSCOR (*11 parameters*) **Precision:** Single (Double: DFSCOR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FSDIVFV.** Broadcast divide. The full-word scalar A(IA) is divided with each array-element in the full-word vector B and the result is stored into the full-word vector C. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a11 **Usage:** CALL FSDIVFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HSDIVHV) **Availability:** 205 (vectorized)

**FSINT.** Discrete Fourier cosine transformation of an odd sequence. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a3 **Usage:** CALL FSINT (N, SEQ, COEF) **Also see:** FSINI **Precision:** Single (Double: DFSINT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**FSMPYFV.** The full-word scalar A(IA) multiplies each array-element in the full-word vector B, a broadcast multiply. The result is stored into the full-word vector C. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a6 **Usage:** CALL FSMPYFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HSMPYHV) **Availability:** 205 (vectorized)

**FSSUBFV.** Broadcast subtract. Each array-element in the full-word vector B is subtracted from the full-word scalar A(IA). The result is stored into the full-word vector C. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a11 **Usage:** CALL FSSUBFV(N, A, IA, B, IB, C, IC, BIT)3 **Precision:** Single (Half: HSSUBHV) **Availability:** 205 (vectorized)

**FSTOFV.** The full-word scalar A(IA) is broadcast into the full-word target-vector starting with B(IB). *Fortran/meta subroutine in MAGEV.* **Classes:** D1a5 **Usage:** CALL FSTOFV(N, A, IA, B, IB, BIT) **Precision:** Single (Half: HSTOHV) **Availability:** 205 (vectorized)

**FTARPS.** Preliminary estimation of the autoregressive parameters in an ARIMA stochastic model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2d1 **Usage:** CALL FTARPS (*8 parameters*) **Also see:** FTAUTO **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTAUTO.** Mean, variance, autocovariances, autocorrelations, and partial autocorrelations for a stationary time series. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2a **Usage:** CALL FTAUTO (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTCAST.** Time series forecasts and probability limits using an ARIMA (Box-Jenkins) model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2d3 **Usage:** CALL FTCAST (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTCP.** Non-seasonal ARIMA (Box-Jenkins) stochastic model analysis for a single time series with full parameter iteration and maximum likelihood estimation. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2d2 **Usage:** CALL FTCP (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTCROS.** Means, variances, cross-covariances, and cross-correlations for two mutually stationary n channel time series. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10d **Usage:** CALL FTCROS (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTCRXY.** Cross-covariance of two mutually stationary time series. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10b2a **Usage:** CALL FTCRXY (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTFPS.** Fast Fourier transform estimates of power spectra and cross spectra of time series. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a3a3, L10b3a3 **Usage:** CALL FTFPS (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTFREQ.** Single or multichannel time series analysis in the time and frequency domains. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2, L10a3, L10c **Usage:** CALL FTFREQ (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTGEN.** Generation of a time series from a given ARIMA (Box-Jenkins) stochastic model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a20 **Usage:** CALL FTGEN (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTKALM.** Kalman filtering. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2e, L10c **Usage:** CALL FTKALM (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTMA.** Preliminary estimation of the moving average parameters in an ARIMA stochastic model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2d1 **Usage:** CALL FTMA (*8 parameters*) **Also see:** FTAUTO FTARPS **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTML.** Maximum likelihood estimation of autoregressive and moving average parameters in an ARIMA (Box-Jenkins) stochastic model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2d2 **Usage:** CALL FTML (*9 parameters*) **Also see:** FTARPS FTMA **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTRDIF.** Transformations, differences and seasonal differences of a time series for model identification. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2a, L10a1b, L10a1c1 **Usage:** CALL FTRDIF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTTR.** Parameter estimates for a univariate transfer function model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10b2b **Usage:** CALL FTTR (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTWEIN.** Wiener forecast for a stationary stochastic process. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10a2c **Usage:** CALL FTWEIN (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTWENM.** Multichannel Wiener forecast. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10d **Usage:** CALL FTWENM (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FTWENX.** Maximum likelihood parameter estimates for a multichannel, single output time series model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L10d **Usage:** CALL FTWENX (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**FVADDFV.** Vector addition, C=A+B. A, B and C are full-word vectors and addition is done elementwise. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a11 **Usage:** CALL FVADDFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HVADDHV) **Availability:** 205 (vectorized)

**FVDIVFV.** Vector division, C=A/B. A, B and C are full-word vectors and division is done elementwise. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a11 **Usage:** CALL FVDIVFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HVDIVHV) **Availability:** 205 (vectorized)

**FVMPYFV.** To perform vector multiplication. Corresponding elements in the full-word vectors, A and B, are multiplied and stored into the corresponding element in the full-word vector, C. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a6 **Usage:** CALL FVMPYFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HVMPYHV) **Availability:** 205 (vectorized)

**FVSUBFV.** Vector subtraction. Each element of the full-word vector B is subtracted from the corresponding element in the full-word vector A. The result is stored in the corresponding element of the full-word vector C. *Fortran/meta subroutine in MAGEV.* **Classes:** D1a11 **Usage:** CALL FVSUBFV(N, A, IA, B, IB, C, IC, BIT) **Precision:** Single (Half: HVSUBHV) **Availability:** 205 (vectorized)

**FVTOFV.** The full-word vector starting with A(IA) is copied into the full-word target-vector starting with B(IB). *Fortran/meta subroutine in MAGEV.* **Classes:** D1a5 **Usage:** CALL FVTOFV(N, A, IA, B, IB, BIT) **Precision:** Single (Half: HVTOHV) **Availability:** 205 (vectorized)

**FZERO.** Finds zero of REAL FUNCTION F(X) on interval [B,C] to relative and absolute accuracy RE, AE. Requires EXTERNAL statement for user selected function name. R is input estimate of location of zero, if this is known. Best output estimate of zero returned in B. *Portable Fortran software in NMS library.* **Classes:** F1b **Usage:** CALL FZERO(F, B, C, R, RE, AE, IFLAG) **Precision:** Single (Double: DFZERO) **Availability:** PC.

# G

**G01AAF.** Calculates the mean, standard deviation, coefficients of skewness and kurtosis, and the maximum and minimum values for a set of ungrouped data. Weighting may be used. *Proprietary Fortran subroutine in NAG.* **Classes:** L1a1 **Usage:** CALL G01AAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01ABF.** Computes the means, standard deviations, corrected sums of squares and products, maximum and minimum values, and the product-moment correlation coefficient for two variables. Unequal weighting may be given. *Proprietary Fortran subroutine in NAG.* **Classes:** L1b **Usage:** CALL G01ABF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01ADF.** Calculates the mean, standard deviation and coefficients of skewness and kurtosis for data grouped on a frequency distribution. *Proprietary Fortran subroutine in NAG.* **Classes:** L1a3 **Usage:** CALL G01ADF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01AEF.** Constructs a frequency distribution of a variable, according to either user-supplied, or routine-calculated class boundary values. *Proprietary Fortran subroutine in NAG.* **Classes:** L2b **Usage:** CALL G01AEF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01AFF.** Performs the analysis of a two-way rc contingency table or classification. If r=c=2, and the total number of objects classified is 40 or fewer, then the probabilities for Fisher's exact test are computed. Otherwise, a test statistic is computed (with Yates' correction when r=c=2), which under the assumption of no association between the classifications has approximately a chi-square distribution with (r-1)(c-1) degrees of freedom. *Proprietary Fortran subroutine in NAG.* **Classes:** L9b **Usage:** CALL G01AFF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01AGF.** Performs a scatter plot of two variables on a character printing device, with a chosen number of character positions in each direction. *Proprietary Fortran subroutine in NAG.* **Classes:** L3b3a **Usage:** CALL G01AGF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01AHF.** Performs a scatter plot of a vector against the Normal scores for a sample of the same size, on a character printing device, with a chosen number of character positions in each direction. *Proprietary Fortran subroutine in NAG.* **Classes:** L4a1a2n **Usage:** CALL G01AHF (*12 parameters*) **Also see:** G01DAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01AJF.** Prints a histogram on a character printing device, with user control over size, positioning, and the range of data values included. *Proprietary Fortran subroutine in NAG.* **Classes:** L3a1 **Usage:** CALL G01AJF(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01BAF.** Returns the probability associated with the lower tail of the Student's t distribution with n degrees of freedom, through the routine name. *Proprietary Fortran function in NAG.* **Classes:** L5a1t **Usage:** R = G01BAF(IDF, T, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01BBF.** Returns the probability associated with the upper tail of the F or variance-ratio distribution with m and n degrees of freedom, through the routine name. *Proprietary Fortran function in NAG.* **Classes:** L5a1f **Usage:** R = G01BBF(I1, I2, A, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01BCF.** Returns the probability, P(x,n), associated with the upper tail of the chi-square distribution with n degrees of freedom, through the function name. *Proprietary Fortran function in NAG.* **Classes:** L5a1c **Usage:** R = G01BCF(X, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01BDF.** Returns the probability, I(x)(a,b) associated with the lower tail of the Beta distribution of the first kind with parameters a and b, through the routine name. *Proprietary Fortran function in NAG.* **Classes:** L5a1b **Usage:** R = G01BDF(X, A, B, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01CAF.** Returns the deviate, x(p) associated with the given lower tail probability p of Student's t distribution with n degrees of freedom, through the function name. *Proprietary Fortran function in NAG.* **Classes:** L5a2t **Usage:** R = G01CAF(P, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01CBF.** Returns the deviate, x(p) associated with the lower tail probability p of the F variance-ratio distribution with m and n degrees of freedom, through the function name. *Proprietary Fortran function in NAG.* **Classes:** L5a2f **Usage:** R = G01CBF(P, M, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01CCF.** Returns the deviate, x(p), associated with the given lower tail probability p of the chi-square distribution with n degrees of freedom, through the function name. *Proprietary Fortran function in NAG.* **Classes:** L5a2c **Usage:** R = G01CCF(P, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01CDF.** Returns the deviate, x(p), associated with the given lower tail probability p of the Beta distribution of the first kind with parameters a and b, through the function name. *Proprietary Fortran function in NAG.* **Classes:** L5a2b **Usage:** R = G01CDF(P, A, B, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01CEF.** Returns the deviate, x(p), associated with the given lower tail probability p of the standardized Normal distribution, through the function name. *Proprietary Fortran function in NAG.* **Classes:** L5a2n **Usage:** R =

G01CEF(P, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01DAF.** Computes a set of Normal scores, i.e. the expected values of an ordered set of independent observations from a Normal distribution with mean 0.0 and standard deviation 1.0. *Proprietary Fortran subroutine in NAG.* **Classes:** L5a2n **Usage:** CALL G01DAF (*7 parameters*) **Also see:** G01AHF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01DBF.** Calculates an approximation to the set of Normal Scores, i.e. the expected values of an ordered set of independent observations from a Normal distribution with mean 0.0 and standard deviation 1.0. *Proprietary Fortran subroutine in NAG.* **Classes:** L5a2n **Usage:** CALL G01DBF (N, PP, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01DCF.** Computes an approximation to the variance-covariance matrix of an ordered set of independent observations from a Normal distribution with mean 0.0 and standard deviation 1.0. *Proprietary Fortran subroutine in NAG.* **Classes:** L5b2n **Usage:** CALL G01DCF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G01DDF.** Calculates Shapiro and Wilk's W statistic and its significance level for testing Normality. *Proprietary Fortran subroutine in NAG.* **Classes:** L4a1a4n **Usage:** CALL G01DDF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BAF.** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in the array X. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c1 **Usage:** CALL G02BAF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BBF.** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BBF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BCF.** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BCF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BDF.** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in the array X. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c1 **Usage:** CALL G02BDF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BEF.** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BEF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BFF.** Computes means and standard deviations of variables, sums of squares and cross-products about zero and correlation-like coefficients for a set of data in the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BFF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BGF.** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in specified columns of the array X. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c1 **Usage:** CALL G02BGF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BHF.** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in specified columns of the array X, omitting completely any cases with a missing observation for any variable (either over all variables in the data array or over only those variables in the selected subset). *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BHF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BJF.** Computes means and standard deviations of variables, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients for a set of data in specified columns of the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BJF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BKF.** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in specified columns of the array X. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c1 **Usage:** CALL G02BKF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BLF.** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in specified columns of the array X, omitting completely any cases with a missing observation for any variable (either over all variables in the data array or over only those variables in the selected subset). *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BLF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BMF.** Computes means and standard deviations of variables, sums of squares and cross-products about zero, and correlation-like coefficients for a set of data in specified columns of the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BMF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BNF.** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X; the data array X is overwritten, and on exit it contains the ranks of the observations. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c1b **Usage:** CALL G02BNF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BPF.** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable; the data array X is overwritten, and on exit contains the ranks of the observations. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BPF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BQF.** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X; the data array X is preserved, and the ranks of the observations are not available on exit from the routine. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c1b **Usage:** CALL G02BQF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BRF.** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X, omitting completely any cases with a missing observation for any variable; the data array X is preserved, and the ranks of the observations are not available on exit from the routine. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BRF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02BSF.** Computes Kendall and/or Spearman non-parametric rank correlation coefficients for a set of data in the array X, omitting cases with missing values from only those calculations involving the variables for which the values are missing; the data array X is preserved, and the ranks of the observations are not available on exit from the routine. *Proprietary Fortran subroutine in NAG.* **Classes:** L1c2 **Usage:** CALL G02BSF (*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CAF.** Performs a simple linear regression with dependent variable Y and independent variable X. *Proprietary Fortran subroutine in NAG.* **Classes:** L8a1a1 **Usage:** CALL G02CAF (N, X, Y, RESULT, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CBF.** Performs a simple linear regression with no constant with dependent variable Y and independent variable X. *Proprietary Fortran subroutine in NAG.* **Classes:** L8a1a1 **Usage:** CALL G02CBF (N, X, Y, RESULT, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CCF.** Performs a simple linear regression with dependent variable Y and independent variable X, omitting cases involving missing values. *Proprietary Fortran subroutine in NAG.* **Classes:** L8a1a1 **Usage:** CALL G02CCF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CDF.** Performs a simple linear regression with no constant, with dependent variable Y and independent variable X, omitting cases involving missing values. *Proprietary Fortran subroutine in NAG.* **Classes:** L8a1a1 **Usage:** CALL G02CDF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CEF.** Takes selected elements from two vectors (typically vectors of means and standard deviations) to form two smaller vectors, and selected rows and columns from two matrices (typically either matrices of sums of squares and cross-products of deviations from means and Pearson product-moment correlation coefficients, or matrices of sums of squares and cross-products about zero and correlation-like coefficients) to form two smaller matrices, allowing re-ordering of elements in the process. *Proprietary Fortran subroutine in NAG.* **Classes:** L2c, L8i **Usage:** CALL G02CEF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CFF.** Re-orders the elements in two vectors (typically vectors of means and standard deviations), and the rows and columns in two matrices (typically either matrices of sums of squares and cross-products of deviations from means and Pearson product-moment correlation coefficients, or matrices of sums of squares and cross-products about zero and correlation-like coefficients). *Proprietary Fortran subroutine in NAG.* **Classes:** L8i **Usage:** CALL G02CFF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CGF.** Performs a multiple linear regression on a set of variables whose means, sums of squares and cross-products of deviations from means, and Pearson product-moment correlation coefficients are given. *Proprietary Fortran subroutine in NAG.* **Classes:** L8c1b2 **Usage:** CALL G02CGF (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CHF.** Performs a multiple linear regression with no constant on a set of variables whose sums of squares and cross-products about zero and correlation-like coefficients are given. *Proprietary Fortran subroutine in NAG.* **Classes:** L8c1b2 **Usage:** CALL G02CHF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G02CJF.** Performs one or more multiple linear regressions, regressing each of a set of dependent variables separately on the same set of independent variables. Input to the routine is in the form of raw data. Output includes, for each dependent variable, estimates of the regression coefficients, and an estimate of the variance of residuals. *Proprietary Fortran subroutine in NAG.* **Classes:** L8c2 **Usage:** CALL G02CJF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G04ADF.** Performs the analysis of variance for a Latin square design. *Proprietary Fortran subroutine in NAG.* **Classes:** L7c **Usage:** CALL G04ADF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G04AEF.** Performs an analysis of variance for a one-way classification with treatment groups of possibly unequal size, and also computes the treatment group means. A fixed effects model is assumed. *Proprietary Fortran subroutine in NAG.* **Classes:** L7a1 **Usage:** CALL G04AEF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G04AFF.** Performs an analysis of variance for a two-way cross-classification with equal cell frequencies. A fixed effects model is assumed. *Proprietary Fortran subroutine in NAG.* **Classes:** L7b **Usage:** CALL G04AFF (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G04AGF.** Performs an analysis of variance for a two-way hierarchical classification with subgroups of possibly unequal size, and also computes the treatment group and subgroup means. *Proprietary Fortran subroutine in NAG.* **Classes:** L7b **Usage:** CALL G04AGF (Y, N, K, LSUB, NOBS, L, NGP, GBAR, SGBAR, GM, SS, IDF, F, FP, IFAIL **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05CAF.** Returns a pseudo-random number taken from a uniform distribution between 0 and 1. *Proprietary Fortran function in NAG.* **Classes:** L6a21 **Usage:** R = G05CAF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05CBF.** Sets the basic generator routine G05CAF to a repeatable initial state. *Proprietary Fortran subroutine in NAG.* **Classes:** L6c **Usage:** CALL G05CBF (I) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05CCF.** Sets the basic generator routine G05CAF to a non-repeatable initial state. *Proprietary Fortran subroutine in NAG.* **Classes:** L6c **Usage:** CALL G05CCF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05CFF.** Save state of random number generating routines. *Proprietary Fortran subroutine in NAG.* **Classes:** L6c **Usage:** CALL G05CFF (IA, NI, XA, NX, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05CGF.** Restores the state of the basic generator routine G05CAF after a previous call to G05CFF. *Proprietary Fortran subroutine in NAG.* **Classes:** L6c **Usage:** CALL G05CGF (IA, NI, XA, NX, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DAF.** Returns a pseudo-random real number taken from a uniform distribution between A and B. *Proprietary Fortran function in NAG.* **Classes:** L6a21 **Usage:** R = G05DAF(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DBF.** Returns a pseudo-random real number taken from a (negative) exponential distribution with mean A. *Proprietary Fortran function in NAG.* **Classes:** L6a5 **Usage:** R = G05DBF(A) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DCF.** Returns a pseudo-random real number taken from a logistic distribution with mean A and spread B. *Proprietary Fortran function in NAG.* **Classes:** L6a12 **Usage:** R = G05DCF(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DDF.** Returns a pseudo-random real number taken from a normal (Gaussian) distribution with mean A and standard deviation B. *Proprietary Fortran function in NAG.* **Classes:** L6a14 **Usage:** R = G05DDF(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DEF.** Returns a pseudo-random real number taken from a lognormal distribution with parameters A and B. *Proprietary Fortran function in NAG.* **Classes:** L6a12 **Usage:** R = G05DEF(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DFF.** Returns a pseudo-random real number taken from a Cauchy distribution with median A and semi-interquartile range B. *Proprietary Fortran function in NAG.* **Classes:** L6a3 **Usage:** R = G05DFF(A, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DGF.** Returns a pseudo-random real number taken from a gamma distribution with parameters G and H. *Proprietary Fortran function in NAG.* **Classes:** L6a7 **Usage:** R = G05DGF(G, H, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DHF.** Returns a pseudo-random real number taken from a chi-square distribution with N degrees of freedom. *Proprietary Fortran function in NAG.* **Classes:** L6a3 **Usage:** R = G05DHF(N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DJF.** Returns a pseudo-random real number taken from a Student's t-distribution with N degrees of freedom. *Proprietary Fortran function in NAG.* **Classes:** L6a20 **Usage:** R = G05DJF(N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DKF.** Returns a pseudo-random real number taken from Snedecor's F (or Fisher's variance ratio) distribution with M and N degrees of freedom. *Proprietary Fortran function in NAG.* **Classes:** L6a6 **Usage:** R = G05DKF(M, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DLF.** Returns a pseudo-random real number taken from a beta distribution of the first kind with parameters G and H. *Proprietary Fortran function in NAG.* **Classes:** L6a2 **Usage:** R = G05DLF(G, H, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DMF.** Returns a pseudo-random real number taken from a beta distribution of the second kind with parameters G and H. *Proprietary Fortran function in NAG.* **Classes:** L6a2 **Usage:** R = G05DMF(G, H, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DPF.** Returns a pseudo-random number taken from a two parameter Weibull distribution with shape parameter A and scale parameter B. *Proprietary Fortran function in NAG.* **Classes:** L6a23 **Usage:** R = G05DPF(A, B, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DYF.** Returns a pseudo-random integer taken from a uniform distribution between M and N (inclusive). *Proprietary Fortran function in NAG.* **Classes:** L6a21 **Usage:** I = G05DYF(M, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05DZF.** Returns a pseudo-random logical value – TRUE with probability P and FALSE with probability (1-P). *Proprietary Fortran function in NAG.* **Classes:** L6a2 **Usage:** L = G05DZF(P) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EBF.** Sets up the reference vector R for a discrete uniform distribution between M and N inclusive. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a21 **Usage:** CALL G05EBF (M, N, R, NR, IFAIL) **Also see:** G05EYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05ECF.** Sets up the reference vector R for a Poisson distribution with mean T. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a16 **Usage:** CALL G05ECF (T, R, NR, IFAIL) **Also see:** G05EYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EDF.** Sets up the reference vector R for a binomial distribution of the number of successes in N trials, each with probability of success P. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a2 **Usage:** CALL G05EDF (N, P, R, NR, IFAIL) **Also see:** G05EYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EEF.** Sets up the reference vector R for a negative binomial distribution of the number of successes before N failures, where each trial has probability of success P. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a14 **Usage:** CALL G05EEF (N, P, R, NR, IFAIL) **Also see:** G05EYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EFF.** Sets up a reference vector R for a hypergeometric distribution of the number of specified items in a sample of size L, taken from a population of size N with M specified items in it. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a8 **Usage:** CALL G05EFF (L, M, N, R, NR, IFAIL) **Also see:** G05EYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EGF.** Sets up a reference vector for an autoregressive moving-average (ARMA) time series model with Normally distributed errors, so that G05EWF may be used to generate successive terms. It also initializes the series to a stationary position. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a20 **Usage:** CALL G05EGF (*9 parameters*) **Also see:** G05EWF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EHF.** Performs a pseudo-random permutation of a vector of integers. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a16 **Usage:** CALL G05EHF(INDEX, N, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EJF.** Selects a pseudo-random sample from an integer vector. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a19 **Usage:** CALL G05EJF(IA, N, IB, M, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EWF.** Generates the next term from an autoregressive moving-average time series using a reference vector set up by G05EGF. *Proprietary Fortran function in NAG.* **Classes:** L6a20 **Usage:** R = G05EWF(R, NR, IFAIL) **Also see:** G05EGF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EXF.** Sets up the reference vector R for a discrete distribution with PDF (probability density function) or CDF (cumulative distribution function) P. *Proprietary Fortran subroutine in NAG.* **Classes:** L6a7 **Usage:** CALL G05EXF (*7 parameters*) **Also see:** G05EYF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EYF.** Returns a pseudo-random integer taken from a discrete distribution defined by a reference vector R. *Proprietary Fortran function in NAG.* **Classes:** L6a21 **Usage:** I = G05EYF(R, NR) **Also see:** G05EBF G05ECF G05EDF G05EEF G05EFF G05EXF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G05EZF.** Returns a pseudo-random multivariate Normal vector taken from a distribution described by a reference vector set up by G05EAF. *Proprietary Fortran subroutine in NAG.* **Classes:** L6b14 **Usage:** CALL G05EZF(Z, N, R, NR, IFAIL) **Also see:** G05EAF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08AAF.** Performs the Sign test on two related samples of size N. *Proprietary Fortran subroutine in NAG.* **Classes:** L4b3 **Usage:** CALL G08AAF (X, Y, N, IS, N1, P, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08ABF.** Performs the Wilcoxon matched pairs signed ranks test on two related samples of size N. *Proprietary Fortran subroutine in NAG.* **Classes:** L4b3 **Usage:** CALL G08ABF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08ACF.** Performs the Median test on two independent samples of possibly unequal size. *Proprietary Fortran subroutine in NAG.* **Classes:** L4b1b **Usage:** CALL G08ACF (*8 parameters*) **Precision:** Single **Availability:**

855NOS, 855VE, 205.

**G08ADF.** Performs the Mann-Whitney U test on two independent samples of possibly unequal size. *Proprietary Fortran subroutine in NAG.* **Classes:** L4b1b **Usage:** CALL G08ADF (X, N, N1, W, U, P, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08AEF.** Performs the Friedman two-way analysis of variance by ranks on K related samples of size N. *Proprietary Fortran subroutine in NAG.* **Classes:** L7b **Usage:** CALL G08AEF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08AFF.** Performs the Kruskal-Wallis one-way analysis of variance by ranks on k independent samples of possibly unequal sizes. *Proprietary Fortran subroutine in NAG.* **Classes:** L7a2 **Usage:** CALL G08AFF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08BAF.** Performs Mood's and David's tests for dispersion differences between two independent samples of possibly unequal size. *Proprietary Fortran subroutine in NAG.* **Classes:** L4b1b **Usage:** CALL G08BAF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08CAF.** Performs the one sample Kolmogorov-Smirnov distribution test. *Proprietary Fortran subroutine in NAG.* **Classes:** L4a1c **Usage:** CALL G08CAF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08DAF.** Calculates Kendall's coefficient of concordance on k independent rankings of n objects or individuals. *Proprietary Fortran subroutine in NAG.* **Classes:** L4c1b **Usage:** CALL G08DAF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08RAF.** Calculates the parameter estimates, score statistics and their variance-covariance matrices for the linear model using a likelihood based on the ranks of the observations. *Proprietary Fortran subroutine in NAG.* **Classes:** L8c6 **Usage:** CALL G08RAF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G08RBF.** Calculates the parameter estimates, score statistics and their variance-covariance matrices for the linear model using a likelihood based on the ranks of the observations when some of the observations may be right censored. *Proprietary Fortran subroutine in NAG.* **Classes:** L8c6 **Usage:** CALL G08RBF (*22 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G11SAF.** Fits a latent variable model (with a single factor) to data consisting of a set of measurements on individuals in the form of binary-valued sequences (generally referred to as score patterns). Various measures of goodness of fit are calculated along with the factor (theta) scores. *Proprietary Fortran subroutine in NAG.* **Classes:** L13a **Usage:** CALL G11SAF (*34 parameters*) **Also see:** G11SBF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G11SBF.** Is a service routine which may be used prior to calling NAG Library routine G11SAF to calculate the frequency distribution of a set of dichotomous score patterns. *Proprietary Fortran subroutine in NAG.* **Usage:** CALL G11SBF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13AAF.** Carries out non-seasonal and seasonal differencing on a time series. Information which allows the original series to be reconstituted from the differenced series is also produced. This information is required in time series forecasting. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a1c1 **Usage:** CALL G13AAF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13ABF.** Computes the sample autocorrelation function of a time series. It also computes the sample mean, the sample variance and a statistic which may be used to test the hypothesis that the true autocorrelation function is zero. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2a1 **Usage:** CALL G13ABF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13ACF.** Calculates partial autocorrelation coefficients given a set of autocorrelation coefficients. It also calculates the predictor error variance ratios for increasing order of finite lag autoregressive predictor, and the autoregressive parameters associated with the predictor of maximum order. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2a2, L10a2c1 **Usage:** CALL G13ACF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13ADF.** Calculates preliminary estimates of the parameters of an autoregressive integrated moving average (ARIMA) model from the autocorrelation function of the appropriately differenced time series. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2d1 **Usage:** CALL G13ADF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13AEF.** Fits a seasonal autoregressive-integrated moving average (ARIMA) model to an observed time series, using a non-linear least squares procedure incorporating backforecasting. Parameter estimates are obtained, together with appropriate standard errors. The residual series is returned, and information for use in forecasting the time series is produced for use by the routines G13AGF and G13AHF. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2d2 **Usage:** CALL G13AEF (*32 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13AFF.** Is an easy-to-use version of G13AEF. It fits a seasonal autoregressive integrated moving average (ARIMA) model to an observed time series, using a non-linear least squares procedure incorporating backforecasting. Parameter estimates are obtained, together with appropriate standard errors. The residual series is returned, and information for use in forecasting the time series is produced for use in the routines G13AGF and G13AHF. *Propri-*

*etary Fortran subroutine in NAG.* **Classes:** L10a2d2 **Usage:** CALL G13AFF (*23 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13AGF.** Accepts a series of new observations of a time series, the model of which is already fully specified, and updates the state set information for use in constructing further forecasts. The previous specifications of the time series model should have been obtained by using G13AEF or G13AFF to estimate the relevant parameters. The supplied state set will originally have been produced by G13AEF or G13AFF, but may since have been updated by earlier calls to G13AGF. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2d3 **Usage:** CALL G13AGF (*12 parameters*) **Also see:** G13AEF G13AFF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13AHF.** Produces forecasts of a time series, given a time series model which has already been fitted to the time series using routine G13AEF or G13AFF. The original observations are not required, since G13AHF uses as input either the original state set produced by G13AEF or G13AFF or the state set updated by a series of new observations using G13AGF. Standard errors of the forecasts are also provided. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2d3 **Usage:** CALL G13AHF (*13 parameters*) **Also see:** G13AEF G13AFF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13AJF.** Applies a fully specified seasonal ARIMA model to an observed time series, generates the state set for forecasting and (optionally) derives a specified number of forecasts together with their standard deviations. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2d3 **Usage:** CALL G13AJF(*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BAF.** Filters a time series by an ARIMA model. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a1c4 **Usage:** CALL G13BAF(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BBF.** Filters a time series by a transfer function model. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a1c4 **Usage:** CALL G13BBF (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BCF.** Calculates cross correlations between two time series. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b2a **Usage:** CALL G13BCF(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BDF.** Calculates preliminary estimates of the parameters of a transfer function model. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b2b **Usage:** CALL G13BDF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BEF.** Estimates the parameters in a multi-input model relating one output series to one or more input series, using a choice of non-linear least squares, exact likelihood, or marginal likelihood for estimation criteria. If no input series are present, estimates a univariate ARIMA model. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a2d2, L10d **Usage:** CALL G13BEF (*31 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BGF.** Accepts a series of new observations of an output time series and any associated input time series, for which a multi-input model is already fully specified, and updates the state set information for use in constructing further forecasts. *Proprietary Fortran subroutine in NAG.* **Classes:** L10d **Usage:** CALL G13BGF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BHF.** Produces forecasts of a time series (the output series) which depends on one or more other (input) series via a multi-input model which will usually have been fitted using G13BEF. *Proprietary Fortran subroutine in NAG.* **Classes:** L10d **Usage:** CALL G13BHF (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13BJF.** Produces forecasts of a time series (the output series) which depends on one or more other (input) series via a previously estimated multi-input model for which the state set information is not available. *Proprietary Fortran subroutine in NAG.* **Classes:** L10d **Usage:** CALL G13BJF (*25 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CAF.** Calculates the smoothed sample spectrum of a univariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a3a4 **Usage:** CALL G13CAF(*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CBF.** Calculates the smoothed sample spectrum of a univariate time series using spectral smoothing by the trapezium frequency (Daniell) window. *Proprietary Fortran subroutine in NAG.* **Classes:** L10a3a3 **Usage:** CALL G13CBF(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CCF.** Calculates the smoothed sample cross spectrum of a bivariate time series using one of four lag windows – rectangular, Bartlett, Tukey or Parzen window. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b3a4 **Usage:** CALL G13CCF(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CDF.** Calculates the smoothed sample cross spectrum of a bivariate time series using spectral smoothing by the trapezium frequency (Daniell) window. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b3a3 **Usage:** CALL G13CDF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CEF.** Calculates the cross amplitude spectrum and squared coherency, together with lower and upper bounds from the univariate and bivariate (cross) spectra. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b3a6 **Usage:** CALL G13CEF(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CFF.** Calculates the gain and phase together with lower and upper bounds from the univariate and bivariate spectra. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b3a6 **Usage:** CALL G13CFF(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13CGF.** Calculates the noise spectrum together with multiplying factors for the bounds and the impulse response function and its standard error, from the univariate and bivariate spectra. *Proprietary Fortran subroutine in NAG.* **Classes:** L10b3a6 **Usage:** CALL G13CGF(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13DAF.** Calculates the cross covariance or cross correlation function of a multivariate time series. *Proprietary Fortran subroutine in NAG.* **Classes:** L10c **Usage:** CALL G13DAF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13DBF.** Calculates the multivariate partial autocorrelation function of a multivariate time series. *Proprietary Fortran subroutine in NAG.* **Classes:** L10c **Usage:** CALL G13DBF (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G13DCF.** Fits a vector autoregressive moving average (VARMA) model to an observed vector of time series using the method of maximum likelihood. Standard errors of parameter estimates are computed along with their appropriate correlation matrix. The routine also calculates estimates of the residual series. *Proprietary Fortran subroutine in NAG.* **Classes:** L10c **Usage:** CALL G13DCF (*27 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**G1B2E.** Minimizes the sum of squares of M nonlinear functions in N variables using a modification of the Levenberg-Marquardt algorithm from Minpack. The Jacobian can be approximated by G1B2E or supplied by the user. (Easy to use.) User supplied Jacobian can be checked by G4C, and covariance matrix can be obtained by G4F. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** K1b1a1, K1b1a2 **Usage:** CALL G1B2E(*13 parameters*) **Precision:** Single **Availability:** PC.

**G1B2U.** Minimizes the sum of squares of M nonlinear functions in N variables using a modification of the Levenberg-Marquardt algorithm from Minpack. The Jacobian can be supplied by the user or approximated by G1B2U. User supplied Jacobian can be checked by G4C, covariance matrix can be computed by G4F. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** K1b1a1, K1b1a2 **Usage:** CALL G1B2U(*26 parameters*) **Precision:** Single **Availability:** PC.

**G2A1.** Solves the linear programming problem: minimize (Transpose of costs) × X, subject to the constraint equations AX+W=R and bounds on the components of X and W. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** G2a1 **Usage:** CALL G2A1(*17 parameters*) **Precision:** Single **Availability:** PC.

**G4C.** Checks the gradients of M nonlinear functions in N variables evaluated at a point X, for consistency with the functions themselves. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** G4c **Usage:** CALL G4C(*10 parameters*) **Also see:** G1B2U G1B2E **Precision:** Single **Availability:** PC.

**G4F.** Calculates the covariance matrix for a nonlinear data fitting problem. Intended to be used after a successful return from either G1B2U or G1B2E. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** K6d **Usage:** CALL G4F(*13 parameters*) **Also see:** G1B2U G1B2E **Precision:** Single **Availability:** PC.

**GAMCDF.** Computes the cumulative distribution function value for the gamma distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1g **Usage:** CALL GAMCDF(X, GAMMA, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GAMDF.** Gamma distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1g, C7e **Usage:** R = GAMDF(X, A) **Precision:** Single (Double: DGAMDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMI.** Incomplete gamma function. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7e **Usage:** R = GAMI(A, X) **Precision:** Single (Double: DGAMI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GAMI.** Incomplete gamma function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7e **Usage:** R = GAMI(A, X) **Precision:** Single (Double: DGAMI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMIC.** Complementary incomplete gamma function. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7e **Usage:** R = GAMIC(A, X) **Precision:** Single (Double: DGAMIC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GAMIC.** Complementary incomplete gamma function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7e **Usage:** R = GAMIC(A, X) **Precision:** Single (Double: DGAMIC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMIT.** Tricomi's incomplete gamma function. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7e **Usage:** R = GAMIT(A, X) **Precision:** Single (Double: DGAMIT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GAMIT.** Tricomi's incomplete gamma function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7e **Usage:** R = GAMIT(A, X) **Precision:** Single (Double: DGAMIT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMLN.** $\ln(\Gamma(x))$, for non-negative x. *Portable Fortran function in CMLIB (AMOSLIB sublibrary).* **Classes:** C7a **Usage:** R = GAMLN(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GAMMA.** $\Gamma(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** R = GAMMA(X)

**Also see:** GAMLIM **Precision:** Single (Double: DGAMMA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GAMMA.** $\Gamma(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C7a **Usage:** R = GAMMA(Y) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMMA.** $\Gamma(x)$. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** R = GAMMA(X) **Precision:** Single (Double: DGAMMA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMMA.** $\Gamma(x)$. *Portable Fortran software in NMS library.* **Classes:** C7a **Usage:** R = GAMMA(X) **Precision:** Single (Double: DGAMMA) **Availability:** PC.

**GAMMA PPCC PLOT.** Generates a probability plot correlation coefficient plot for the gamma distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging form 1 to 100 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3g **Usage:** GAMMA PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**GAMMA PROBABILITY PLOT.** Generates a gamma probability plot with tail length parameter $\gamma$, mean $\gamma$, and standard deviation $\sqrt{\gamma}$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2g **Usage:** GAMMA PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**GAMPLT.** Generates a gamma probability plot with tail length parameter GAMMA, mean GAMMA, and standard deviation $\sqrt{\text{GAMMA}}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2g **Usage:** CALL GAMPLT(X, N, GAMMA) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GAMPPF.** Computes the percent point function value for the gamma distribution with mean GAMMA and standard deviation $\sqrt{\text{GAMMA}}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a2g **Usage:** CALL GAMPPF(P, GAMMA, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GAMR.** $1/\Gamma(x)$. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** R = GAMR(X) **Precision:** Single (Double: DGAMR) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GAMR.** $1/\Gamma(x)$. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** R = GAMR(X) **Precision:** Single (Double: DGAMR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GAMRAN.** Generates a random sample of size N from the gamma distribution with tail length parameter GAMMA, mean GAMMA and standard deviation $\sqrt{\text{GAMMA}}$. *Fortran subroutine in DATAPAC.* **Classes:** L6a7 **Usage:** CALL GAMRAN(N, GAMMA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GAUSQ.** Finds the abscissae and weights for Gauss quadrature on the interval (a,b) for a general weight function with known moments. *Proprietary Fortran subroutine in PORT.* **Classes:** H2c **Usage:** CALL GAUSQ (N, A, B, C, NU, X, W) **Precision:** Single (Double: DGAUSQ) **Availability:** 855NOS, 205.

**GCDF.** Evaluate a general continuous cumulative distribution function given ordinates of the density. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1g **Usage:** R = GCDF(X0, IOPT, M, X, F) **Precision:** Single (Double: DGCDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GCIN.** Evaluate the inverse of a general continuous cumulative distribution function given ordinates of the density. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a2g **Usage:** R = GCIN(P, IOPT, M, X, F) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GCSCP.** Generate centered variables, squares, and crossproducts. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L2a **Usage:** CALL GCSCP (*12 parameters*) **Precision:** Single (Double: DGCSCP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GDATA.** Retrieves one of several data sets frequently cited in the statistical literature, including the Longley data, Fisher's iris data, and data from books by Anderson, Draper and Smith, Box and Jenkins, and Robinson. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L17 **Usage:** CALL GDATA (*7 parameters*) **Precision:** Single (Double: DGDATA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GDHES.** Approximate the Hessian using forward differences and a user-supplied gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4f **Usage:** CALL GDHES (*8 parameters*) **Precision:** Single (Double: DGDHES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GEL.** To solve a system of linear equations, i.e., the matrix equation $AX = B$, where A is a full matrix. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a1 **Usage:** CALL GEL(*11 parameters*) **Precision:** Single (Half: HGEL) **Availability:** 205 (vectorized)

**GENBUN.** Solves certain block tridiagonal systems of linear equations arising from Helmholtz or Poisson equations in two Cartesian coordinates. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b4b **Usage:** CALL GENBUN(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**GEOCDF.** Computes the geometric cumulative distribution function value at the value X with parameter P. *Fortran subroutine in DATAPAC.* **Classes:** L5a1g **Usage:** CALL GEOCDF(X, P, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GEOMETRIC PPCC PLOT.** Generates a probability plot correlation coefficient plot for the geometric distribution (plot of probability plot correlation coefficient vs. parameter P for P ranging from 0 to 1, or in a user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3g **Usage:** GEOMETRIC PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**GEOMETRIC PROBABILITY PLOT.** Generates a probability plot for the geometric distribution with parameter P. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2g **Usage:** GEOMETRIC PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**GEOPLT.** Generates a geometric probability plot with parameter P. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2g **Usage:** CALL GEOPLT(X, N, P) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GEOPPF.** Computes the percent point function value for the geometric distribution with parameter PPAR. *Fortran subroutine in DATAPAC.* **Classes:** L5a2g **Usage:** CALL GEOPPF(P, PPAR, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GEORAN.** Generates a random sample of size N from the geometric distribution with parameter P. *Fortran subroutine in DATAPAC.* **Classes:** L6a7 **Usage:** CALL GEORAN(N, P, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**GFARF.** Compute and plot gain and phase functions of autoregressive or difference filter. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c3 **Usage:** CALL GFARF (PHI, IAR) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**GFARFS.** Compute and optionally plot gain and phase functions of autoregressive or difference filter; with user-supplied control values; return gain and phase function values and corresponding frequency values. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c3 **Usage:** CALL GFARFS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**GFIT.** Chi-squared goodness of fit test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1c **Usage:** CALL GFIT (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GFSLF.** Compute and plot gain function of symmetric linear filter. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL GFSLF (H, K) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**GFSLFS.** Compute and optionally plot gain function of symmetric linear filter with user-supplied control values; return gain function values and corresponding frequency values. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL GFSLFS (*9 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**GGAMR.** One parameter gamma random deviate generator, and usable as basis for 2 parameter gamma, exponential, chi-squared, chi, beta, t and F deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a7 **Usage:** CALL GGAMR (DSEED, A, NR, WK, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGBN.** Binomial random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a2 **Usage:** CALL GGBN (DSEED, NR, NIND, P, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGBNR.** Negative binomial random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a14 **Usage:** CALL GGBNR (DSEED, K, P, NR, WK, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGBTR.** Beta random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a2 **Usage:** CALL GGBTR (DSEED, P, Q, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGCAY.** Cauchy random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a3 **Usage:** CALL GGCAY (DSEED, NR, WK, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGCHS.** Chi-squared random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a3 **Usage:** CALL GGCHS (DSEED, N, R, CHI2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGCOR.** Generate a random orthogonal matrix and a random correlation matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6b3, L6b15 **Usage:** CALL GGCOR (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGDA.** General discrete distribution random deviate generator using alias method. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a7 **Usage:** CALL GGDA (DSEED, NR, NDMP, P, IA, WK, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGDT.** General discrete distribution random deviate generator using table lookup method. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a7 **Usage:** CALL GGDT (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGEOT.** Geometric random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a7 **Usage:** CALL GGEOT (DSEED, NR, P, WK, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGEXN.** Exponential random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a5 **Usage:** CALL GGEXN (DSEED, XM, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGEXT.** Random deviate generator for a mixture of two exponentials. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a5 **Usage:** CALL GGEXT (DSEED, P, XM1, XM3, NR, R, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGHPR.** Hypergeometric random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a8 **Usage:** CALL GGHPR (DSEED, N, L, M, NR, WK, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGMTN.** Multinomial random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6b13 **Usage:** CALL GGMTN (DSEED, NR, NIND, K, P, IIR, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNLG.** Lognormal random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a12 **Usage:** CALL GGNLG (DSEED, NR, XM, S, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNML.** Normal or Gaussian random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a14 **Usage:** CALL GGNML (DSEED, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNO.** Generate set of order statistics from normal distribution. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a14 **Usage:** CALL GGNO (DSEED, IFIRST, ILAST, N, R, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNPM.** Normal random deviate generator via the polar method. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a14 **Usage:** CALL GGNPM (DSEED, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNPP.** Nonhomogeneous Poisson process generator with rate function lambda(t) – fixed interval, fixed number, or one at a time. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a16 **Usage:** CALL GGNPP (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNQF.** Normal random deviate generator. Function form of GGNML. *Proprietary Fortran function in IMSL.* **Classes:** L6a14 **Usage:** R = GGNQF(DSEED) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGNSM.** Multivariate normal random deviate generator with given covariance matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6b14 **Usage:** CALL GGNSM (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGPER.** Generate a random permutation of the integers 1 to k. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a16 **Usage:** CALL GGPER (DSEED, K, IPER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGPON.** Poisson random deviate generator where the Poisson parameter changes frequently. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a16 **Usage:** CALL GGPON (RLAM, DSEED, NR, IR, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGPOS.** Poisson random deviate generator where the Poisson parameter does not change often. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a16 **Usage:** CALL GGPOS (RLAM, DSEED, NR, IR, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGSPH.** Generation of uniform random deviates from the surface of the unit sphere in 3 or 4 space. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6b21 **Usage:** CALL GGSPH (DSEED, NR, IOPT, IZ, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGSRS.** Generate a simple random sample from a finite population. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a19 **Usage:** CALL GGSRS (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGSTA.** Stable distribution random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a19 **Usage:** CALL GGSTA (DSEED, ALPHA, BPRIM, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGTAB.** Generate a random contingency table with given row and column totals. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6b3 **Usage:** CALL GGTAB (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGTRA.** Triangular distribution random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a20 **Usage:** CALL GGTRA (DSEED, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGUBFS.** Basic uniform (0,1) random number generator. Function form of GGUBS. *Proprietary Fortran function in IMSL.* **Classes:** L6a21 **Usage:** R = GGUBFS(DSEED) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGUBS.** Basic uniform (0,1) pseudo-random number generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a21 **Usage:** CALL GGUBS (DSEED, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**GGUBT.** Uniform (0,1) pseudo-random number generator using alternate multiplier. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a21 **Usage:** CALL GGUBT (DSEED, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGUD.** Discrete uniform random number generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a21 **Usage:** CALL GGUD (DSEED, KN, R, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGUES.** Generate points in an N-dimensional space. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G4d **Usage:** CALL GGUES (N, A, B, K, IDO, S) **Precision:** Single (Double: DGGUES) **Availability:**

855NOS, 855VE, 205, 840NOS.

**GGUO.** Generate set of order statistics from uniform (0,1) distribution. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a21 **Usage:** CALL GGUO (DSEED, IFIRST, ILAST, N, R, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGUW.** Uniform (0,1) random number generator with shuffling. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a21 **Usage:** CALL GGUW (DSEED, NR, IOPT, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGVCR.** General continuous distribution random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a7 **Usage:** CALL GGVCR (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGVMS.** Von Mises random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a22 **Usage:** CALL GGVMS (DSEED, C, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GGWIB.** Weibull random deviate generator. *Proprietary Fortran subroutine in IMSL.* **Classes:** L6a23 **Usage:** CALL GGWIB (DSEED, A, NR, R) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GLM.** Performs simple and multiple least-squares regression, analysis of variance (especially for unbalanced data), analysis of covariance, response-surface regression, polynomial regression, partial correlation, multivariate analysis of variance, and repeated measures analysis of variance. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L7d3, L7e **Precision:** Single **Availability:** CAMVAX.

**GPICG.** Compute the performance index for a generalized complex eigensystem $Az=\lambda Bz$. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = GPICG(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GPIRG.** Compute the performance index for a generalized real eigensystem $Az=\lambda Bz$. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = GPIRG(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GPISP.** Compute the performance index for a generalized real symmetric eigensystem problem. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D4c **Usage:** R = GPISP(*9 parameters*) **Precision:** Single (Double: DGPISP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GQ0IN.** Finds the abscissae and weights for Gauss-Laguerre quadrature on the interval (0, +infinity). *Proprietary Fortran subroutine in PORT.* **Classes:** H2c **Usage:** CALL GQ0IN (N, X, W) **Precision:** Single (Double: DGQ0IN) **Availability:** 855NOS, 205.

**GQM11.** Finds the abscissae and weights for Gauss Legendre quadrature on the interval (-1,1). *Proprietary Fortran subroutine in PORT.* **Classes:** H2c **Usage:** CALL GQM11 (N, X, W) **Precision:** Single (Double: DGQM11) **Availability:** 855NOS, 205.

**GQRCF.** Compute a Gauss, Gauss-Radau or Gauss-Lobatto quadrature rule given the recurrence coefficients for the monic polynomials orthogonal with respect to the weight function. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2c **Usage:** CALL GQRCF (*7 parameters*) **Precision:** Single (Double: DGQRCF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GQRUL.** Compute a Gauss, Gauss-Radau or Gauss-Lobatto quadrature rule with various classical weight functions. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2c **Usage:** CALL GQRUL (*8 parameters*) **Precision:** Single (Double: DGQRUL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GRPES.** Compute basic statistics from grouped data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L1a3 **Usage:** CALL GRPES (*6 parameters*) **Precision:** Single (Double: DGRPES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTCN.** Sample size or number of class intervals determination for chi-squared test applications. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1c **Usage:** CALL GTCN (Q, IOPT, B, K, N, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTD2T.** The d-square test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1d **Usage:** CALL GTD2T (COUNT, K, E, CS, STD, Q, IER) **Also see:** GTDDU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTDDU.** D-square tally. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL GTDDU (R, M, COUNT, K, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTMNT.** Moments and standardized moments of uniform random numbers. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4u **Usage:** CALL GTMNT (RBAR, R, N, NR, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTNOR.** Test for normality of random deviates. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a4n **Usage:** CALL GTNOR (R, N, K, STAT, OBSC, CSOBS, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTPBC.** Count of the number of zero bits in a given subset of a real word. *Proprietary Fortran subroutine in IMSL.* **Classes:** N2 **Usage:** CALL GTPBC (I1, I2, R, KZERO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTPKP.** Probability distribution of n elements into two equi-probable states. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1b **Usage:** CALL GTPKP (N, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTPL.** Poker test tally of hand types and statistics. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL GTPL (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTPOK.** The poker test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a5 **Usage:** CALL GTPOK (*7 parameters*) **Also see:** GTPL **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTPR.** Tally of coordinates of pairs (or lagged pairs) of random numbers. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL GTPR (R, N, K, L, A, IA, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTPST.** Pairs test or Goods serial test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1d **Usage:** CALL GTPST (A, IA, K, CS, Q, STD, IER) **Also see:** GTCN GTPR **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTRN.** Runs test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1d **Usage:** CALL GTRN (RUNS, N, E, CS, Q, STD, IER) **Also see:** GTRTN **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTRTN.** Tally of number of runs up and down. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL GTRTN (R, N, IOPT, WK, RUNS) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTTRT.** Tally for triplets test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2b **Usage:** CALL GTTRT (R, N, K, A, IA1, IA2, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GTTT.** Triplets test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1d **Usage:** CALL GTTT (A, IA1, IA2, K, CS, Q, STD, IER) **Also see:** GTCN GTTRT **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GVCCG.** Compute all of the eigenvalues and eigenvectors of a generalized complex eigensystem $Az=\lambda Bz$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4b4 **Usage:** CALL GVCCG (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GVCRG.** Compute all of the eigenvalues and eigenvectors of a generalized real eigensystem $Az=\lambda Bz$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4b2 **Usage:** CALL GVCRG (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GVCSP.** Compute all of the eigenvalues and eigenvectors of the generalized real symmetric eigenvalue problem $Az=\lambda Bz$, with B symmetric positive definite. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4b1 **Usage:** CALL GVCSP (*8 parameters*) **Precision:** Single (Double: DGVCSP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**GVLCG.** Compute all of the eigenvalues of a generalized complex eigensystem $Az=\lambda Bz$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4b4 **Usage:** CALL GVLCG (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GVLRG.** Compute all of the eigenvalues of a generalized real eigensystem $Az=\lambda Bz$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D4b2 **Usage:** CALL GVLRG (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**GVLSP.** Compute all of the eigenvalues of the generalized real symmetric eigenvalue problem $Az=\lambda Bz$, with B symmetric positive definite. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D4b1 **Usage:** CALL GVLSP (N, A, LDA, B, LDB, EVAL) **Precision:** Single (Double: DGVLSP) **Availability:** 855NOS, 855VE, 205, 840NOS.

# H

**H02BAF.** Solves the integer linear programming problem, with all integer coefficients, via Gomory's method. It is enhanced by including the technique known as Wilson's cuts. *Proprietary Fortran subroutine in NAG.* **Classes:** G2c6 **Usage:** CALL H02BAF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**H03ABF.** Solves the classical Transportation ( Hitchcock ) problem. *Proprietary Fortran subroutine in NAG.* **Classes:** G2b **Usage:** CALL H03ABF (*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**H2A1.** Automatically evaluates the definite integral of a user defined function of one variable. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** H2a1a1 **Usage:** CALL H2A1(A, B, EPS, R, E, KF, KE) **Precision:** Single **Availability:** PC.

**H2A1T.** Computes the integral of the array f between $x_i$ and $x_j$, given n points in the plane ($x_k$, $f_k$), k=1, ..., n. *Proprietary Fortran function in the Scientific Desk.* **Classes:** H2a1b2 **Usage:** R = H2A1T(N, X, F, I, J, W, KE) **Precision:** Single **Availability:** PC.

**H2A1U.** Automatically evaluates the definite integral of a user defined function of one variable. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** H2a1a1 **Usage:** CALL H2A1U(*14 parameters*) **Precision:** Single **Availability:** PC.

**H2B2A.** Computes the two-dimensional integral of a function f over a region consisting of n triangles. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** H2b2a1 **Usage:** CALL H2B2A(*17 parameters*) **Precision:** Single **Availability:** PC.

**HALFNORMAL PROBABILITY PLOT.** Generates a halfnormal probability plot with mean $\sqrt{2/\pi}$ and standard deviation 1. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2h **Usage:** HALFNORMAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**HAZEZ.** Perform nonparametric hazard rate estimation using kernel functions. Easy-to-use version of HAZRD. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L15 **Usage:** CALL HAZEZ (*14 parameters*) **Precision:** Single (Double: DHAZEZ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HAZRD.** Perform nonparametric hazard rate estimation using kernel functions and quasi-likelihoods. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L15 **Usage:** CALL HAZRD (*22 parameters*) **Precision:** Single (Double: DHAZRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HAZST.** Perform hazard rate estimation over a grid of points using a kernel function. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L15 **Usage:** CALL HAZST (*13 parameters*) **Precision:** Single (Double: DHAZST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HBDGEL.** *Half precision version of BDGEL.*

**HCMPRS.** Compress out selected half-word elements from a half-word source vector, storing them contiguously in memory. The hardware instruction CPSV is used. *Fortran/meta function in MAGEV.* **Classes:** D1a11 **Usage:** I = HCMPRS(N, A, IA, B, IB, BIT) **Precision:** Half **Availability:** 205 (vectorized)

**HECOMP.** Performs Householder reduction of a rectangular matrix to upper triangular form. Usually used in conjunction with HOLVE to find the least squares solution of an overdetermined linear system. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** D9a1 **Precision:** Single **Availability:** 855NOS.

**HFFT1D.** *Half precision version of FFT1D.*

**HFNCDF.** Computes the cumulative distribution function value for the halfnormal distribution with mean $\sqrt{2/\pi}$ and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1h **Usage:** CALL HFNCDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**HFNPLT.** Generates a halfnormal probability plot with mean $\sqrt{2/\pi}$ and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2h **Usage:** CALL HFNPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**HFNPPF.** Computes the percent point function value for the halfnormal distribution with mean $\sqrt{2/\pi}$ and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2h **Usage:** CALL HFNPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**HFNRAN.** Generates a random sample of size N from the halfnormal distribution with mean $\sqrt{2/\pi}$ and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L6a8 **Usage:** CALL HFNRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**HFTI.** Solves linear least squares problem AX=B. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** D9a1 **Usage:** CALL HFTI(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HGEL.** *Half precision version of GEL.*

**HHSTP.** Print a horizontal histogram. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3a1

**Usage:** CALL HHSTP (*8 parameters*) **Precision:** Single (Double: DHHSTP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HILOGLINEAR.** Estimates parameters of hierarchical log-linear models for frequency tables by iterative proportional fitting. Options: backward elimination, cell weights, plots, algorithm tuning parameters, plots, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L9c **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**HIMTQL.** *Half precision version of IMTQL.*

**HINVSPD.** *Half precision version of INVSPD.*

**HINVU.** *Half precision version of INVU.*

**HIPASS.** Filter series with symmetric linear high-pass filter; return filtered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL HIPASS (Y, N, FC, K, HHP, YF, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**HIST.** Prints a univariate histogram. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3a1 **Usage:** CALL HIST(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**HIST.** Produces 2 histograms (with differing class widths) of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L3a1 **Usage:** CALL HIST(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**HIST.** Compute and print a histogram and summary statistics, with automatic selection of number of cells and cell boundaries. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a1 **Usage:** CALL HIST (Y, N, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**HISTC.** Compute and print a histogram and summary statistics with user control of number of cells and cell boundaries. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a1 **Usage:** CALL HISTC (*6 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**HISTOGRAM.** Generates one of the following plots: histogram, relative histogram, cumulative histogram, relative cumulative histogram. Optional control of upper and lower limits and/or class width. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a1 **Usage:** [RELATIVE/CUMULATIVE/RELATIVE CUMULATIVE] HISTOGRAM [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**HISTOGRAM.** Prints a histogram of the values in each of one or more vectors, with optional user-specification of the first midpoint and the interval width. *Command in MINITAB proprietary interactive system.* **Classes:** L3a1, L3e1 **Usage:** HISTogram of C [... and C, ] [first midpoint K, interval width K] **Precision:** Single **Availability:** 855NOS.

**HMERGE.** To merge elements of two half-word vectors, or of a scalar and a vector, into a half-word result vector. The Cyber 205 hardware instruction MRGV is utilized. *Fortran/meta subroutine in MAGEV.* **Classes:** N7 **Usage:** CALL HMERGE(N, A, IA, B, IB, C, IC, BIT) **Precision:** Half **Availability:** 205 (vectorized)

**HMPYUTU.** *Half precision version of MPYUTU.*

**HMXCMP.** *Half precision version of MXCMP.*

**HMXEQ.** *Half precision version of MXEQ.*

**HMXMOV.** *Half precision version of MXMOV.*

**HMXMPY.** *Half precision version of MXMPY.*

**HMXMPYI.** *Half precision version of MXMPYI.*

**HMXMPYT.** *Half precision version of MXMPYT.*

**HOMOSCEDASTICITY PLOT.** Generate a homoscedasticity plot. Horizontal axis the subset standard deviation and the vertical axis the subset mean. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a3 **Usage:** HOMOSCEDASTICITY PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**HPCOEF.** Compute symmetric linear high-pass filter coefficients; return coefficients. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL HPCOEF (HLP, K, HHP) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**HQR.** Computes eigenvalues of a real upper Hessenberg matrix using the QR method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL HQR(NM, N, LOW, IGH, H, WR, WI, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**HQR2.** Computes eigenvalues and eigenvectors of real upper Hessenberg matrix using QR method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL HQR2(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HREV.** To reverse the elements of a half-word vector. *Fortran/meta subroutine in MAGEV.* **Classes:** N8 **Usage:** CALL HREV(N, A, IA, B, IB) **Precision:** Half **Availability:** 205 (vectorized)

**HRRRR.** Compute the Hadamard product of two real matrices. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b **Usage:** CALL HRRRR(*12 parameters*) **Precision:** Single (Double: DHRRRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HS3CRI.** Initialization routine for HS3CRT. *Portable Fortran subroutine in CMLIB (VHS3 sublibrary).* **Usage:** CALL HS3CRI(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**HS3CRT.** Sets up and solves the standard seven-point finite difference approximation on a staggered grid to the Helmholtz equation in Cartesian coordinates with a variety of possible boundary conditions. *Portable Fortran subroutine in CMLIB (VHS3 sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HS3CRT(*11 parameters*) **Also see:** HS3CRI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**HSADDHV.** *Half precision version of FSADDFV.*

**HSDIVHV.** *Half precision version of FSDIVFV.*

**HSMPYHV.** *Half precision version of FSMPYFV.*

**HSSUBHV.** *Half precision version of FSSUBFV.*

**HSTCRT.** Solves the Helmholtz or Poisson equations in two dimensions in Cartesian coordinates on a staggered grid. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HSTCRT(*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HSTCSP.** Solves a modified Helmholtz equation in spherical coordinates with axisymmetry using a staggered grid. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HSTCSP(*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HSTCYL.** Solves a modified Helmholtz equation in cylindrical coordinates on a staggered grid. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HSTCYL(*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HSTOHV.** *Half precision version of FSTOFV.*

**HSTPLR.** Solves the Helmholtz or Poisson equation in polar coordinates on a staggered grid. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HSTPLR(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HSTSSP.** Solves the Helmholtz or Poisson equation in spherical coordinates on the surface of a sphere using a staggered grid. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HSTSSP(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HSYMSTO.** *Half precision version of SYMSTO.*

**HSYMUUT.** *Half precision version of SYMUUT.*

**HTPMOV.** *Half precision version of TPMOV.*

**HTRED2.** *Half precision version of TRED2.*

**HTRIB3.** Computes eigenvectors of complex Hermitian matrix from eigenvectors of real symmetric tridiagonal matrix output from HTRID3. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL HTRIB3(NM, N, A, TAU, M, ZR, ZI) **Also see:** HTRID3 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**HTRIBK.** Forms eigenvectors of complex Hermitian matrix from eigenvectors of real symmetric tridiagonal matrix output from HTRIDI. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL HTRIBK(NM, N, AR, AI, TAU, M, ZR, ZI) **Also see:** HTRIDI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**HTRID.** *Half precision version of TRID.*

**HTRID3.** Reduces complex Hermitian (packed) matrix to real symmetric tridiagonal matrix by unitary similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b1 **Usage:** CALL HTRID3(NM, N, A, D, E, E2, TAU) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HTRIDI.** Reduces complex Hermitian matrix to real symmetric tridiagonal matrix using unitary similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b1 **Usage:** CALL HTRIDI(NM, N, AR, AI, D, E, E2, TAU) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**HUMSL.** Minimizes a general unconstrained objective function using (analytic) gradient and Hessian provided by the user. *Portable Fortran subroutine in CMLIB (NL2SN sublibrary).* **Classes:** G1b1c **Usage:** CALL HMSL(*12 parameters*) **Precision:** Single (Double: DHUMSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HVADDHV.** *Half precision version of FVADDFV.*

**HVDIVHV.** *Half precision version of FVDIVFV.*

**HVMPYHV.** *Half precision version of FVMPYFV.*

**HVSUBHV.** *Half precision version of FVSUBFV.*

**HVTOHV.** *Half precision version of FVTOFV.*

**HW3CRT.** Solves the Helmholtz or Poisson equation in three dimensions using Cartesian coordinates. *Portable*

*Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HW3CRT(*25 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HWSCRT.** Solves the Helmholtz or Poisson equation in two dimensions in Cartesian coordinates. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HWSCRT(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HWSCSP.** Solves a modified Helmholtz equation in spherical coordinates with axisymmetry. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HWSCSP(*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HWSCYL.** Solves a modified Helmholtz equation in cylindrical coordinates. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HWSCYL(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HWSPLR.** Solves the Helmholtz or Poisson equation in polar coordinates. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HWSPLR(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HWSSSP.** Solves the Helmholtz or Poisson equation in spherical coordinates on the surface of a sphere. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a1a **Usage:** CALL HWSSSP(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**HYPDF.** Evaluate the hypergeometric distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1h **Usage:** R = HYPDF(K, N, M, L) **Precision:** Single (Double: DHYPDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HYPOT.** SQRT($A^2+B^2$) without underflow or overflow. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** C2 **Usage:** R = HYPOT(A, B) **Precision:** Single (Double: DHYPOT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**HYPPR.** Evaluate the hypergeometric probability function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1h **Usage:** R = HYPPR(K, N, M, L) **Precision:** Single (Double: DHYPPR) **Availability:** 855NOS, 855VE, 205, 840NOS.

# I

**I PLOT.** Plots I plot(s) – median, minimum, and maximum – for either univariate distributional analysis, analysis of one-factor models, or display of uncertainty intervals about estimates. Default plot character for the estimator is "x" and for the top and the bottom of the uncertainty interval is "-". *Command(s) in DATAPLOT interactive system.* **Classes:** L3a3, L3e4 **Usage:** IPLOT <RESPONSE VARIABLE> [<INDEPENDENT VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**I1A1A.** Integrates a system of neqn first order ordinary differential equations of the form dy(i)/dt = f(t,y(1),y(2),...,y(neqn)), where the y(i) are given at t (Runge-Kutta-Fehlberg method). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** I1a1a **Usage:** CALL I1A1A(*10 parameters*) **Precision:** Single (Double: I1A1AD) **Availability:** PC.

**I1A1AD.** *Double precision version of I1A1A.*

**I1A2.** Numerical integration of initial value problems for ordinary differential equations, implicit equations, sparse Jacobians, root finding. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** I1a1b, I1a2 **Usage:** CALL I1A2(*28 parameters*) **Precision:** Single (Double: I1A2D) **Availability:** PC.

**I1A2D.** *Double precision version of I1A2.*

**I1A2E.** Numerical integration of initial value problems for ordinary differential equations, Gear stiff formulas; easy to use. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** I1a1b, I1a2 **Usage:** CALL I1A2E(*9 parameters*) **Precision:** Single (Double: I1A2ED) **Availability:** PC.

**I1A2ED.** *Double precision version of I1A2E.*

**I1A2F.** Numerical integration of initial value problems for ordinary differential equations, Gear/Adams formulas, root finding. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** I1a1b, I1a2 **Usage:** CALL I1A2F(*16 parameters*) **Precision:** Single (Double: I1A2FD) **Availability:** PC.

**I1A2FD.** *Double precision version of I1A2F.*

**I1MACH.** Provides integer machine dependent information, e.g., largest integer. *Portable Fortran function in CMLIB (MACHCON sublibrary).* **Classes:** R1 **Usage:** I = I1MACH(I) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**I1MACH.** Provides integer machine dependent information, e.g., largest integer. *Proprietary Fortran function in PORT.* **Classes:** R1 **Usage:** I = I1MACH(I) **Precision:** Single **Availability:** 855NOS, 205.

**IACHAR.** Return the integer ASCII value of a character argument. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N3 **Usage:** I = IACHAR(CH) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IADD.** Add a scalar to each component of a vector, x = x + a, all integer. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL IADD (N, IA, IX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IBCCCU.** Bicubic spline two-dimensional coefficient calculator. *Proprietary Fortran subroutine in IMSL.* **Classes:** E2a **Usage:** CALL IBCCCU(F, X, NX, Y, NY, C, IC, WK, IER) **Also see:** IBCEVL DBCEVL **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IBCEVL.** Evaluation of a bicubic spline. *Proprietary Fortran subroutine in IMSL.* **Classes:** E3a1, K6a1 **Usage:** CALL IBCEVL (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**IBCIEU.** Bicubic spline two-dimensional interpolator. *Proprietary Fortran subroutine in IMSL.* **Classes:** E2a **Usage:** CALL IBCIEU (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICAMAX.** Find smallest index of maximum magnitude component of a complex vector. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a2, D1a3c, N5a **Usage:** I = ICAMAX(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**ICAMAX.** Find the smallest index of the component of a complex vector having maximum magnitude. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, D1a3c, N5a **Usage:** I = ICAMAX(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ICAMIN.** Find the smallest index of the component of a complex vector having minimum magnitude. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = ICAMIN(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ICASE.** Convert from character to the integer ASCII value without regard to case. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N3 **Usage:** I = ICASE(CH) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ICEIL.** Smallest integer greater than or equal to x. Input is real, output is integer. *Proprietary Fortran function in*

*PORT.* **Classes:** C1 **Usage:** I = ICEIL(X) **Precision:** Single (Double: IDCEIL) **Availability:** 855NOS, 205.

**ICOPY.** Copy a vector X to a vector Y, both integer. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a5 **Usage:** CALL ICOPY (N, IX, INCX, IY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ICSCCU.** Cubic spline interpolation. *Proprietary Fortran subroutine in IMSL.* **Classes:** E1a **Usage:** CALL ICSCCU (X, Y, NX, C, IC, IER) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSEVU.** Evaluation of a cubic spline. *Proprietary Fortran subroutine in IMSL.* **Classes:** E3a1, K6a1 **Usage:** CALL ICSEVU (X, Y, NX, C, IC, U, S, M, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSFKU.** Least squares approximation by cubic splines with fixed knots. *Proprietary Fortran subroutine in IMSL.* **Classes:** K1a1a1 **Usage:** CALL ICSFKU (*12 parameters*) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSICU.** Interpolatory approximation by cubic splines with arbitrary second derivative end conditions. *Proprietary Fortran subroutine in IMSL.* **Classes:** E1a **Usage:** CALL ICSICU (X, Y, NX, BPAR, C, IC, IER) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSMOU.** One-dimensional data smoothing by error detection. *Proprietary Fortran subroutine in IMSL.* **Classes:** K5 **Usage:** CALL ICSMOU (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSPLN.** Cubic spline interpolation with periodic end conditions. *Proprietary Fortran subroutine in IMSL.* **Classes:** E1a **Usage:** CALL ICSPLN (X, Y, NX, C, IC, WK, IER) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSSCU.** Cubic spline data smoother. *Proprietary Fortran subroutine in IMSL.* **Classes:** K5 **Usage:** CALL ICSSCU (*10 parameters*) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSSCV.** Cubic spline data smoother. *Proprietary Fortran subroutine in IMSL.* **Classes:** K5 **Usage:** CALL ICSSCV (*9 parameters*) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ICSVKU.** Least squares approximation by cubic splines with variable knots. *Proprietary Fortran subroutine in IMSL.* **Classes:** K1a1a1 **Usage:** CALL ICSVKU (*11 parameters*) **Also see:** ICSEVU DCSEVU **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**IDAMAX.** *Double precision version of ISAMAX.*

**IDAMIN.** *Double precision version of ISAMIN.*

**IDCEIL.** *Double precision version of ICEIL.*

**IDFLR.** *Double precision version of IFLR.*

**IDLUMB.** *Double precision version of ILUMB.*

**IDLUMD.** *Double precision version of ILUMD.*

**IDMAX.** *Double precision version of ISMAX.*

**IDMIN.** *Double precision version of ISMIN.*

**IDMNPB.** *Double precision version of IMNPB.*

**IDPUMB.** *Double precision version of IPUMB.*

**IDPUMD.** *Double precision version of IPUMD.*

**IDUMB.** *Double precision version of IUMB.*

**IDUMD.** *Double precision version of IUMD.*

**IDYWK.** Compute the day of the week for a given date. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** R **Usage:** I = IDYWK(IDAY, IMONTH, IYEAR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IERCD.** Retrieve the code for an informational error. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** R3c **Usage:** I = IERCD() **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IFLR.** Largest integer less than or equal to x. Input is real, output is integer. *Proprietary Fortran function in PORT.* **Classes:** C1 **Usage:** I = IFLR(X) **Precision:** Single (Double: IDFLR) **Availability:** 855NOS, 205.

**IFLSQ.** Least squares approximation with user supplied functions. *Proprietary Fortran subroutine in IMSL.* **Classes:** K1a1a3 **Usage:** CALL IFLSQ (F, X, Y, M, A, N, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IICSR.** Compare two character strings using the ASCII collating sequence but without regard to case. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N3 **Usage:** I = IICSR(STR1, STR2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IIDEX.** Determine the string position indicating the starting position at which a key character sequence begins without regard to case. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N3, N5c **Usage:** I = IIDEX(CHRSTR, KEY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IIMAX.** Find the smallest index of the maximum component of an integer vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = IIMAX(N, IX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IIMIN.** Find the smallest index of the minimum of an integer vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = IIMIN(N, IX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ILUMB.** Given a basic mesh, this subdivides each interval into the same number of uniformly spaced points for B-spline use. *Proprietary Fortran function in PORT.* **Classes:** E3b, K6b **Usage:** I = ILUMB(XB, NXB, N, K, NX) **Precision:** Single (Double: IDLUMB) **Availability:** 855NOS, 205.

**ILUMD.** Given a basic mesh, this subdivides each interval into the same number or uniformly spaced points for B-spline use. *Proprietary Fortran function in PORT.* **Classes:** E3b, K6b **Usage:** I = ILUMD(XB, NXB, N, NX) **Precision:** Single (Double: IDLUMD) **Availability:** 855NOS, 205.

**IMACH.** Retrieve integer machine constants. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** R1 **Usage:** I = IMACH(N) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IMNPB.** Creates a B-spline mesh from an array of fitting points, using at least n fitting points in each mesh interval. *Proprietary Fortran function in PORT.* **Classes:** E3b, K6b **Usage:** I = IMNPB(X, NX, N, K, NT) **Precision:** Single (Double: IDMNPB) **Availability:** 855NOS, 205.

**IMTQL.** To find all eigenvalues and, optionally, all eigenvectors of a symmetric tridiagonal matrix. The complete eigensystem of a full symmetric matrix can also be determined, provided the matrix has first been made tridiagonal by calling HTRED2 or TRED2. *Fortran/meta subroutine in MAGEV.* **Classes:** D4a5, D4a1 **Usage:** CALL IMTQL(*9 parameters*) **Precision:** Single (Half: HIMTQL) **Availability:** 205 (vectorized)

**IMTQL1.** Computes eigenvalues of symmetric tridiagonal matrix using implicit QL method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL IMTQL1(N, D, E, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**IMTQL2.** Computes eigenvalues and eigenvectors of symmetric tridiagonal matrix using implicit QL method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL IMTQL2(NM, N, D, E, Z, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**IMTQLV.** Computes eigenvalues of symmetric tridiagonal matrix using implicit QL method. Eigenvectors may be computed later. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL IMTQLV(N, D, E, E2, W, IND, IERR, RV1) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**IN.** Reads in raw data matrix and labels for the cases and variables and sets up the data structures necessary for most of the subroutines in the CLUSTER sublibrary. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** CALL IN(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**INCLD.** Perform an includance test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b1b **Usage:** CALL INCLD (*10 parameters*) **Precision:** Single (Double: DINCLD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**INITDS.** *Double precision version of INITS.*

**INITS.** Returns the number of terms of a given orthogonal series needed to insure that the error is no larger than the requested accuracy. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C3a **Usage:** I = INITS(OS, NOS, ETA) **Precision:** Single (Double: INITDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**INLAP.** Inverse Laplace transform of a complex function. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J3 **Usage:** CALL INLAP (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**INTRV.** Computes the index into a knot or breakpoint sequence corresponding to a given point X. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3c, K6c **Usage:** CALL INTRV(KT, LXT, X, ILD, ILEFT, MFLAG) **Precision:** Single (Double: DINTRV) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**INTRVD.** *Double precision version of INTRVR.*

**INTRVI.** Finds the interval in an integer array to which an integer element belongs. *Proprietary Fortran function in PORT.* **Classes:** N5b **Usage:** I = INTRVI(N, I, II) **Precision:** Single **Availability:** 855NOS, 205.

**INTRVR.** Finds the interval in a real array to which a real element belongs. *Proprietary Fortran function in PORT.* **Classes:** N5b **Usage:** I = INTRVR(N, R, RR) **Precision:** Single (Double: INTRVD) **Availability:** 855NOS, 205.

**INVAR1.** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results. Line printer graphics only. *INVAR is an interactive system.* **Classes:** K1b1a1, K1b1a2, L8e1 **Precision:** Single **Availability:** 855NOS.

**INVAR2.** Interactive program for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results and DISSPLA graphics. *INVAR is an interactive system.* **Classes:** K1b1a1, K1b1a2, L8e1 **Precision:** Single **Availability:** 855NOS.

**INVERT.** Computes the inverse and determinant of a symmetric matrix (e.g. a covariance matrix). *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** CALL INVERT(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**INVIT.** Computes eigenvectors of upper Hessenberg (real) matrix associated with specified eigenvalues by inverse iteration. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2b **Usage:** CALL INVIT(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**INVSPD.** To compute the inverse of a positive definite symmetric matrix. *Fortran/meta subroutine in MAGEV.* **Classes:** D2b1b **Usage:** CALL INVSPD(N, A, KA, D, NERR, NLP) **Precision:** Single (Half: HINVSPD) **Availability:** 205 (vectorized)

**INVU.** To compute the upper triangular inverse R of a non-singular upper triangular matrix U. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a3 **Usage:** CALL INVU(N, U, KU, R, KR, D, NLP) **Precision:** Single (Half: HINVU) **Availability:** 205 (vectorized)

**IPGM.** Compute and print an integrated periodogram analysis of a series; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL IPGM (YFFT, N, LYFFT, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**IPGMP.** Compute and print an integrated periodogram analysis of a series; input periodogram rather than original series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL IPGMP (PER, FREQ, NF, N, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**IPGMPS.** Compute and optionally print an integrated periodogram analysis of a series; input periodogram rather than original series; return integrated periodogram and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL IPGMPS (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**IPGMS.** Compute and optionally print an integrated periodogram analysis of a series; use FFT for computations; return integrated periodogram and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL IPGMS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**IPUMB.** Given a basic mesh, this subdivides each interval. Number of points per interval can vary, but uniform in each subdivision. *Proprietary Fortran function in PORT.* **Classes:** E3b, K6b **Usage:** I = IPUMB(XB, NXB, NA, K, NX) **Precision:** Single (Double: IDPUMB) **Availability:** 855NOS, 205.

**IPUMD.** Given a basic mesh, this subdivides each interval with a variable number of points. Points are uniform in each interval. *Proprietary Fortran function in PORT.* **Classes:** E3b, K6b **Usage:** I = IPUMD(XB, NXB, NA, NX) **Precision:** Single (Double: IDPUMD) **Availability:** 855NOS, 205.

**IQHSCU.** Visually pleasing interpolant of one dimensional data via piecewise cubic Hermite function. *Proprietary Fortran subroutine in IMSL.* **Classes:** E1a **Usage:** CALL IQHSCU (X, Y, NX, C, IC, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**IQHSCV.** Smooth surface fitting with irregularly distributed data points. *Proprietary Fortran subroutine in IMSL.* **Classes:** E2b **Usage:** CALL IQHSCV (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IRANDOM.** Generates K pseudo-random integers in a specified interval. *Command in MINITAB proprietary interactive system.* **Classes:** L6a21 **Usage:** IRANdom K random integers between K and K, put into C **Precision:** Single **Availability:** 855NOS.

**IRATCU.** Rational weighted Chebyshev approximation of a continuous function. *Proprietary Fortran subroutine in IMSL.* **Classes:** K2 **Usage:** CALL IRATCU (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IRLS.** Robust regression by iteratively reweighted least squares. Weights are determined from the data and are computed as functions of the scaled residuals. Options: $L_1$ and $L_2$ starts, eight weight functions, rank determination, and convergence criterion. *ROSEPAK is a portable interactive system.* **Classes:** L8a3, L8c4 **Precision:** Single **Availability:** 855NOS.

**IRNSE.** Compute estimates of the impulse response weights and noise series of a univariate transfer function model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10b2b **Usage:** CALL IRNSE (*14 parameters*) **Precision:** Single (Double: DIRNSE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISAMAX.** Find smallest index of maximum magnitude component of a single precision vector. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a2, D1a3c, N5a **Usage:** I = ISAMAX(N, SX, INCX) **Precision:**

Single (Double: IDAMAX) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**ISAMAX.** Find the smallest index of the component of a single-precision vector having maximum absolute value. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** D1a2, D1a3c, N5a **Usage:** I = ISAMAX(N, SX, INCX) **Precision:** Single (Double: IDAMAX) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISAMAX.** Find the smallest index of the component of a single-precision vector having maximum absolute value. *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = ISAMAX(N, SX, INCX) **Precision:** Single (Double: IDAMAX) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISAMIN.** Find the smallest index of the minimum magnitude component of a real vector. *Portable Fortran function in CMLIB (XBLAS sublibrary).* **Classes:** D1a2, N5a **Usage:** I = ISAMIN(N, SX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ISAMIN.** Find the smallest index of the component of a single-precision vector having minimum absolute value. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = ISAMIN(N, SX, INCX) **Precision:** Single (Double: IDAMIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISET.** Set the components of a vector to a scalar, all integer. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a1 **Usage:** CALL ISET (N, IA, IX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISMAX.** Find the smallest index of the maximum component of a real vector. *Portable Fortran function in CMLIB (XBLAS sublibrary).* **Classes:** D1a2 **Usage:** I = ISMAX(N, SX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ISMAX.** Find the smallest index of the component of a single-precision vector having maximum value. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = ISMAX(N, SX, INCX) **Precision:** Single (Double: IDMAX) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISMIN.** Find the smallest index of the minimum component of a real vector. *Portable Fortran function in CMLIB (XBLAS sublibrary).* **Classes:** D1a2, N5a **Usage:** I = ISMIN(N, SX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ISMIN.** Find the smallest index of the component of a single-precision vector having minimum value. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a2, N5a **Usage:** I = ISMIN(N, SX, INCX) **Precision:** Single (Double: IDMIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISORT.** Sorts an integer array in either increasing or decreasing order. Optionally another integer array can be carried along. *Portable Fortran subroutine in CMLIB (SSORT sublibrary).* **Classes:** N6a2a **Usage:** CALL ISORT(IX, IY, N, KFLAG) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ISORTDA.** To sort the elements of a real or integer array in a non-increasing, or descending, order with respect to absolute value. *Fortran/meta subroutine in MAGEV.* **Classes:** N6a2a, N6a2b **Usage:** CALL ISORTDA(N, X, IX) **Precision:** Single **Availability:** 205 (vectorized)

**ISRCH.** Search a sorted integer vector for a given integer and return its index. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N5c **Usage:** CALL ISRCH (N, IVALUE, IX, INCX, INDEX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISTKGT.** Allocates (gets) an array from the storage stack for PORT library programs. *Proprietary Fortran function in PORT.* **Classes:** N4 **Usage:** I = ISTKGT(NITEMS, ITYPE) **Availability:** 855NOS, 205.

**ISTKIN.** Initialize the length of the dynamic storage stack for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** N4 **Usage:** CALL ISTKIN (NITEMS, ITYPE) **Availability:** 855NOS, 205.

**ISTKMD.** Changes size of last stack allocation for PORT library programs. *Proprietary Fortran function in PORT.* **Classes:** N4 **Usage:** I = ISTKMD(NITEMS) **Availability:** 855NOS, 205.

**ISTKQU.** Returns the number of available items that remain in the stack for PORT library programs. *Proprietary Fortran function in PORT.* **Classes:** N4 **Usage:** I = ISTKQU(ITYPE) **Availability:** 855NOS, 205.

**ISTKRL.** Releases the last storage allocations requested for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** N4 **Usage:** CALL ISTKRL (NUMBER) **Availability:** 855NOS, 205.

**ISTKST.** Returns information on the status of the stack for PORT library programs. *Proprietary Fortran function in PORT.* **Classes:** N4 **Usage:** I = ISTKST(NFACT) **Availability:** 855NOS, 205.

**ISUB.** Subtract each component of a vector from a scalar, $x = a - x$, all integer. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL ISUB (N, IA, IX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISUM.** Sum the values of an integer vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** I = ISUM(N, IX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ISWAP.** Interchange vectors X and Y, both integer. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a5 **Usage:** CALL ISWAP (N, IX, INCX, IY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IUMB.** Given interval endpoints, this generates a uniform mesh for B-spline use. *Proprietary Fortran function* *in PORT.* **Classes:** E3b, K6b **Usage:** I = IUMB(A, B, NAB, K, NX) **Precision:** Single (Double: IDUMB) **Availability:** 855NOS, 205.

**IUMD.** Given interval endpoints, this generates a uniform mesh. *Proprietary Fortran function in PORT.* **Classes:** E3b, K6b **Usage:** I = IUMD(A, B, NAB) **Precision:** Single (Double: IDUMD) **Availability:** 855NOS, 205.

**IVPAG.** Solve an initial-value problem for ordinary differential equations using an Adams-Moulton or Gear method. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I1a1b **Usage:** CALL IVPAG (*10 parameters*) **Precision:** Single (Double: DIVPAG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**IVPBS.** Solve an initial-value problem for ordinary differential equations using the Bulirsch-Stoer extrapolation method. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I1a1c **Usage:** CALL IVPBS (*8 parameters*) **Precision:** Single (Double: DIVPBS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**IVPRK.** Solve an initial-value problem for ordinary differential equations using the Runge-Kutta-Verner fifth- and sixth-order method. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I1a1a **Usage:** CALL IVPRK (*8 parameters*) **Precision:** Single (Double: DIVPRK) **Availability:** 855NOS, 855VE, 205, 840NOS.

**IWKCIN.** Initialize bookkeeping locations describing the character workspace stack. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** N4 **Usage:** CALL IWKCIN (NELMTS, NALC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**IWKIN.** Initialize bookkeeping locations describing the workspace stack. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** N4 **Usage:** CALL IWKIN (NSU) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

# J

**J1A2B.** Backward complex discrete Fourier transform, FFT. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a2 **Usage:** CALL J1A2B(N, C, WSAVE) **Also see:** J1A21 J1A2F **Precision:** Single **Availability:** PC.

**J1A2F.** Forward complex discrete Fourier transform, FFT. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a2 **Usage:** CALL J1A2F(N, C, WSAVE) **Also see:** J1A21 J1A2B **Precision:** Single **Availability:** PC.

**J1A3B.** Computes a real periodic sequence from its Fourier coefficients. Inverse FFT. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a1 **Usage:** CALL J1A3B(N, R, WSAVE) **Also see:** J1A31 J1A3F **Precision:** Single **Availability:** PC.

**J1A3C.** Discrete Fourier cosine transform of an even sequence X. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a3 **Usage:** CALL J1A3C(N, X, WSAVE) **Also see:** J1A32 **Precision:** Single **Availability:** PC.

**J1A3D.** Backward fast Fourier transform (FFT) of quarter wave data. That is, computes a sequence from its representation in terms of a sine series with odd wave numbers. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a3 **Usage:** CALL J1A3D(N, X, WSAVE) **Also see:** J1A34 J1A3E **Precision:** Single **Availability:** PC.

**J1A3E.** Fast Fourier transform (FFT) of quarter wave data. That is, computes the coefficients in a sine series representation with only odd wave numbers. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a3 **Usage:** CALL J1A3E(N, X, WSAVE) **Also see:** J1A34 J1A3D **Precision:** Single **Availability:** PC.

**J1A3F.** Fast Fourier transform (FFT) of a real periodic sequence. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a1 **Usage:** CALL J1A3F(N, R, WSAVE) **Also see:** J1A31 J1A3B **Precision:** Single **Availability:** PC.

**J1A3G.** Fast Fourier transform (FFT) of quarter wave data. That is, computes the coefficients in a cosine series representation with only odd wave numbers. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a3 **Usage:** CALL J1A3G(N, X, WSAVE) **Also see:** J1A34 J1A3H **Precision:** Single **Availability:** PC.

**J1A3H.** Backward fast Fourier transform of quarter wave data. That is, computes a sequence from its representation in terms of a cosine series with odd wave numbers. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a3 **Usage:** CALL J1A3H(N, X, WSAVE) **Also see:** J1A34 J1A3G **Precision:** Single **Availability:** PC.

**J1A3S.** Discrete Fourier sine transform (FFT) of an odd sequence X. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a3 **Usage:** CALL J1A3S(N, X, WSAVE) **Also see:** J1A33 **Precision:** Single **Availability:** PC.

**J1AEB.** Computes a real periodic sequence from its Fourier coefficients. Inverse Fourier transform. This routine is a simplified version of J1A3B. It is not as fast as J1A3B since scaling and initialization are computed for each transform. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a1 **Usage:** CALL J1AEB(N, R, AZERO, A, B, WSAVE) **Also see:** J1AET **Precision:** Single **Availability:** PC.

**J1AET.** Computes the Fourier coefficients of a real periodic sequence. Fast Fourier transform. This routine is a simplified version of J1A3F. It is not as fast as J1A3F since scaling and initialization are computed for each transform. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1a1 **Usage:** CALL J1AET(N, R, AZERO, A, B, WSAVE) **Also see:** J1AEB **Precision:** Single **Availability:** PC.

**J1B2T.** Two dimensional fast Fourier transform (FFT), forward or reverse, of a complex n-by-m matrix F. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J1b **Usage:** CALL J1B2T(F, N, M, LDF, W, FORWARD) **Precision:** Single **Availability:** PC.

**J3LD.** Computes an approximation to the inverse Laplace transform F(T) for the value of the independent variable equal to T. The Durbin formula and the epsilon algorithm are used. The (hopeful) accuracy claim is $|F(T)-RESULT| \leq \max(EPSABS, EPSREL \times |F(T)|)$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** J3 **Usage:** CALL J3LD(*9 parameters*) **Precision:** Double **Availability:** PC.

**JACKNIFE PLOT.** Generate a jacknife plot for a given statistic, where the statistic is selected from among 29 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics or is user defined. The generated jacknife samples are available for further analysis. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1b1, L4b1b **Usage:** JACKNIFE <STATISTIC> PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**JCGRC.** Solve a real symmetric definite linear system using the Jacobi-preconditioned conjugate gradient method with reverse communication. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b4 **Usage:** CALL JCGRC (*9 parameters*) **Precision:** Single (Double: DJCGRC) **Availability:** 855NOS, 855VE, 205, 840NOS.

JOIN. Uses a general joining algorithm to form and output a tree of clusters of cases. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a1 **Usage:** CALL JOIN(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

JOIN. Merges constants and/or vectors into vectors. *Command in MINITAB proprietary interactive system.* **Classes:** L2d **Usage:** JOIN E to the bottom of E [to the bottom of E, ..., to E] put into C **Precision:** Single **Availability:** 855NOS.

JOIN2. Joins data values in a case-by-variable matrix into blocks until all within-block variances are greater than a user-specified threshold. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14b **Usage:** CALL JOIN2(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

# K

**K1A2CD.** Solves the least squares problem consisting of the linear constraints CX=Y and the least squares equations EX=F. Vectors X and Y are both unknowns, both of which may have user-specified bounds on each component. The user can specify equality and inequality constraints as well as simple bounds on the solution components. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D9b1 **Usage:** CALL K1A2CD(*15 parameters*) **Precision:** Double **Availability:** PC.

**K1A2D.** Solves EX=F in the least squares sense with bounds on selected X values. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** D9b1 **Usage:** CALL K1A2D(*13 parameters*) **Precision:** Double **Availability:** PC.

**KALMN.** Perform Kalman filtering and evaluate the likelihood function for the state-space model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2e, L10c **Usage:** CALL KALMN (*23 parameters*) **Precision:** Single (Double: DKALMN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KAPMR.** Compute Kaplan-Meier estimates of survival probabilities in stratified samples. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L15 **Usage:** CALL KAPMR (*12 parameters*) **Precision:** Single (Double: DKAPMR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KENDL.** Compute and test Kendall's rank correlation coefficient. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b1b **Usage:** CALL KENDL (*6 parameters*) **Precision:** Single (Double: DKENDL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KENDP.** Compute the frequency distribution of the total score in Kendall's rank correlation coefficient. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L5a1k **Usage:** CALL KENDP (NOBS, K, FREQ, PROB) **Precision:** Single (Double: DKENDP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KMEAN.** Perform a K-means (centroid) cluster analysis. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L14a1b **Usage:** CALL KMEAN (*17 parameters*) **Precision:** Single (Double: DKMEAN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KRSKL.** Perform a Kruskal-Wallis test for identical population medians. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4c1b **Usage:** CALL KRSKL (NGROUP, NI, Y, FUZZ, STAT) **Precision:** Single (Double: DKRSKL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KRUSKAL-WALLIS.** Perform Kruskal-Wallis test, based on ranks, of the null hypothesis that there is no difference among K population locations against the alternative of at least one difference. (This is a K-sample generalization of the Mann-Whitney-Wilcoxon test and is a nonparametric alternative to one-way ANOVA.) *Command in MINITAB proprietary interactive system.* **Classes:** L4c1b, L7a2 **Usage:** KRUSkal-Wallis test for data in C, subscripts in C **Precision:** Single **Availability:** 855NOS.

**KSONE.** Perform a Kolmogorov-Smirnov one-sample test for continuous distributions. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1c **Usage:** CALL KSONE (CDF, NOBS, X, PDIF, NMISS) **Precision:** Single (Double: DKSONE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KSTWO.** Perform a Kolmogorov-Smirnov two-sample test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b1c **Usage:** CALL KSTWO (*7 parameters*) **Precision:** Single (Double: DKSTWO) **Availability:** 855NOS, 855VE, 205, 840NOS.

**KTRND.** Perform K-sample trends test against ordered alternatives. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4c1b **Usage:** CALL KTRND (NGROUP, NI, X, STAT) **Precision:** Single (Double: DKTRND) **Availability:** 855NOS, 855VE, 205, 840NOS.

# L

**L1A1C.** Computes the mean and second moment about the mean for the sample contained in x. On option, the third and fourth moments can be computed. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L1a1 **Usage:** CALL L1A1C(X, N, IOPT, XMOM) **Precision:** Single **Availability:** PC.

**L1E1B.** Computes the mean, variance estimates, and correlation coefficients for the n observations on the m variables stored in matrix A. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L1c1 **Usage:** CALL L1E1B(A, LDA, N, M, VM, VV, C, KE) **Precision:** Single **Availability:** PC.

**L2SFF.** Obtains a weighted least square expansion of a known function in terms of B-splines of order K, at given mesh points. *Proprietary Fortran subroutine in PORT.* **Classes:** K1a1a1 **Usage:** CALL L2SFF (FW, K, T, NT, A) **Also see:** SPLNE SPLND SPLNI SPLN1 SPLN2 EEBSF EEBSI EESFF EESFI **Precision:** Single (Double: DL2SFF) **Availability:** 855NOS, 205.

**L2SFH.** Obtains a weighted least square expansion of a known function in and its derivatives in terms of B-splines of order K at given mesh points. *Proprietary Fortran subroutine in PORT.* **Classes:** K1a1a1 **Usage:** CALL L2SFH (FW, MD, K, T, NT, A) **Also see:** SPLNE SPLND SPLNI SPLN1 SPLN2 **Precision:** Single (Double: DL2SFH) **Availability:** 855NOS, 205.

**L3C3.** Plots tabled values of up to five functions. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** Q **Usage:** CALL L3C3(*13 parameters*) **Precision:** Single **Availability:** PC.

**L4A1BU.** Tests for uniformity of deviates when samples are small. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1a4u **Usage:** CALL L4A1BU(N, R, P, PR, KE) **Precision:** Single **Availability:** PC.

**L4A1CC.** Performs a chi-squared goodness of fit test that given data are from a hypothesized density function. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1c **Usage:** CALL L4A1CC(*10 parameters*) **Precision:** Single **Availability:** PC.

**L4A1CD.** Performs a chi-squared goodness of fit test that given data fall into categories in a hypothesized ratio. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1c **Usage:** CALL L4A1CD(*8 parameters*) **Precision:** Single **Availability:** PC.

**L4A1D.** Computes the first four moments about the mean for the sample contained in x, and performs a goodness of fit test for uniformity. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1a4u **Usage:** CALL L4A1D(X, N, K, XMOM, FREQ, KE) **Precision:** Single **Availability:** PC.

**L4A1DP.** Performs the pairs or serial test for uniformity of random deviates by tallying counts of coordinate occurrences and then performing a chi-squared test on the tally matrix. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1d **Usage:** CALL L4A1DP(*11 parameters*) **Precision:** Single **Availability:** PC.

**L4A1DT.** Performs the triplets test for uniformity of random deviates by tallying counts of coordinate occurrences and then performing a chi-squared test on the tally matrix. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1d **Usage:** CALL L4A1DT(R, N, T, LDT, M, C, S, P, D, KE) **Precision:** Single **Availability:** PC.

**L4A1M.** Uses the Mann-Wald criterion to determine the number of categories, or the sample size, for performance of a subsequent chi-squared test. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1c **Usage:** CALL L4A1M(OPTION, P, CN, KE) **Precision:** Single **Availability:** PC.

**L4ARD.** Approximates the expected values of normal order statistics. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L5a2n **Usage:** CALL L4ARD(S, N, KE) **Precision:** Single **Availability:** PC.

**L4B1.** Performs two sample tests on the mean of samples from normal populations, and estimates the means and variances of the samples. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4b1a4 **Usage:** CALL L4B1(X1, N1, X2, N2, IOPT, STAT, KE) **Precision:** Single **Availability:** PC.

**L4B1BR.** Performs the runs-up and runs-down test for uniformity of deviates. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L4a1d **Usage:** CALL L4B1BR(X, N, RUNUP, RUNDN, US, DS, KE) **Precision:** Single **Availability:** PC.

**L5A1.** Calculates the probability that a chi-squared distributed random variable (f degrees of freedom) is greater than a statistic x. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L5a1c **Usage:** CALL L5A1(X, F, KE) **Precision:** Single **Availability:** PC.

**L5A11.** Calculates (1) the integral of the normal probability density function from -infinity to x, (2) the integral of the normal probability density function from x to infinity, and (3) the density of the normal probability function at x. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L5a1n **Usage:** CALL L5A11(X, PL, PR, DEN) **Precision:** Single **Availability:** PC.

**L5A1B.** Incomplete beta cumulative probability distribution function. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L5a1b **Usage:** CALL L5A1B(X, P, Q, PROB, KE) **Precision:** Single **Availability:** PC.

**L5A1F.** Returns the probability that a random variable distributed as F with m and n degrees of freedom is greater than the statistic F. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L5a1f **Usage:** CALL L5A1F(M, N, F, Q, KE) **Precision:** Single **Availability:** PC.

**L5A1H.** Calculates cumulative hypergeometric probabilities (the probability of obtaining l marked items in a random sample of size k taken without replacement from a population containing m items, n of which are marked). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L5a1h **Usage:** CALL L5A1H(MODE, K, L, M, N, KE) **Precision:** Single **Availability:** PC.

**L5A1L.** Computes the natural logarithm of the normal distribution function (that is, the natural log of the integral from -infinity to x of the standard normal density). *Proprietary Fortran function in the Scientific Desk.* **Classes:** L5a1l **Usage:** R = L5A1L(X) **Precision:** Single **Availability:** PC.

**L5A1T.** Computes the integral from t to positive infinity of Student's t distribution. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L5a1t **Usage:** R = L5A1T(T, D, KE) **Precision:** Single **Availability:** PC.

**L5A2C.** Computes the inverse of the chi-squared distribution function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L5a2c **Usage:** R = L5A2C(Q, F, KE) **Precision:** Single **Availability:** PC.

**L5A2N.** Calculates the inverse of the cumulative normal distribution function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L5a2n **Usage:** R = L5A2N(P) **Precision:** Single **Availability:** PC.

**L6A12.** Generates one lognormal (e,s) deviate by transforming one normal(0, 1) deviate x by using y=exp(e+sx). *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a12 **Usage:** R = L6A12(X, E, S, KE) **Precision:** Single **Availability:** PC.

**L6A14F.** Generates one normal(0,1) random deviate by inverting the distribution function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a14 **Usage:** R = L6A14F(X) **Precision:** Single **Availability:** PC.

**L6A14S.** Generates n normal (0,1) single precision random numbers. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a14 **Usage:** CALL L6A14S(X, R, N, IOPT) **Precision:** Single **Availability:** PC.

**L6A14T.** Generates n normal(0,1) random deviates by inverting the distribution function. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a14 **Usage:** CALL L6A14T(X, R, N) **Precision:** Single **Availability:** PC.

**L6A16.** Generates one Poisson deviate. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a16 **Usage:** R = L6A16(X, ALAM, KE) **Precision:** Single **Availability:** PC.

**L6A2.** Generates one beta(p,q) deviate, using a rejection technique. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a2 **Usage:** R = L6A2(X, P, Q, KE) **Precision:** Single **Availability:** PC.

**L6A20.** Generates one deviate from a symmetric triangular distribution in (0,1). *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a20 **Usage:** R = L6A20(X) **Precision:** Single **Availability:** PC.

**L6A21F.** Generates one uniform single precision random number in the range (0,1). *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a21 **Usage:** R = L6A21F(X) **Precision:** Single **Availability:** PC.

**L6A21S.** Generates n shuffled uniform single precision random numbers in the range (0,1). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a21 **Usage:** CALL L6A21S(X, R, N, IOPT) **Precision:** Single **Availability:** PC.

**L6A21T.** Generates n uniform single precision random numbers in the range (0,1). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a21 **Usage:** CALL L6A21T(X, R, N) **Precision:** Single **Availability:** PC.

**L6A23.** Generates one Weibull deviate. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a23 **Usage:** R = L6A23(X, A, KE) **Precision:** Single **Availability:** PC.

**L6A24.** Generates two pseudo-random deviates from the error function density. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a5 **Usage:** CALL L6A24(X, R) **Precision:** Single **Availability:** PC.

**L6A25.** Generates one pseudo-random integer from the Pascal density function. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a16 **Usage:** CALL L6A25(X, A, IX) **Precision:** Single **Availability:** PC.

**L6A2B.** Generates one binomial(n,p) deviate. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a2 **Usage:** R = L6A2B(X, N, P, KE) **Precision:** Single **Availability:** PC.

**L6A3.** Generates one Cauchy random deviate. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a3 **Usage:** R = L6A3(X) **Precision:** Single **Availability:** PC.

**L6A3C.** Returns one chi-squared deviate (n degrees of freedom). *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a3 **Usage:** R = L6A3C(X, N, W, KE) **Precision:** Single **Availability:** PC.

**L6A5.** Generates one exponential deviate with mean and standard deviation u by inverting the distribution function. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a5 **Usage:** R = L6A5(X, U) **Precision:** Single **Availability:** PC.

**L6A7.** Generates one gamma deviate. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a7 **Usage:** R = L6A7(X, ALPHA, KE) **Precision:** Single **Availability:** PC.

**L6A76.** Generates one geometric deviate. *Proprietary Fortran function in the Scientific Desk.* **Classes:** L6a7 **Usage:** R = L6A76(X, P, KE) **Precision:** Single **Availability:** PC.

**L6A9.** Generates n uniform integer random numbers in the inclusive range (ia,ib). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6a21 **Usage:** CALL L6A9(X, IA, IB, IR, N) **Precision:** Single **Availability:** PC.

**L6B13.** Generates one multinomial (k-tuple) deviate. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6b13 **Usage:** CALL L6B13(X, N, K, P, WK, GMUL, KE) **Precision:** Single **Availability:** PC.

**L6B14.** Generates k-plet multinormal deviates with mean 0 and covariance matrix cov. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L6b14 **Usage:** CALL L6B14(X, IOPT, N, K, COV, R, NRR, W, KE) **Precision:** Single **Availability:** PC.

**L7A11.** Computes averages, within treatment standard deviations, and the associated analysis of variance table for a one-way classification with an equal or unequal number of observations per treatment. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L7a1 **Usage:** CALL L7A11(*10 parameters*) **Precision:** Single **Availability:** PC.

**L8A1A.** Performs simple linear regression for the model Y=A+BX. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L8a1a1 **Usage:** CALL L8A1A(Y, X, N, RES, STAT, KE) **Precision:** Single **Availability:** PC.

**L8A2BD.** Performs weighted polynomial regression using orthogonal polynomials. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L8b1b2 **Usage:** CALL L8A2BD(*17 parameters*) **Also see:** L8A2CD **Precision:** Double **Availability:** PC.

**L8CP.** Estimates (and to estimate the best subset of) the parameters which fit a linear regression under a least absolute value criterion. This program utilizes the simplex method of linear programming within a branch-and-bound algorithm to solve the subset problem. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L8c3 **Usage:** CALL L8CP(*16 parameters*) **Precision:** Single **Availability:** PC.

**L8CS.** Calculates simple linear regression least absolute value estimates. The least absolute curve fitting problem is to find A and B such that the sum of the absolute values of $Y_i - A - X_iB$ is minimized, (i=1,...,n). *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** L8a2 **Usage:** CALL L8CS(*10 parameters*) **Precision:** Single (Double: L8CSD) **Availability:** PC.

**L8CSD.** *Double precision version of L8CS.*

**L8G1A.** Determines an $r$ for the equation $Y_i = b_0 + b_1 X_i^R$ so that $Y_i = b_0 + b_1 Z_i$, where $Z_i = X_i^r$ plots as a straight line. The iteration scheme used for R is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $r$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** K1b1a1, L2a, L8h **Usage:** CALL L8G1A(*13 parameters*) **Precision:** Single **Availability:** PC.

**L8G1E.** Determines an $a$ for the equation $Y_i = b_0 + b_1 a^{X_i}$ so that $Y_i = b_0 + b_1 Z_i$, where $Z_i = a^{X_i}$ plots as a straight line. The iteration scheme used for R is based on a first-order Taylor series expansion. Estimates of $b_0$ and $b_1$ are obtained by usual linear least squares techniques for the selected $a$. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** K1b1a1, L2a, L8h **Usage:** CALL L8G1E(*13 parameters*) **Precision:** Single **Availability:** PC.

**LAG.** Computes lagged observations in a time series. *Command in MINITAB proprietary interactive system.* **Classes:** L2a, L10a1a **Usage:** LAG [K] data in C, put into C **Precision:** Single **Availability:** 855NOS.

**LAG PLOT.** Plots a lag plot (for a user-specified lag) for equi-spaced univariate or bivariate time series data. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a7, L3b3c **Usage:** LAG <POSITIVE INTEGER> PLOT <RESPONSE VARIABLE> [<RESPONSE VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LAMBDA PPCC PLOT.** Generates a probability plot correlation coefficient plot for Tukey lambda distribution (plot of probability plot correlation coefficient vs. the tail parameter $\lambda$ for $\lambda$ ranging from -2 to 2 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3l **Usage:** TUKEY [LAMBDA] PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LAMBDA PROBABILITY PLOT.** Generates a probability plot for the Tukey lambda distribution with tail length parameter $\lambda$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2l **Usage:** [TUKEY] LAMBDA PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LAMCDF.** Computes the cumulative distribution function value for the (Tukey) lambda distribution with tail length parameter ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1l **Usage:** CALL LAMCDF(X, ALAMBA, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LAMPDF.** Computes the probability density function value for the (Tukey) lambda distribution with tail length parameter ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1l **Usage:** CALL LAMPDF(X, ALAMBA, PDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LAMPLT.** Generates a (Tukey) lambda distribution probability plot with tail length parameter ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2l **Usage:** CALL LAMPLT(X, N, ALAMBA) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LAMPPF.** Computes the percent point function value for the (Tukey) lambda distribution with tail length parameter ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L5a2l **Usage:** CALL LAMPPF(P, ALAMBA, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LAMRAN.** Generates a random sample of size N from the (Tukey) lambda distribution with tail length parameter ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L6a12 **Usage:** CALL LAMRAN(N, ALAMBA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LAMSF.** Computes the sparsity function value for the (Tukey) lambda distribution with tail length parameter

ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L5a2l **Usage:** CALL LAMSF(P, ALAMBA, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LCHRG.** Compute the Cholesky decomposition of a symmetric positive semidefinite matrix with optional column pivoting. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LCHRG (*7 parameters*) **Precision:** Single (Double: DLCHRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LDNCH.** Downdate the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is removed. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D7b **Usage:** CALL LDNCH (*8 parameters*) **Precision:** Single (Double: DLDNCH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LEAVE.** Restores prior error recovery mode and resets the stack for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3c **Usage:** CALL LEAVE **Precision:** Single **Availability:** 855NOS, 205.

**LEQ1PB.** Linear equation solution – positive definite symmetric band matrix – band symmetric storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LEQ1PB (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LEQ1S.** Linear equation solution – indefinite matrix – symmetric storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1a **Usage:** CALL LEQ1S (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LEQ2C.** Linear equation solution – complex matrix high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2c1 **Usage:** CALL LEQ2C (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQ2PB.** Linear equation solution – positive definite band symmetric matrix – band symmetric storage mode – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LEQ2PB (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQ2S.** Linear equation solution – indefinite matrix – symmetric storage mode – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1a **Usage:** CALL LEQ2S (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LEQIF.** Linear equation solution – full matrices (virtual memory version). *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LEQIF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQOF.** Linear equation solution – full matrices (out-of-core version). *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LEQOF(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LEQT1B.** Linear equation solution – band storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a2 **Usage:** CALL LEQT1B (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQT1C.** Matrix decomposition, linear equation solution – space economizer solution complex matrices. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2c1 **Usage:** CALL LEQT1C (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQT1F.** Linear equation solution – full storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LEQT1F (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQT1P.** Linear equation solution – positive definite matrix – symmetric storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LEQT1P (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LEQT2B.** Linear equation solution – band storage mode – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a2 **Usage:** CALL LEQT2B (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQT2F.** Linear equation solution – full storage mode – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LEQT2F (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LEQT2P.** Linear equations solution – positive definite matrix – symmetric storage mode – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LEQT2P (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LET.** Carries out general transformations of variables and assigns values to variables or parameters for function evaluation. Options: statistics (mean, median, autocorrelation, etc.), mathematical operations (sum, root, derivative, integral, etc.), random number generation (from 24 distributions and families), sequencing, patterning. *Command(s) in DATAPLOT interactive system.* **Classes:** L2a, L2e **Usage:** LET <VARIABLE NAME OR PARAMETER> = <VALUE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LET FUNCTION.** Creates functions. Options: intrinsic functions, concatenation or build-up of functions, definition before or after the parameters and variables used in them are created. *Command(s) in DATAPLOT interactive system.* **Classes:** L2a **Usage:** LET FUNCTION <FUNCTION NAME> = <FUNCTION EXPRESSION> **Precision:**

Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LETREE.** Uses a leader clustering algorithm to construct a tree whose levels are determined by user-defined thresholds. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a1 **Usage:** CALL LETREE(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LETTR.** Produce a letter value summary. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L1a1 **Usage:** CALL LETTR (NOBS, X, NUM, SUMRY, NMISS) **Precision:** Single (Double: DLETTR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCCB.** Compute the LU factorization of a complex matrix in band storage mode and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c2 **Usage:** CALL LFCCB (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCCG.** Compute the LU factorization of a complex general matrix and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LFCCG (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCCT.** Estimate the condition number of a complex triangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c3 **Usage:** CALL LFCCT (N, A, LDA, IPATH, RCOND) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCDH.** Compute the $R^HR$ factorization of a complex Hermitian positive definite matrix and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1b **Usage:** CALL LFCDH (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCDS.** Compute the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LFCDS (*6 parameters*) **Precision:** Single (Double: DLFCDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCHF.** Compute the $UDU^H$ factorization of a complex Hermitian matrix and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LFCHF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCQH.** Compute the $R^HR$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d2 **Usage:** CALL LFCQH (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCQS.** Compute the Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LFCQS (*7 parameters*) **Precision:** Single (Double: DLFCQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCRB.** Compute the LU factorization of a real matrix in band storage mode and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a2 **Usage:** CALL LFCRB (*9 parameters*) **Precision:** Single (Double: DLFCRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCRG.** Compute the LU factorization of a real general matrix and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LFCRG (*7 parameters*) **Precision:** Single (Double: DLFCRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCRT.** Estimate the condition number of a real triangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a3 **Usage:** CALL LFCRT (N, A, LDA, IPATH, RCOND) **Precision:** Single (Double: DLFCRT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFCSF.** Compute the $UDU^T$ factorization of a real symmetric matrix and estimate its $L_1$ condition number. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1a **Usage:** CALL LFCSF (*7 parameters*) **Precision:** Single (Double: DLFCSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDCB.** Compute the determinant of a complex matrix given the LU factorization of the matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3c2 **Usage:** CALL LFDCB (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDCG.** Compute the determinant of a complex general matrix given the LU factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3a1 **Usage:** CALL LFDCG (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDCT.** Compute the determinant of a complex triangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3c3 **Usage:** CALL LFDCT (N, A, LDA, DET1, DET2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDDH.** Compute the determinant of a complex Hermitian positive definite matrix given the $R^TR$ Cholesky factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3d1b **Usage:** CALL LFDDH (N, FAC, LDFAC, DET1, DET2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDDS.** Compute the determinant of a real symmetric positive definite matrix given the $R^TR$ Cholesky factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3a1 **Usage:** CALL LFDDS (N, FAC, LDFAC, DET1, DET2) **Precision:** Single (Double: DLFDDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDHF.** Compute the determinant of a complex Hermitian matrix given the $UDU^H$ factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LFDHF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDQH.** Compute the determinant of a complex Hermitian positive definite matrix given the $R^TR$ Cholesky factorization in band Hermitian storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3d2 **Usage:** CALL LFDQH (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDQS.** Compute the determinant of a real symmetric positive definite matrix in band symmetric storage mode given its Cholesky factorization. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LFDQS (*6 parameters*) **Precision:** Single (Double: DLFDQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDRB.** Compute the determinant of a real matrix in band storage mode given the LU factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3a1 **Usage:** CALL LFDRB (*8 parameters*) **Precision:** Single (Double: DLFDRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDRG.** Compute the determinant of a real general matrix given the LU factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D3a1 **Usage:** CALL LFDRG (*6 parameters*) **Precision:** Single (Double: DLFDRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDRT.** Compute the determinant of a real triangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3a3 **Usage:** CALL LFDRT (N, A, LDA, DET1, DET2) **Precision:** Single (Double: DLFDRT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFDSF.** Compute the determinant of a real symmetric matrix given the $UDU^T$ factorization of the matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D3a1 **Usage:** CALL LFDSF (*6 parameters*) **Precision:** Single (Double: DLFDSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFICB.** Use iterative refinement to improve the solution of a complex system of linear equations in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c2 **Usage:** CALL LFICB (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFICG.** Use iterative refinement to improve the solution of a complex general system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LFICG (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIDH.** Use iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1b **Usage:** CALL LFIDH (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIDS.** Use iterative refinement to improve the solution of a real symmetric positive definite system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LFIDS (*8 parameters*) **Precision:** Single (Double: DLFIDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIHF.** Use iterative refinement to improve the solution of a complex Hermitian system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LFIHF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIQH.** Use iterative refinement to improve the solution of a complex Hermitian positive definite system of linear equations in band Hermitian storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d2 **Usage:** CALL LFIQH (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIQS.** Use iterative refinement to improve the solution of a real symmetric positive definite system of linear equations in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LFIQS (*9 parameters*) **Precision:** Single (Double: DLFIQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIRB.** Use iterative refinement to improve the solution of a real system of linear equations in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a2 **Usage:** CALL LFIRB (*12 parameters*) **Precision:** Single (Double: DLFIRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFIRG.** Use iterative refinement to improve the solution of a real general system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LFIRG (*10 parameters*) **Precision:** Single (Double: DLFIRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFISF.** Use iterative refinement to improve the solution of a real symmetric system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1a **Usage:** CALL LFISF (*9 parameters*) **Precision:** Single (Double: DLFISF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSCB.** Solve a complex system of linear equations given the LU factorization of the coefficient matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c2 **Usage:** CALL LFSCB (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSCG.** Solve a complex general system of linear equations given the LU factorization of the coefficient matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LFSCG (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSDH.** Solve a complex Hermitian positive definite system of linear equations given the $R^HR$ factorization of the coefficient matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1b **Usage:** CALL LFSDH (N, FAC, LDFAC, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSDS.** Solve a real symmetric positive definite system of linear equations given the $R^TR$ Cholesky factorization of the coefficient matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LFSDS (N, FAC, LDFAC, B, X) **Precision:** Single (Double: DLFSDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSHF.** Solve a complex Hermitian system of linear equations given the $UDU^H$ factorization of the coefficient matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LFSHF (N, FAC, LDFAC, IPVT, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSQH.** Solve a complex Hermitian positive definite system of linear equations given the factorization of the coefficient matrix in band Hermitian storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d2 **Usage:** CALL LFSQH (N, FAC, LDFAC, NCODA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSQS.** Solve a real symmetric positive definite system of linear equations given the factorization of the coefficient matrix in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LFSQS (N, FAC, LDFAC, NCODA, B, X) **Precision:** Single (Double: DLFSQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSRB.** Solve a real system of linear equations given the LU factorization of the coefficient matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a2 **Usage:** CALL LFSRB (*9 parameters*) **Precision:** Single (Double: DLFSRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSRG.** Solve a real general system of linear equations given the LU factorization of the coefficient matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LFSRG (*7 parameters*) **Precision:** Single (Double: DLFSRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFSSF.** Solve a real symmetric system of linear equations given the $UDU^T$ factorization of the coefficient matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1a **Usage:** CALL LFSSF (N, FAC, LDFAC, IPVT, B, X) **Precision:** Single (Double: DLFSSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTCB.** Compute the LU factorization of a complex matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c2 **Usage:** CALL LFTCB (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTCG.** Compute the LU factorization of a complex general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LFTCG (N, A, LDA, FAC, LDFAC, IPVT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTDH.** Compute the $R^HR$ factorization of a complex Hermitian positive definite matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1b **Usage:** CALL LFTDH (N, A, LDA, FAC, LDFAC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTDS.** Compute the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LFTDS (N, A, LDA, FAC, LDFAC) **Precision:** Single (Double: DLFTDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTHF.** Compute the $UDU^H$ factorization of a complex Hermitian matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LFTHF (N, A, LDA, FAC, LDFAC, IPVT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTQH.** Compute the $R^HR$ factorization of a complex Hermitian positive definite matrix in band Hermitian storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d2 **Usage:** CALL LFTQH (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTQS.** Compute the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LFTQS (*6 parameters*) **Precision:** Single (Double: DLFTQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTRB.** Compute the LU factorization of a real matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a2 **Usage:** CALL LFTRB (*8 parameters*) **Precision:** Single (Double: DLFTRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTRG.** Compute the LU factorization of a real general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LFTRG (N, A, LDA, FAC, LDFAC, IPVT) **Precision:** Single (Double: DLFTRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LFTSF.** Compute the $UDU^T$ factorization of a real symmetric matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1a **Usage:** CALL LFTSF (N, A, LDA, FAC, LDFAC, IPVT) **Precision:** Single (Double: DLFTSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LGINF.** Generalized inverse of a real matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D9c **Usage:** CALL LGINF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LGNCDF.** Computes the cumulative distribution function value for the lognormal distribution with mean $\sqrt{e}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a1l **Usage:** CALL LGNCDF(X, CDF) **Precision:** Single **Availability:**

855NOS, 840NOS.

**LGNPLT.** Generates a lognormal probability plot with mean $\sqrt{e}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2l **Usage:** CALL LGNPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LGNPPF.** Computes the percent point function value for the lognormal distribution with mean $\sqrt{e}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a2l **Usage:** CALL LGNPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LGNRAN.** Generates a random sample of size N from the lognormal distribution with mean $\sqrt{e}$. *Fortran subroutine in DATAPAC.* **Classes:** L6a12 **Usage:** CALL LGNRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LIFEREG.** Fits parametric models to failure-time data that may be right censored. Models include exponential, Weibull, log normal, and log logistic. Parameters are estimated by maximum likelihood using a Newton-Raphson algorithm. Independent variable may be continuous or discrete. *Proprietary stand-alone program using SAS command language.* **Classes:** L15 **Precision:** Single **Availability:** CAMVAX.

**LIFETEST.** Computes nonparametric estimates of the survival distribution (by the product limit method or the life table method) and computes rank tests for association of the response variable with covariates for stratified data that may be right censored. Options: tests homogeneity between strata, missing values, printer plots. *Proprietary stand-alone program using SAS command language.* **Classes:** L15 **Precision:** Single **Availability:** CAMVAX.

**LIN1PB.** Inversion of a matrix – positive definite band symmetric matrix – band symmetric storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LIN1PB (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LIN2PB.** Inversion of matrix – positive definite band symmetric matrix – band symmetric storage mode – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LIN2PB (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINCG.** Compute the inverse of a complex general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LINCG (N, A, LDA, AINV, LDAINV) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINCT.** Compute the inverse of a complex triangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c3 **Usage:** CALL LINCT (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINDS.** Compute the inverse of a real symmetric positive definite matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LINDS (N, A, LDA, AINV, LDAINV) **Precision:** Single (Double: DLINDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINE.** Produces a line profile of the variables for each case. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e4 **Usage:** CALL LINE(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LINEQ.** Solves a real system of linear equations, AX=B, where B is allowed to be a matrix or a vector. *Proprietary Fortran subroutine in PORT.* **Classes:** D2a1 **Usage:** CALL LINEQ (N, A, B, NB, X) **Precision:** Single (Double: DLINEQ) **Availability:** 855NOS, 205.

**LINFS.** Solves the simple model, $Y=\beta_1+\beta_2X$, under the Chebychev norm criterion. *Portable Fortran subroutine in CMLIB (SLRPACK sublibrary).* **Classes:** L8a2 **Usage:** CALL LINFS(*11 parameters*) **Precision:** Single (Double: DLINFS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**LINK.** Constructs and prints tree of clusters obtained by adding cases in succession to minimize the sum of the linking distances. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a1 **Usage:** CALL LINK(MM, M, N, A, P, IWORK, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LINRG.** Compute the inverse of a real general matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LINRG (N, A, LDA, AINV, LDAINV) **Precision:** Single (Double: DLINRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINRT.** Compute the inverse of a real triangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a3 **Usage:** CALL LINRT (*6 parameters*) **Precision:** Single (Double: DLINRT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINV1F.** Inversion of a matrix – full storage mode space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LINV1F (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LINV1P.** Inversion of matrix – positive definite symmetric storage mode – space economizer solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LINV1P (A, N, AINV, IDGT, D1, D2, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINV2F.** Inversion of a matrix – full storage mode high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LINV2F (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LINV2P.** Inversion of a matrix – positive definite symmetric storage mode – high accuracy solution. *Proprietary*

*Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LINV2P (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LINV3F.** In place inverse, equation solution, and/or determinant evaluation – full storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1, D3a1 **Usage:** CALL LINV3F (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LINV3P.** In place inverse, equation solution, positive definite matrix – symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LINV3P (A, B, IJOB, N, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LLBQF.** Solution of linear least squares problems – high accuracy solution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D9b1 **Usage:** CALL LLBQF (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LLS.** Compute and print a four-part unweighted linear least squares analysis with user-specified model (design matrix); return residuals. *Portable Fortran subroutine in STARPAC.* **Classes:** L8a1a1, L8c1b1 **Usage:** CALL LLS (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSIA.** Computes least squares solution to AX=B with A an m-by-n matrix with m≥n. Flexible version of SGLSS. *Portable Fortran subroutine in CMLIB (SGLSS sublibrary).* **Classes:** D9a1 **Usage:** CALL LLSIA(*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**LLSP.** Compute and print a four-part unweighted linear least squares analysis with polynomial model (design matrix); return residuals. *Portable Fortran subroutine in STARPAC.* **Classes:** L8b1b1 **Usage:** CALL LLSP (Y, X, N, NDEG, RES, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSPS.** Compute and optionally print a four-part unweighted linear least squares analysis with polynomial model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters. *Portable Fortran subroutine in STARPAC.* **Classes:** L8b1b1 **Usage:** CALL LLSPS (*16 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSPW.** Compute and print a four-part weighted linear least squares analysis with polynomial model (design matrix); return residuals. *Portable Fortran subroutine in STARPAC.* **Classes:** L8b1b1 **Usage:** CALL LLSPW (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSPWS.** Compute and optionally print a four-part weighted linear least squares analysis with polynomial model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters. *Portable Fortran subroutine in STARPAC.* **Classes:** L8b1b1 **Usage:** CALL LLSPWS (*17 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSQF.** Solution of linear least squares problems. *Proprietary Fortran subroutine in IMSL.* **Classes:** D9b1 **Usage:** CALL LLSQF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LLSS.** Compute and optionally print a four-part unweighted linear least squares analysis with user-specified model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters. *Portable Fortran subroutine in STARPAC.* **Classes:** L8a1a1, L8c1b1 **Usage:** CALL LLSS (*15 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSW.** Compute and print a four-part weighted linear least squares analysis with user-specified model (design matrix); return residuals. *Portable Fortran subroutine in STARPAC.* **Classes:** L8a1a2, L8c1b1 **Usage:** CALL LLSW (*8 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LLSWS.** Compute and optionally print a four-part weighted linear least squares analysis with user-specified model (design matrix); return residuals, parameter estimates, residual standard deviation, predicted values, standard deviations of the predicted values, standardized residuals, and variance-covariance matrix of parameters. *Portable Fortran subroutine in STARPAC.* **Classes:** L8a1a2, L8c1b1 **Usage:** CALL LLSWS (*16 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LOC.** Computes 4 location estimates (midrange, mean, midmean, and median) of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL LOC(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOCF.** To obtain the address of an array element. *Fortran/meta function in MAGEV.* **Classes:** R **Usage:** I =

LOCF(A) **Precision:** Single **Availability:** 205.

**LOGCDF.** Computes the cumulative distribution function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a1l **Usage:** CALL LOGCDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOGISTIC PROBABILITY PLOT.** Generates a probability plot for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2l **Usage:** LOGISTIC PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LOGLINEAR.** Models cell frequencies using multinomial response model and produces maximum likelihood estimates of parameters by means of the Newton-Raphson algorithm. Options: cell weights, plots, linear combinations, contrasts, algorithm tuning parameters, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L9c **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**LOGNORMAL PROBABILITY PLOT.** Generates a probability plot for the log normal distribution with mean $\sqrt{e}$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2l **Usage:** LOGNORMAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**LOGPDF.** Computes the probability density function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a1l **Usage:** CALL LOGPDF(X, PDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOGPLT.** Generates a logistic probability plot with mean 0 and standard deviation $\pi/\sqrt{3}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2l **Usage:** CALL LOGPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOGPPF.** Computes the percent point function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a2l **Usage:** CALL LOGPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOGRAN.** Generates a random sample of size N from the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$. *Fortran subroutine in DATAPAC.* **Classes:** L6a12 **Usage:** CALL LOGRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOGSF.** Computes the sparsity function value for the logistic distribution with mean 0 and standard deviation $\pi/\sqrt{3}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a2l **Usage:** CALL LOGSF(P, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**LOPASS.** Filter series with symmetric linear low-pass filter; return filtered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL LOPASS (Y, N, FC, K, HLP, YF, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LOTPS.** Passes smooth function through two-dimensional scattered data and returns an array of interpolated values on user specified grid. Based on local thin-plate splines. *Portable Fortran subroutine in CMLIB (LOTPS sublibrary).* **Classes:** E2b **Usage:** CALL LOTPS(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**LOWESS SMOOTH.** Perform a locally weighted least squares analysis of a two-dimensional data set. Output includes predicted and residual values. The user can specify the smoothing interval. *Command(s) in DATAPLOT interactive system.* **Classes:** L8a3, L8h **Usage:** LOWESS SMOOTH <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**LPCOEF.** Compute symmetric linear low-pass filter coefficients; return filter coefficients. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL LPCOEF (FC, K, HLP) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**LPDP.** Solves least projected distance problem. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** D9b1 **Usage:** CALL LPDP(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**LPLOT.** Prints a letter plot with symbols corresponding to numerical "tag" values. Scale specification is optional. *Command in MINITAB proprietary interactive system.* **Classes:** L3b3b **Usage:** LPLOt C [from K to K] vs C [from K to K] using tags in C **Precision:** Single **Availability:** 855NOS.

**LQERR.** Accumulate the orthogonal matrix Q from its factored form given the QR factorization of a rectangular matrix A. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D5 **Usage:** CALL LQERR (*7 parameters*) **Precision:** Single (Double: DLQERR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LQRRR.** Compute the QR decomposition using Householder transformations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D5 **Usage:** CALL LQRRR (*10 parameters*) **Precision:** Single (Double: DLQRRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LQRSL.** Compute the coordinate transformation, projection, and solution for the least squares problem. *Proprietary*

*Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D9a1 **Usage:** CALL LQRSL (*12 parameters*) **Precision:** Single (Double: DLQRSL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSACB.** Solve a complex system of linear equations in band storage mode with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c2 **Usage:** CALL LSACB (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSACG.** Solve a complex general system of linear equations with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LSACG (N, A, LDA, B, IPATH, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSADH.** Solve a Hermitian positive definite system of linear equations with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1b **Usage:** CALL LSADH (N, A, LDA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSADS.** Solve a real symmetric positive definite system of linear equations with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LSADS (N, A, LDA, B, X) **Precision:** Single (Double: DLSADS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSAHF.** Solve a complex Hermitian system of linear equations with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LSAHF (N, A, LDA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSAQH.** Solve a complex Hermitian positive definite system of linear equations in band Hermitian storage mode with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d2 **Usage:** CALL LSAQH (N, A, LDA, NCODA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSAQS.** Solve a real symmetric positive definite system of linear equations in band symmetric storage mode with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LSAQS (N, A, LDA, NCODA, B, X) **Precision:** Single (Double: DLSAQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSARB.** Solve a real system of linear equations in band storage mode with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a2 **Usage:** CALL LSARB (*8 parameters*) **Precision:** Single (Double: DLSARB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSARG.** Solve a real general system of linear equations with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LSARG (N, A, LDA, B, IPATH, X) **Precision:** Single (Double: DLSARG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSASF.** Solve a real symmetric system of linear equations with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1a **Usage:** CALL LSASF (N, A, LDA, B, X) **Precision:** Single (Double: DLSASF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSBRR.** Solve a linear least squares problem with iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D9a1 **Usage:** CALL LSBRR (*9 parameters*) **Precision:** Single (Double: DLSBRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSEI.** Solves linearly constrained least squares problem with equality and inequality constraints. Covariance matrix optionally computed. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** D9b1 **Usage:** CALL LSEI(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**LSGRR.** Compute the generalized inverse of a real matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D9c **Usage:** CALL LSGRR (*8 parameters*) **Precision:** Single (Double: DLSGRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLCB.** Solve a complex system of linear equations in band storage mode without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c2 **Usage:** CALL LSLCB (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLCC.** Solve a complex circulant linear system. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a **Usage:** CALL LSLCC (N, A, B, IPATH, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLCG.** Solve a complex general system of linear equations without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c1 **Usage:** CALL LSLCG (N, A, LDA, B, IPATH, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLCT.** Solve a complex triangular system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2c3 **Usage:** CALL LSLCT (N, A, LDA, B, IPATH, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLDH.** Solve a complex Hermitian positive definite system of linear equations without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1b **Usage:** CALL LSLDH (N, A, LDA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLDS.** Solve a real symmetric positive definite system of linear equations without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1b **Usage:** CALL LSLDS (N, A, LDA, B, X) **Precision:** Single (Double: DLSLDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLHF.** Solve a complex Hermitian system of linear equations without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d1a **Usage:** CALL LSLHF (N, A, LDA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLQH.** Solve a complex Hermitian positive definite system of linear equations in band Hermitian storage mode without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2d2 **Usage:** CALL LSLQH (N, A, LDA, NCODA, B, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLQS.** Solve a real symmetric positive definite system of linear equations in band symmetric storage mode without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b2 **Usage:** CALL LSLQS (N, A, LDA, NCODA, B, X) **Precision:** Single (Double: DLSLQS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLRB.** Solve a real system of linear equations in band storage mode without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a2 **Usage:** CALL LSLRB (*8 parameters*) **Precision:** Single (Double: DLSLRB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLRG.** Solve a real general system of linear equations without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D2a1 **Usage:** CALL LSLRG (N, A, LDA, B, IPATH, X) **Precision:** Single (Double: DLSLRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLRT.** Solve a real triangular system of linear equations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a3 **Usage:** CALL LSLRT (N, A, LDA, B, IPATH, X) **Precision:** Single (Double: DLSLRT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLSF.** Solve a real symmetric system of linear equations without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b1a **Usage:** CALL LSLSF (N, A, LDA, B, X) **Precision:** Single (Double: DLSLSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLTC.** Solve a complex Toeplitz linear system. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a **Usage:** CALL LSLTC (N, A, B, IPATH, X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSLTO.** Solve a real Toeplitz linear system. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2a **Usage:** CALL LSLTO (N, A, B, IPATH, X) **Precision:** Single (Double: DLSLTO) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSQRR.** Solve a linear least squares problem without iterative refinement. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D9a1 **Usage:** CALL LSQRR (*9 parameters*) **Precision:** Single (Double: DLSQRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSTSQ.** Finds the least squares solution of a system of linear equations, AX=B. B may be a matrix. *Proprietary Fortran subroutine in PORT.* **Classes:** D9a1 **Usage:** CALL LSTSQ (MDIM, NDIM, M, N, A, B, NB, X) **Precision:** Single (Double: DLSTSQ) **Availability:** 855NOS, 205.

**LSVCR.** Compute the singular value decomposition of a complex matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D6 **Usage:** CALL LSVCR (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSVDB.** Singular value decomposition of a bidiagonal matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D6 **Usage:** CALL LSVDB (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSVDF.** Singular value decomposition of a real matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D6 **Usage:** CALL LSVDF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LSVRR.** Compute the singular value decomposition of a real matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D6 **Usage:** CALL LSVRR (*12 parameters*) **Precision:** Single (Double: DLSVRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LUDAPB.** Decomposition of a positive definite band symmetric matrix – band symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LUDAPB (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUDATF.** LU decomposition by the Crout algorithm with optional accuracy test. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LUDATF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUDECP.** Decomposition of a positive definite matrix symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LUDECP (A, UL, N, D1, D2, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**LUELMF.** Elimination part of solution of Ax=b (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LUELMF (A, B, IPVT, N, IA, X) **Also see:** LUDATF **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUELMP.** Elimination part of the solution of Ax=b positive definite matrix – symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LUELMP (A, B, N, X) **Also see:** LUDECP **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUELPB.** Elimination part of solution of Ax=b positive definite band symmetric matrix-band symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LUELPB (UL, B, N, NC, IA, X) **Also**

see: LUDAPB **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUMB.** Given a basic mesh, this subdivides each interval uniformly for B-spline use. Multiplicities are allowed. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL LUMB (XB, NXB, N, K, X, NX) **Precision:** Single (Double: DLUMB) **Availability:** 855NOS, 205.

**LUMD.** Given a basic mesh, this subdivides each interval into the same number of uniformly spaced points. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL LUMD (XB, NXB, N, X, NX) **Precision:** Single (Double: DLUMD) **Availability:** 855NOS, 205.

**LUPCH.** Update the $R^TR$ Cholesky factorization of a real symmetric positive definite matrix after a rank-one matrix is added. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D7b **Usage:** CALL LUPCH (*8 parameters*) **Precision:** Single (Double: DLUPCH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LUPQR.** Compute an updated QR factorization after the rank-one matrix $\alpha xy^T$ is added. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D7c **Usage:** CALL LUPQR (*14 parameters*) **Precision:** Single (Double: DLUPQR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**LUREFF.** Refinement of solution to linear equations full storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2a1 **Usage:** CALL LUREFF (*11 parameters*) **Also see:** LUDATF LUELMF **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUREFP.** Refinement of solution to linear equations positive definite matrix – symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b1b **Usage:** CALL LUREFP (A, B, UL, N, X, IDGT, RES, IER) **Also see:** LUDECP LUELMP **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LUREPB.** Refinement of solution to linear equations positive definite band symmetric matrix-band symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D2b2 **Usage:** CALL LUREPB (*11 parameters*) **Also see:** LUDAPB **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**LVALS.** Prints letter-value display – median, hinges, eighths, etc., and optionally saves results. *Command in MINITAB proprietary interactive system.* **Classes:** L1a1 **Usage:** LVALs display of C [put letter values into C [mids into C [spreads into C]]] **Precision:** Single **Availability:** 855NOS.

# M

**M01AAF.** Sorts an array of pointers via a detached key sort to give the ascending order of a given real array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1b **Usage:** CALL M01AAF (A, M, N, IP, IST, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ABF.** Sorts an array of pointers via a detached key sort to give the descending order of a given real array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1b **Usage:** CALL M01ABF (A, M, N, IP, IST, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ACF.** Sorts an array of pointers via a detached key sort to give the ascending order of a given integer array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1a **Usage:** CALL M01ACF (IA, M, N, IP, IST, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ADF.** Sorts an array of pointers via a detached key sort to give the descending order of a given integer array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1a **Usage:** CALL M01ADF (IA, M, N, IP, IST, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AEF.** Sorts the rows of a real matrix into ascending order of an index column. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01AEF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AFF.** Sorts the rows of a real matrix into descending order of an index column. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01AFF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AGF.** Sorts the rows of an integer matrix into ascending order of an index column. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01AGF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AHF.** Sorts the rows of an integer matrix into descending order of an index column. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01AHF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AJF.** Sorts a vector of real numbers into ascending order and provides an index indicating the position of the sorted numbers in the original array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01AJF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AKF.** Sorts a vector of real numbers into descending order and provides an index indicating the position of the sorted numbers in the original array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01AKF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ALF.** Sorts a vector of integer numbers into ascending order and provides an index indicating the position of the sorted numbers in the original array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01ALF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AMF.** Sorts a vector of integer numbers into descending order and provides an index indicating the position of the sorted numbers in the original array. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01AMF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ANF.** Sorts a vector of real numbers into ascending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01ANF (A, I, J, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01APF.** Sorts a vector of real numbers into descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01APF (A, I, J, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01AQF.** Sorts a vector of integer numbers into ascending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01AQF (IA, I, J, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ARF.** Sorts a vector of integer numbers into descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01ARF (IA, I, J, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01BAF.** Sorts an integer vector containing character data into reverse alphabetic order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2c **Usage:** CALL M01BAF (IA, I, J, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01BBF.** Sorts a vector containing character data into alphabetic order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2c **Usage:** CALL M01BBF (IA, I, J, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01BCF.** Sorts selected columns (records) of an integer array containing character data, so that the elements of index rows (keys) are in reverse alphabetic order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2c **Usage:** CALL M01BCF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01BDF.** Sorts selected columns (records) of an integer array containing character data, so that the elements of index rows (keys) are in alphabetic order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2c **Usage:** CALL M01BDF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01CAF.** Rearranges a vector of real numbers into ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2b **Usage:** CALL M01CAF (RV, M1, M2, ORDER, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01CBF.** Rearranges a vector of integer numbers into ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2a **Usage:** CALL M01CBF (IV, M1, M2, ORDER, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01CCF.** Rearranges a vector of character data so that a specified substring is in ASCII or reverse ASCII order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a2c **Usage:** CALL M01CCF (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DAF.** Ranks a vector of real numbers in ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1b **Usage:** CALL M01DAF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DBF.** Ranks a vector of integer numbers in ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1a **Usage:** CALL M01DBF (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DCF.** Ranks a vector of character data in ASCII or reverse ASCII order of a specified substring. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1c **Usage:** CALL M01DCF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DFF.** Ranks the rows of a matrix of integer numbers in ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1a **Usage:** CALL M01DFF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DJF.** Ranks the columns of a matrix of real numbers in ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1b **Usage:** CALL M01DJF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DKF.** Ranks the columns of a matrix of integer numbers in ascending or descending order. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1a **Usage:** CALL M01DKF (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01DZF.** Ranks arbitrary data according to a user-supplied comparison routine. *Proprietary Fortran subroutine in NAG.* **Classes:** N6a1 **Usage:** CALL M01DZF (*5 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01EAF.** Rearranges a vector of real numbers into the order specified by a vector of ranks. *Proprietary Fortran subroutine in NAG.* **Classes:** N8 **Usage:** CALL M01EAF (RV, M1, M2, IRANK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01EBF.** Rearranges a vector of integer numbers into the order specified by a vector of ranks. *Proprietary Fortran subroutine in NAG.* **Classes:** N8 **Usage:** CALL M01EBF (IV, M1, M2, IRANK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ECF.** Rearranges a vector of character data into the order specified by a vector of ranks. *Proprietary Fortran subroutine in NAG.* **Classes:** N8 **Usage:** CALL M01ECF (CH, M1, M2, IRANK, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ZAF.** Inverts a permutation, and hence converts a rank vector to an index vector, or vice versa. *Proprietary Fortran subroutine in NAG.* **Classes:** N8 **Usage:** CALL M01ZAF (IPERM, M1, M2, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ZBF.** Checks the validity of a permutation. *Proprietary Fortran subroutine in NAG.* **Classes:** N8 **Usage:** CALL M01ZBF (IPERM, M1, M2, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**M01ZCF.** Decomposes a permutation into cycles, as an aid to re-ordering ranked data. *Proprietary Fortran subroutine in NAG.* **Classes:** N8 **Usage:** CALL M01ZCF (IPERM, M1, M2, ICYCL, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**MAFLT.** Perform simple moving average; return filtered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL MAFLT (Y, N, K, YF, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MAMME.** Compute method of moments estimates of the moving average parameters of an ARMA model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2d2 **Usage:** CALL MAMME (*9 parameters*) **Precision:** Single (Double: DMAMME) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MANN-WHITNEY.** Performs one- or two-sided two-sample rank test (a.k.a. Wilcoxon rank test) for the difference between two population medians, and calculates the corresponding point and confidence interval estimates. *Command in MINITAB proprietary interactive system.* **Classes:** L4b1b **Usage:** MANN-Whitney [alternative K], [percent confidence K] for data in C and C **Precision:** Single **Availability:** 855NOS.

**MANOVA.** Generalized multivariate analysis of variance for designs including randomized block, split-plot, crossed and/or nested, and repeated measures, with or without covariates. Options: multivariate statistics, contrasts, plots, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L7d **Precision:** Single

**Availability:** 855NOS, 855VE, 840NOS.

**MAPFL.** To map (part of) a common block into a local file, rather than letting the system map it to the drop-file. *Fortran/meta subroutine in MAGEV.* **Classes:** R **Usage:** CALL MAPFL(LFN, X, LEN, MAP) **Precision:** Single **Availability:** 205.

**MATLAB.** An interactive system for defining and manipulating matrices. It includes solving linear systems, linear least squares, eigenvalue and eigenvector calculation, QR decomposition, singular value decomposition and inverses. *MATLAB is an interactive system.* **Classes:** D **Availability:** 855NOS, CAMVAX.

**MAX.** Computes the sample maximum of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** D1a2, N5a **Usage:** CALL MAX(X, N, IWRITE, XMAX) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MCCF.** Compute the multichannel cross-correlation function of two mutually stationary multichannel time series. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10d **Usage:** CALL MCCF (*21 parameters*) **Precision:** Single (Double: DMCCF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MCRCR.** Multiply two complex rectangular matrices, AB. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b6 **Usage:** CALL MCRCR (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDBETA.** Beta probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1b **Usage:** CALL MDBETA (X, A, B, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDBETI.** Inverse beta probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2b **Usage:** CALL MDBETI (P, A, B, X, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDBIN.** Binomial probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1b **Usage:** CALL MDBIN (K, N, P, PS, PK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDBNOR.** Bivariate normal probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5b1n **Usage:** CALL MDBNOR (X, Y, RHO, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDCH.** Chi-squared probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1c **Usage:** CALL MDCH (CS, DF, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDCHI.** Inverse chi-squared probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2c **Usage:** CALL MDCHI (P, DF, X, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDCHN.** Non-central chi-squared probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1c **Usage:** CALL MDCHN (CS, DF, PNONC, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDFD.** F probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1f **Usage:** CALL MDFD (F, N1, N2, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDFDRE.** F probability distribution function (integer or fractional degrees of freedom). *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1f **Usage:** CALL MDFDRE (X, DFN, DFD, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDFI.** Inverse F probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2f **Usage:** CALL MDFI (P, D1, D2, X, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDFLT.** Smooth a periodogram by applying a sequence of modified Daniell filters; return the smoothed periodogram. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL MDFLT (*6 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MDGAM.** Gamma probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1g **Usage:** CALL MDGAM (X, P, PROB, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDGC.** General cumulative probability distribution function, given ordinates of the density. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1g **Usage:** CALL MDGC (X, F, M, IOPT, B, C, IC, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDGCI.** Inverse of a general cumulative probability distribution function, given ordinates of the density. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2g **Usage:** CALL MDGCI (P, F, M, IOPT, B, C, IC, X, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDHYP.** Hypergeometric probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1h **Usage:** CALL MDHYP (K, N, L, ND, PEQK, PLEK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDNOR.** Normal or Gaussian probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1n, C8a **Usage:** CALL MDNOR (Y, P) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDNRIS.** Inverse standard normal (Gaussian) probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2n **Usage:** CALL MDNRIS (P, Y, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDSMR.** Kolmogorov-Smirnov statistics asymptotic probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1k **Usage:** CALL MDSMR (X, P1, P2) **Precision:** Single **Availability:** 855NOS,

855VE, 205, 840NOS.

**MDSTI.** Inverse of a modification of Student's t probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2t **Usage:** CALL MDSTI (Q, F, X, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDTD.** Student's t probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1t **Usage:** CALL MDTD (TVAL, DF, Q, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDTN.** Non-central t probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1t **Usage:** CALL MDTN (TVAL, IDF, D, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDTNF.** Integral related to calculation of non-central t and bivariate normal probability distribution functions. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1t, L5b1n **Usage:** CALL MDTNF (Y, Z, EPS, T) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MDTPS.** Cumulative probability and, optionally, individual terms of the Poisson probability distribution function. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1p **Usage:** CALL MDTPS (K, RLAM, IOPT, T, P) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MEAN.** Computes the sample mean of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL MEAN(X, N, IWRITE, XMEAN) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MEANS.** Produces univariate descriptive statistics for numeric variables in an entire data set or for groups of observations in the data set. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L1a1, L1c1 **Precision:** Single **Availability:** CAMVAX.

**MEDIAN.** Computes the sample median of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL MEDIAN(X, N, IWRITE, XMED) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MEDIAN POLISH.** Performs a 1-5 way median polish on balanced data. Output includes coefficients, standard deviations of the coefficients, predicted values, residuals, residual standard deviations, residual degrees of freedom, as well as replication standard deviation, replication degrees of freedom, and lack-of-fit F-statistic if there is replication. *Command(s) in DATAPLOT interactive system.* **Classes:** L7b, L7d1, L9d **Usage:** MEDIAN POLISH <RESPONSE VARIABLE> <LIST OF ONE TO FIVE INDEPENDENT VARIABLES> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**MEDPL.** Compute a median polish of a two-way table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7b, L9d **Usage:** CALL MEDPL (*8 parameters*) **Precision:** Single (Double: DMEDPL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MERFCI.** Inverse complementary error function. *Proprietary Fortran subroutine in IMSL.* **Classes:** C8a, L5a2e **Usage:** CALL MERFCI (P, Y, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MERFI.** Inverse error function. *Proprietary Fortran subroutine in IMSL.* **Classes:** C8a, L5a2e **Usage:** CALL MERFI (P, Y, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MERRCZ.** Computes approximate values of exp($-z^2$)erfc(-iz) for complex z. *Proprietary Fortran subroutine in IMSL.* **Classes:** C8a **Usage:** CALL MERRCZ(Z, W, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MHIST.** Produces multivariate histograms. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e1 **Usage:** CALL MHIST(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**MHSWIEN.** *Half precision version of MSWIEN.*

**MIDM.** Computes the sample midmean, i.e., the sample 25% (on each side) trimmed mean of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL MIDM(X, N, IWRITE, XMIDM) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MIDR.** Computes the sample midrange of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL MIDR(X, N, IWRITE, XMIDR) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MIN.** Computes the sample minimum of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** D1a2, N5a **Usage:** CALL MIN(X, N, IWRITE, XMIN) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MINFIT.** Compute singular value decomposition of rectangular real matrix and solve related linear least squares problem. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D6 **Usage:** CALL MINFIT(NM, M, N, A, W, IP, B, IERR, RV1) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**MINITAB.** Vector and matrix commands include COPY (vectors, vectors to matrices, and conversely), DIAGONAL (create a diagonal matrix or extract the diagonal of a matrix), TRANSPOSE, INVERSE, and EIGEN (calculate eigenvalues and eigenvectors for a symmetric matrix). *Commands in MINITAB proprietary interactive system.* **Classes:** D **Precision:** Single **Availability:** 855NOS.

**MINITAB.** Subsetting commands include PICK, CHOOSE, and OMIT for selecting or deleting entries in a vector in a Minitab worksheet. *Commands in MINITAB proprietary interactive system.* **Classes:** L2c **Precision:** Single **Availability:** 855NOS.

**MINITAB.** Vector summarization commands include COUNT, SUM, MEAN, MAX, MIN, MEDIAN, N, NMISS (number of missing values), STDEV, SSQ (sum of squares), DESCRIBE (N, MEAN, MEDIAN, STDEV, MAX, MIN, 5% trimmed mean, quartiles) for columns or rows (use prefix R, e.g., RMEAN) of data in the Minitab worksheet. *Commands in MINITAB proprietary interactive system.* **Classes:** L1a1, L1c1 **Precision:** Single **Availability:** 855NOS.

**MINITAB.** Vector transformation commands include ADD, SUBTRACT, MULTIPLY, DIVIDE, RAISE, SIN, COS, TAN, ASIN, ACOS, ATAN, LOGE, LOGTEN, EXPONENTIAL, ANTILOG, ABSOLUTE VALUE, ROUND, SIGNS, SQRT, INDICATOR, RECODE, SUBSTITUTE, CONVERT, PARSUM, PARPRODUCT, and LET (to combine commands). *Commands in MINITAB proprietary interactive system.* **Classes:** L2a **Precision:** Single **Availability:** 855NOS.

**MISS.** Replaces a missing data value by the cluster mean. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** CALL MISS(MM, M, N, A, NC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**MIX.** Fits the mixture model by a maximum log-likelihood criterion. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL MIX(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**MIXIND.** Fits the mixture model from k multivariate normals where k is the desired number of clusters. The variables are assumed to have variance constant over different clusters. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL MIXIND(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**MKFL.** Given M, the mantissa of a floating-point number, and E, its exponent, MKFL returns $MB^E$, where B is the base of the machine. *Proprietary Fortran function in PORT.* **Classes:** A6c **Usage:** R = MKFL(E, M) **Also see:** UMKFL **Precision:** Single (Double: DMKFL) **Availability:** 855NOS, 205.

**MLOCAR.** Locally fits autoregressive models to non-stationary time series by minimum AIC procedure. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2f **Precision:** Single **Availability:** 855NOS.

**MLOMAR.** Locally fits multivariate autoregressive models to non-stationary time series by the minimum AIC procedure using the Householder transformation. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10c **Precision:** Single **Availability:** 855NOS.

**MLSE.** Compute least squares estimates of a linear regression model for a multichannel time series with a specified base channel. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10d **Usage:** CALL MLSE (*12 parameters*) **Precision:** Single (Double: DMLSE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSI0.** Modified Bessel function $I_0(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C10b1 **Usage:** R = MMBSI0(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSI1.** Modified Bessel function $I_1(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C10b1 **Usage:** R = MMBSI1(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSIN.** Sequence of modified Bessel functions $I_n(x)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10b1 **Usage:** CALL MMBSIN(ARG, N, B, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSIR.** Sequence of modified Bessel functions $I_r(x)$ or $\exp(-x)I_r(x)$, for r real. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10b3 **Usage:** CALL MMBSIR(ARG, ORDER, NB, IOPT, B, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSJ0.** Bessel function $J_0(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C10a1 **Usage:** R = MMBSJ0(ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSJ1.** Bessel function $J_1(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C10a1 **Usage:** R = MMBSJ1(ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSJN.** Sequence of Bessel functions $J_n(x)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10a1 **Usage:** CALL MMBSJN(ARG, N, B, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSJR.** Sequence of Bessel functions $J_r(x)$, for r real and positive. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10a3 **Usage:** CALL MMBSJR(ARG, ORDER, N, RJ, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSK0.** Modified Bessel function $K_0(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C10b1 **Usage:** R = MMBSK0(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSK1.** Modified Bessel function $K_1(x)$. *Proprietary Fortran function in IMSL.* **Classes:** C10b1 **Usage:** R = MMBSK1(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSKR.** Sequence of modified Bessel functions $K_r(x)$ or $\exp(x)K_r(x)$, for r real. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10b3 **Usage:** CALL MMBSKR(ARG, ORDER, N, BK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBSYN.** Sequence of Bessel functions $Y_r(x)$, for r real and positive. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10a3 **Usage:** CALL MMBSYN (ARG, ORDER, N, YN, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBZIN.** Sequence of modified Bessel functions $I_n(z)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10b2 **Usage:** CALL MMBZIN(X, Y, N, BR, BI, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMBZJN.** Sequence of Bessel functions $J_n(z)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10a2 **Usage:**

CALL MMBZJN(X, Y, N, BR, BI, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMDAS.** Dawson's integral. *Proprietary Fortran function in IMSL.* **Classes:** C8c **Usage:** R = MMDAS(ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMDEI.** Exponential integrals. *Proprietary Fortran function in IMSL.* **Classes:** C5 **Usage:** R = MMDEI(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMDELE.** Complete elliptic integral E(m), see [AMS55 (17.3.3)]. *Proprietary Fortran function in IMSL.* **Classes:** C14 **Usage:** R = MMDELE(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMDELK.** Complete elliptic integral K(m), see [AMS55 (17.3.1)]. *Proprietary Fortran function in IMSL.* **Classes:** C14 **Usage:** R = MMDELK(IOPT, ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMDEN.** Exponential integrals of integer order for real argument x scaled by exp(x). *Proprietary Fortran subroutine in IMSL.* **Classes:** C5 **Usage:** CALL MMDEN(X, N, F, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMKEL0.** Kelvin functions $ber_0(x)$, $bei_0(x)$, $ker_0(x)$, $kei_0(x)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10c **Usage:** CALL MMKEL0 (X, BER, BEI, XKER, XKEI, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMKEL1.** Kelvin functions $ber_1(x)$, $bei_1(x)$, $ker_1(x)$, $kei_1(x)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10c **Usage:** CALL MMKEL1 (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMKELD.** Derivatives of the Kelvin functions $ber_0(x)$, $bei_0(x)$, $ker_0(x)$, and $kei_0(x)$. *Proprietary Fortran subroutine in IMSL.* **Classes:** C10c **Usage:** CALL MMKELD (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMLINC.** Evaluates an elementary integral from which inverse circular functions, logarithms or inverse hyperbolic functions may be computed. *Proprietary Fortran function in IMSL.* **Classes:** C4 **Usage:** R = MMLINCC(X, Y, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMLIND.** Carlson's incomplete elliptic integral RD(x,y,z). *Proprietary Fortran function in IMSL.* **Classes:** C14 **Usage:** R = MMLIND(X, Y, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMLINF.** Carlson's incomplete elliptic integral RF(x,y,z). *Proprietary Fortran function in IMSL.* **Classes:** C14 **Usage:** R = MMLINF(X, Y, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMLINJ.** Carlson's incomplete elliptic integral RJ(x,y,z,p). *Proprietary Fortran function in IMSL.* **Classes:** C14 **Usage:** R = MMLINJ(X, Y, Z, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMPSI.** Logarithmic derivative of the gamma function. *Proprietary Fortran function in IMSL.* **Classes:** C7c **Usage:** R = MMPSI(ARG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMWPL.** Weierstrass P-function with primitive half-periods 1/2 [AMS55 (18.1)]. The corresponding invariants are $g_2$=4L, where L=2.62205... is the Lemniscate Constant [AMS55 (18.14.7)] and $g_3$=0. *Proprietary Fortran subroutine in IMSL.* **Classes:** C15 **Usage:** CALL MMWPL(Z, PLEM, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMWPL1.** First derivative of MMWPL. *Proprietary Fortran subroutine in IMSL.* **Classes:** C15 **Usage:** CALL MMWPL1(Z, PLEM1, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMWPQ.** Weierstrass P-function in the equianharmonic case for complex argument with unit period parallelogram. *Proprietary Fortran subroutine in IMSL.* **Classes:** C15 **Usage:** CALL MMWPQ(Z, PEQ, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MMWPQ1.** First derivative of MMWPQ. *Proprietary Fortran subroutine in IMSL.* **Classes:** C15 **Usage:** CALL MMWPQ1(Z, PEQ1, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MNPB.** Creates a B-spline mesh from an array of fitting points, using at least n fitting points in each interval. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL MNPB (X, NX, N, K, T, NT) **Precision:** Single (Double: DMNPB) **Availability:** 855NOS, 205.

**MODAL.** Produces a bivariate histogram. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3b1 **Usage:** CALL MODAL(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**MOL1D.** Solves systems of linear or nonlinear initial-boundary-value problems in one space dimension. Can solve hyperbolic equations with or without discontinuities, parabolic equations (including reaction-diffusion equations). Uses the method of lines based on equi-spaced finite differences. Graphical output available. *Fortran subroutine in PDELIB (MOL1D sublibrary).* **Classes:** I2a1a, I2a2 **Usage:** CALL MOL1D(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 205.

**MOLCH.** Solve a system of partial differential equations of the form $U_t = F(X,t,U,U_x,U_{xx})$ using the method of lines with cubic Hermite polynomials. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** I2a1a **Usage:** CALL MOLCH (*12 parameters*) **Precision:** Single (Double: DMOLCH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MONOD.** *Double precision version of MONOR.*

**MONOI.** Test if an integer vector is monotone increasing or decreasing. *Proprietary Fortran function in PORT.* **Classes:** R2 **Usage:** L = MONOI(X, N, INC) **Precision:** Single **Availability:** 855NOS, 205.

**MONOR.** Test if a real vector is monotone increasing or decreasing. *Proprietary Fortran function in PORT.* **Classes:**

R2 **Usage:** L = MONOR(X, N, INC) **Precision:** Single (Double: MONOD) **Availability:** 855NOS, 205.

**MOVE.** Copies M elements of the vector X (starting with position IX1) into the vector Y (starting with position IY1). *Fortran subroutine in DATAPAC.* **Classes:** D1a5 **Usage:** CALL MOVE(X, M, IX1, IY1, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**MOVEBC.** Move a complex vector using backward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEBC (N, A, B) **Precision:** Single **Availability:** 855NOS, 205.

**MOVEBD.** *Double precision version of MOVEBR.*

**MOVEBI.** Move an integer vector using backward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEBI (N, A, B) **Precision:** Single **Availability:** 855NOS, 205.

**MOVEBL.** Move a logical vector using backward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEBL (N, A, B) **Precision:** Single **Availability:** 855NOS, 205.

**MOVEBR.** Move a real vector using backward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEBR (N, A, B) **Precision:** Single (Double: MOVEBD) **Availability:** 855NOS, 205.

**MOVEFC.** Move a complex vector using forward DO loops. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEFC (N, A, B) **Precision:** Double **Availability:** 855NOS, 205.

**MOVEFD.** Move a double precision vector using forward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEFD (N, A, B) **Precision:** Double **Availability:** 855NOS, 205.

**MOVEFI.** Move an integer vector using forward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEFI (N, A, B) **Precision:** Single **Availability:** 855NOS, 205.

**MOVEFL.** Move a logical vector using forward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEFL (N, A, B) **Precision:** Single **Availability:** 855NOS, 205.

**MOVEFR.** Move a real vector using forward DO loop. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a5 **Usage:** CALL MOVEFR (N, A, B) **Precision:** Single (Double: MOVRFD) **Availability:** 855NOS, 205.

**MPLOT.** Prints multiple scatter diagrams on the same axis. *Command in MINITAB proprietary interactive system.* **Classes:** L3e3a **Usage:** MPLOt C vs C and C vs C [and ...] **Precision:** Single **Availability:** 855NOS.

**MPOLISH.** Uses median polish to fit an additive model to a two-way layout which may be unbalanced and may have empty cells. Options: fit columns first, number of iterations, save results. *Command in MINITAB proprietary interactive system.* **Classes:** L7b, L9d **Usage:** MPOLish the data in C, row levels in C, column levels in C [put residuals into C [fits into C]] [; COLUMNS first; ITERATIONS = K; EFFECTS, put common into K, row effects into C, columns effects into C; COMPARISON values, put into C.] **Precision:** Single **Availability:** 855NOS.

**MPP.** Print plot of multiple Y vectors versus a common X vector; linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3a **Usage:** CALL MPP (YM, X, N, M, IYM) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MPPC.** Print plot of multiple Y vectors versus a common X vector; log or linear axes; user-supplied control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3a **Usage:** CALL MPPC (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MPPL.** Print plot of multiple Y vectors versus a common X vector; log or linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3a **Usage:** CALL MPPL (YM, X, N, M, IYM, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MPPM.** Print plot of multiple Y vectors versus a common X vector; linear axes; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3a **Usage:** CALL MPPM (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MPPMC.** Print plot of multiple Y vectors versus a common X vector; log or linear axes; user-supplied control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3a **Usage:** CALL MPPMC (*14 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MPPML.** Print plot of multiple Y vectors versus a common X vector; log or linear axes; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3a **Usage:** CALL MPPML (*8 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MPYUTU.** To perform the matrix multiplication S = LU, where L is the transpose of the supplied upper triangular matrix U, and S is the symmetric product matrix. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b6 **Usage:** CALL MPYUTU(N, U, KU, S, KS, NLP) **Precision:** Single (Half: HMPYUTU) **Availability:** 205 (vectorized)

**MRGSORT.** To merge up to 4 already sorted subfiles residing in arrays into one sorted file. With subfile or file denoting an array containing a real or integer key (one 64-bit word per key element) of length N words, plus a two-dimensional array A(LR,N) containing the N records of length LR. The sorting order is non-decreasing, or ascending, with respect to the algebraic values of the key-elements. *Fortran/meta subroutine in MAGEV.* **Classes:** N7 **Usage:** CALL MRGSORT(*15 parameters*) **Precision:** Single **Availability:** 205 (vectorized)

**MRRRR.** Multiply two real rectangular matrices, AB. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1b6 **Usage:** CALL MRRRR (*12 parameters*) **Precision:** Single (Double: DMRRRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MSDST.** Compute distances in a multidimensional scaling model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L16 **Usage:** CALL MSDST (*13 parameters*) **Precision:** Single (Double: DMSDST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MSENO.** Expected values of normal order statistics. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a2n **Usage:** CALL MSENO (IFIRST, ILAST, N, EX, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MSIDV.** Perform individual-differences multidimensional scaling for metric data using alternating least squares. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L16 **Usage:** CALL MSIDV (*20 parameters*) **Precision:** Single (Double: DMSIDV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MSMRAT.** Ratio of the ordinate to the upper tail area of the standardized normal (Gaussian) distribution. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1n **Usage:** CALL MSMRAT (X, RM, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MSWIEN.** To solve one or several matrix equations of the type TX=A, where T is a Toeplitz matrix, X is a vector with its first element equal to 1.0, and A is a base vector of the form ($\alpha$,0,0,...,0). The algorithm used is known to geophysicists as the spike version of the Wiener-Levenson algorithm. *Fortran/meta function in MAGEV.* **Classes:** D2a **Usage:** I = MSWIEN(*9 parameters*) **Precision:** Single (Half: MHSWIEN) **Availability:** 205 (vectorized)

**MUCBV.** Multiply a complex band matrix in band storage mode by a complex vector. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b4 **Usage:** CALL MUCBV (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MUCRV.** Multiply a complex rectangular matrix by a complex vector. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b4 **Usage:** CALL MUCRV (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**MULBAR.** Determines multivariate autoregressive models by a Bayesian procedure. The basic least squares estimates of the parameters are obtained by the Householder transformation. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10c **Precision:** Single **Availability:** 855NOS.

**MULMAR.** Fits a multivariate autoregressive model by the minimum AIC procedure. Only the possibilities of zero coefficients at the beginning and end of the model are considered. The least squares estimate of the parameters are obtained by the Householder transformation. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10c **Precision:** Single **Availability:** 855NOS.

**MULT RESPONSE.** Combines elementary variables into multiple dichotomy groups and multiple response groups to produce univariate and multivariate tables. Options: cell content and percentages, missing values, stub and banner tables. *Proprietary stand-alone program using SPSS command language.* **Classes:** L2a **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**MULTIPLOT.** Generates a matrix of bivariate plots; valid with any DATAPLOT plot command. *Command(s) in DATAPLOT interactive system.* **Classes:** L3e3d **Usage:** MULTIPLOT <NUMBER OF ROWS IN MATRIX> <NUMBER OF COLUMNS IN MATRIX> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**MURBV.** Multiply a real band matrix in band storage mode by a real vector. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b4 **Usage:** CALL MURBV (*10 parameters*) **Precision:** Single (Double: DMURBV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MURRV.** Multiply a real rectangular matrix by a vector. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b4 **Usage:** CALL MURRV (*9 parameters*) **Precision:** Single (Double: DMURRV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MVIND.** Compute a test for the independence of K sets of multivariate normal variables. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4e **Usage:** CALL MVIND (*7 parameters*) **Precision:** Single (Double: DMVIND) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MVP.** Print vertical plot of multiple vectors versus input order on a linear axis, using default control values and axis limits, with no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3c **Usage:** CALL MVP (YM, N, M, IYM, NS) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MVPC.** Print vertical plot of multiple vectors versus input order on a log or linear axis, using user-supplied control values and axis limits, with no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3c **Usage:** CALL MVPC (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MVPL.** Print vertical plot of multiple vectors versus input order on a log or linear axis, using default control values and axis limits, with no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3c **Usage:** CALL MVPL (YM, N, M, IYM, NS, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MVPM.** Print vertical plot of multiple vectors versus input order on a linear axis, using default control values and axis limits, with no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3c **Usage:** CALL MVPM (YM, YMMISS, N, M, IYM, NS) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MVPMC.** Print vertical plot of multiple vectors versus input order on a log or linear axis, using user-supplied control values and axis limits, with no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3c **Usage:** CALL MVPMC (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MVPML.** Print vertical plot of multiple vectors versus input order on a log or linear axis, using default control values and axis limits, with no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3e3c **Usage:** CALL MVPML (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**MWFE.** Compute least squares estimates of the multichannel Wiener filter coefficients for two mutually stationary multichannel time series. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10d **Usage:** CALL MWFE (*16 parameters*) **Precision:** Single (Double: DMWFE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MXCMP.** To compare two matrices A and B by computing the maximum relative error of the columns in B with respect to those in A. Both 2-norms and infinity-norms are computed. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b **Usage:** CALL MXCMP(M, N, A, LA, B, LB, ERR, TOL) **Precision:** Single (Half: HMXCMP) **Availability:** 205 (vectorized)

**MXEQ.** To determine whether two matrices are identical. *Fortran/meta function in MAGEV.* **Classes:** D1b **Usage:** I = MXEQ(M, N, A, LA, B, LB) **Precision:** Single (Half: HMXEQ) **Availability:** 205 (vectorized)

**MXINV.** To compute the inverses of a set of square matrices, using neither scaling nor pivoting. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a1 **Usage:** CALL MXINV(NMX, N, A, B) **Precision:** Single **Availability:** 205 (vectorized)

**MXMOV.** To move a matrix from one location to another. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b8 **Usage:** CALL MXMOV(M, N, A, LA, B, BB) **Precision:** Single (Half: HMXMOV) **Availability:** 205 (vectorized)

**MXMPY.** To perform the matrix multiplication C = AB, where C, A and B are either all columnwise or all rowwise stored rectangular matrices of matching dimensions. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b6 **Usage:** CALL MXMPY(K, L, M, LA, LB, LC, A, B, C, NLP) **Precision:** Single (Half: HMXMPY) **Availability:** 205 (vectorized)

**MXMPYI.** To perform the matrix multiplication C = AB, where C, A and B are either all columnwise or all rowwise stored rectangular matrices of matching dimensions. This routine uses the hardware dot product instruction. Thrashing will result if the matrices are too large. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b6 **Usage:** CALL MXMPYI(K, L, M, LA, LB, LC, A, B, C) **Precision:** Single (Half: HMXMPYI) **Availability:** 205 (vectorized)

**MXMPYT.** To perform the matrix multiplication $C=A^TB$, where C, A and B are either all columnwise or all rowwise stored rectangular matrices of matching dimensions. This routine uses the hardware dot product instruction. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b6 **Usage:** CALL MXMPYT(*10 parameters*) **Precision:** Single (Half: HMXMPYT) **Availability:** 205 (vectorized)

**MXTXF.** Compute the transpose product of a matrix, $A^TA$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1b6 **Usage:** CALL MXTXF (*7 parameters*) **Precision:** Single (Double: DMXTXF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MXTYF.** Multiply the transpose of matrix A by matrix B, $A^TB$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1b6 **Usage:** CALL MXTYF (*12 parameters*) **Precision:** Single (Double: DMXTYF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**MXYTF.** Multiply a matrix A by the transpose of a matrix B, $AB^T$. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b6 **Usage:** CALL MXYTF (*12 parameters*) **Precision:** Single (Double: DMXYTF) **Availability:** 855NOS, 855VE, 205, 840NOS.

# N

**N1ECC.** Allows entry or correction of a rectangular matrix or vector from the keyboard, with input checking. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1ECC(A, LDA, M, N, ENT, KE) **Precision:** Single (Double: N1ECCD) **Availability:** PC.

**N1ECCD.** *Double precision version of N1ECC.*

**N1EEC.** Allows entry or correction of a symmetric matrix stored in packed form from the keyboard, with input checking. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1EEC(A, N, W, ENT, KE) **Precision:** Single (Double: N1EECD) **Availability:** PC.

**N1EECD.** *Double precision version of N1EEC.*

**N1LM.** Allows screen display, entry, and correction, of a large vector or matrix. Elements, rows, or columns can be entered, changed or set to zero. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1LM(A, LDA, M, N, IZERO) **Precision:** Single (Double: N1LMD) **Availability:** PC.

**N1LMD.** *Double precision version of N1LM.*

**N1M.** Prints a matrix on the video display screen or on a printer. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1M(*12 parameters*) **Precision:** Single (Double: N1MD) **Availability:** PC.

**N1MD.** *Double precision version of N1M.*

**N1MSD.** Prints a matrix on the video display screen or on a printer, allowing specification of the number of output digits desired. Accepts either single or double precision input array *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1MSD (*13 parameters*) **Precision:** Single **Availability:** PC (Accepts either single or double precision input array.)

**N1RG.** Writes the Mth row of a matrix, or N elements of a vector, using G15.7 format. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1RG(A, LDA, M, N, ID, CHAR, KE) **Precision:** Single (Double: N1RGD) **Availability:** PC (Double precision routine N1RGD writes in G23.15 format)

**N1RGD.** *Double precision version of N1RG.*

**N1RS.** Writes the Mth row of a symmetric matrix stored in packed form, using G15.7 format. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1RS(A, M, N, ID, CHAR, V, KE) **Precision:** Single (Double: N1RSD) **Availability:** PC (Double precision routine N1RSD writes in G23.15 format.)

**N1RSD.** *Double precision version of N1RS.*

**N1TC.** Identifies array output by printing page number, title, and column numbers for an array that is subsequently to be printed. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1TC(*7 parameters*) **Precision:** Single (Double: N1TCD) **Availability:** PC.

**N1TCD.** *Double precision version of N1TC.*

**N1V.** Prints a vector (array) on the video display screen or a line printer. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1V(*9 parameters*) **Precision:** Single (Double: N1VD) **Availability:** PC.

**N1VD.** *Double precision version of N1V.*

**N1VSD.** Prints a vector (array) on the video display screen or a line printer, allowing specification of the number of output digits desired. Accepts either single or double precision input array. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N1VSD (*10 parameters*) **Precision:** Single **Availability:** PC (Accepts either single or double precision input array.)

**N3.** Reads a datum as a character string of length 20 from the keyboard and converts it to the desired type if it is possible. Otherwise it returns an error code. Also converts all lower case letters to upper case. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N1 **Usage:** CALL N3(IDW, C8, R4, R8, I4) **Availability:** PC.

**N3FNB.** Returns the index of the first non-blank character in an input string. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N3 **Usage:** CALL N3FNB(STRNG, IEND, IND) **Availability:** PC.

**N3LNB.** Returns the index of the last non-blank character in an input string. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N3 **Usage:** CALL N3LNB(STRNG, IEND, IND) **Availability:** PC.

**N3LOW.** Converts an input string to lower case. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N3 **Usage:** CALL N3LOW(STRNG, IEND) **Availability:** PC.

**N3UP.** Converts an input string to upper case. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N3 **Usage:** CALL N3UP(STRNG, IEND) **Availability:** PC.

**N6A2.** Sorts a real array R into ascending order. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N6a2b **Usage:** CALL N6A2(R, N) **Precision:** Single **Availability:** PC.

**N6A2A.** Sorts a real array R into ascending order by absolute value. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N6a2b **Usage:** CALL N6A2A(R, N) **Precision:** Single **Availability:** PC.

**N6A2AD.** Sorts a double precision array D into ascending order by absolute value. *Proprietary Fortran subroutine* *in the Scientific Desk.* **Classes:** N6a2b **Usage:** CALL N6A2AD(D, N) **Precision:** Double **Availability:** PC.

**N6A2D.** Sorts a double precision array D into ascending order. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N6a2b **Usage:** CALL N6A2D(D, N) **Precision:** Double **Availability:** PC.

**N6A2P.** Sorts a real array R into ascending order. A pointer array is also returned. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N6a1b, N6a2b **Usage:** CALL N6A2P(R, N, IPER) **Precision:** Single **Availability:** PC.

**N6A2PD.** Sorts a double precision array D into ascending order. A pointer array is also returned. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** N6a1b, N6a2b **Usage:** CALL N6A2PD(D, N, IPER) **Precision:** Double **Availability:** PC.

**NADCON.** Produces noise adaptive optimal controller gain and simulates the controlled process. The basic state space model is obtained from a fixed input-output model of the controlled system and the locally stationary autoregressive model of the vector noise of the controlled system, fitted by either the least squares method or a Bayesian procedure. *Portable stand-alone program using TIMSAC command language.* **Classes:** L6a20 **Precision:** Single **Availability:** 855NOS.

**NAFRE.** Friedman's test for randomized complete block designs. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7b **Usage:** CALL NAFRE (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NAK1.** Kruskal-Wallis test for identical populations. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4c1b, L7a2 **Usage:** CALL NAK1 (X, NI, M, EPS, IR, R, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NAWNRP.** Wilson's ANOVA (2 or 3 way designs) without replicates. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7b, L7c **Usage:** CALL NAWNRP (Y, N, IR, NF, NDF, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NAWRPE.** Wilsons ANOVA (1, 2, or 3 way designs) with equal replication. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a2, L7b, L7c **Usage:** CALL NAWRPE (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NAWRPU.** Wilsons ANOVA (1, 2, or 3 way designs) with unequal replication. *Proprietary Fortran subroutine in IMSL.* **Classes:** L7a2, L7b, L7c **Usage:** CALL NAWRPU (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NBCDF.** Computes the cumulative distribution function value at X for the negative binomial distribution with parameters P and N. *Fortran subroutine in DATAPAC.* **Classes:** L5a1n **Usage:** CALL NBCDF(X, P, N, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NBCYC.** Noethers test for cyclical trend. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1d **Usage:** CALL NBCYC (X, N, EPS, NSTAT, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NBPPF.** Computes the percent point function value at P for the negative binomial distribution with parameters PPAR and N. *Fortran subroutine in DATAPAC.* **Classes:** L5a2n **Usage:** CALL NBPPF(P, PPAR, N, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NBQT.** Cochran q test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4c1b **Usage:** CALL NBQT (X, N, M, IA, Q, PQ, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NBRAN.** Generates a random sample of size N from the negative binomial distribution with parameters P and NPAR. *Fortran subroutine in DATAPAC.* **Classes:** L6a14 **Usage:** CALL NBRAN(N, P, NPAR, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NBSDL.** Cox and Stuart sign test for trends in dispersion and location. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1d **Usage:** CALL NBSDL (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NBSIGN.** Sign test (for percentiles). *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1b1 **Usage:** CALL NBSIGN (X, N, Q, P, NSIGN, PROB, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NCAR.** A collection of Fortran 77 programs and subroutines for displaying scientific data. Capabilities include x-y coordinate plots, contour plots, world maps, solid-colored maps, 2D vector fields, drawing lines in 3D space, halftone (gray scale), background grids, surface views, iso-surfaces of 3D data, text in various fonts, and dashed lines with user-defined patterns. Many output devices are supported; device-independent output is also supported. *NCAR proprietary Fortran subprogram library.* **Classes:** Q **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 855NOS, 855VE, 205:* Output on Dicomed film recorder, Tektronix devices, QMS printers.

**NCONF.** Solve a general nonlinear programming problem using the successive quadratic programming algorithm and a finite difference gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2h3b1a **Usage:** CALL NCONF (*13 parameters*) **Precision:** Single (Double: DNCONF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NCONG.** Solve a general nonlinear programming problem using the successive quadratic programming algorithm and a user-supplied gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2h3b1b **Usage:** CALL NCONG (*13 parameters*) **Precision:** Single (Double: DNCONG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NCTRD.** Perform the Noether test for cyclical trend. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1d **Usage:** CALL NCTRD (*6 parameters*) **Precision:** Single (Double: DNCTRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NDAYS.** Compute the number of days from January 1, 1900, to the given date. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** R **Usage:** I = NDAYS(IDAY, IMONTH, IYEAR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NDEST.** Evaluate probability density function at specified points. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1g **Usage:** CALL NDEST (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NDKER.** Nonparametric probability density function (one dimensional) estimation by the kernel method. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1b2 **Usage:** CALL NDKER (X, N, H, DEL, XKER, B, NPT, F) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NDMPLE.** Nonparametric probability density function (one dimensional) estimation by the penalized likelihood method. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1b2 **Usage:** CALL NDMPLE (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NDYIN.** Give the date corresponding to the number of days since January 1, 1900. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** R **Usage:** CALL NDYIN (NDAYS, IDAY, MONTH, IYEAR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NEGATIVE BINOMIAL PROBABILITY PLOT.** Generates a probability plot for the negative binomial distribution with parameters P and K. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2n **Usage:** NEGATIVE BINOMIAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**NEIGHBOR.** Performs a nearest neighbor discriminant analysis, classifying observations into groups according to either the nearest neighbor rule or the k-nearest-neighbor when the classes do not have multivariate normal distributions. Proximity is determined by either Mahalanobis or Euclidean distances. Options: use of prior probabilities in the classification, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L12, L14a1b **Precision:** Single **Availability:** CAMVAX.

**NEQNF.** Solve a system of nonlinear equations using the Levenberg-Marquardt algorithm and a finite-difference Jacobian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F2 **Usage:** CALL NEQNF (*7 parameters*) **Precision:** Single (Double: DNEQNF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NEQNJ.** Solve a system of nonlinear equations using the Levenberg-Marquardt algorithm with a user-supplied Jacobian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F2 **Usage:** CALL NEQNJ (*8 parameters*) **Precision:** Single (Double: DNEQNJ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NERROR.** Provides the current error number for PORT library programs. *Proprietary Fortran function in PORT.* **Classes:** R3c **Usage:** I = NERROR(NERR) **Precision:** Single **Availability:** 855NOS, 205.

**NESTED.** Performs analysis of variance and analysis of covariance for nested random designs. Especially good for designs involving large numbers of classification levels and observations. The data set must be sorted by the classification variables (assumed to form a nested set of effects). *Proprietary stand-alone program using SAS command language.* **Classes:** L7d3 **Precision:** Single **Availability:** CAMVAX.

**NHEXT.** Fishers exact method for 2-by-2 tables. *Proprietary Fortran subroutine in IMSL.* **Classes:** L9a **Usage:** CALL NHEXT (ITAB, IT, P, IER) **Also see:** BDCOU1 BDCOU2 **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NHINC.** Includance test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b1b **Usage:** CALL NHINC (X, N, Y, M, I1, I2, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NKS1.** Kolmogorov-Smirnov one-sample test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1c **Usage:** CALL NKS1 (PDF, X, N, PDIF, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NKS2.** Kolmogorov-Smirnov two-sample test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b1c **Usage:** CALL NKS2 (F, N, G, M, PDIF, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NL2S1.** Minimizes a nonlinear sum of squares using both residual and gradient values supplied by the user. *Portable Fortran subroutine in CMLIB (NL2SN sublibrary).* **Classes:** K1b1a2, L8e1b2, L8e1b4 **Usage:** CALL NL2S1(*12 parameters*) **Precision:** Single (Double: DNL2S1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**NL2SN.** Minimizes a nonlinear sum of squares using residual values only. *Portable Fortran subroutine in CMLIB (NL2SN sublibrary).* **Classes:** L8e1b1, L8e1b3, K1b1a1 **Usage:** CALL NL2SN(*11 parameters*) **Precision:** Single (Double: DNL2SN) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**NLIN.** Performs nonlinear least-squares regression using one of four iterative methods (modified Gauss-Newton, Marquardt, gradient, or steepest-descent, and multivariate secant or false position (DUD). User provides starting values for the parameters, and derivatives of the model for all but the DUD method. Options: weights, bounds on the parameter estimates, objective function to be minimized, grid search for starting values. *Proprietary stand-alone program using SAS command language.* **Classes:** L8e1 **Precision:** Single **Availability:** CAMVAX.

**NLS.** Unweighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b1 **Usage:** CALL NLS (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSC.** Unweighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b1 **Usage:** CALL NLSC (*19 parameters*) **Also see:** STPLS STPLSC **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSD.** Unweighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b2 **Usage:** CALL NLSD (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSDC.** Unweighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b2 **Usage:** CALL NLSDC (*20 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSDS.** Unweighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b2 **Usage:** CALL NLSDS (*27 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSS.** Unweighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b1 **Usage:** CALL NLSS (*26 parameters*) **Also see:** STPLS STPLSC **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSW.** Weighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b3 **Usage:** CALL NLSW (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSWC.** Weighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b3 **Usage:** CALL NLSWC (*20 parameters*) **Also see:** STPLS STPLSC **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSWD.** Weighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b4 **Usage:** CALL NLSWD (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSWDC.** Weighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b4 **Usage:** CALL NLSWDC (*21 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSWDS.** Weighted nonlinear regression by a quasi-Newton algorithm using user-supplied analytic derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b4 **Usage:** CALL NLSWDS (*29 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NLSWS.** Weighted nonlinear regression by a quasi-Newton algorithm using finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Statistical results returned via argument list, and computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8e1b3 **Usage:** CALL NLSWS (*28 parameters*) **Also see:** STPLS STPLSC **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NMCC.** Calculate and test the significance of the Kendall coefficient of concordance. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4c1b **Usage:** CALL NMCC (*9 parameters*) **Precision:** Single **Availability:** 855NOS,

855VE, 205, 840NOS.

**NMKN.** Kendall's test for correlation (rank correlation coefficient). *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b1b **Usage:** CALL NMKN (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NMKSF.** Frequency distribution of K and the probability of equalling or exceeding K, where K, the total score from the Kendall rank correlation coefficient calculations, and N, the sample size, are given. *Proprietary Fortran subroutine in IMSL.* **Classes:** L5a1k **Usage:** CALL NMKSF (K, N, Z, ZW, P) **Also see:** NMKN **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NMKTS.** K-sample trends test against ordered alternatives. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4c1b **Usage:** CALL NMKTS (X, XM, K, DSEED, XSTAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NMRANK.** Numerical ranking. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a1b **Usage:** CALL NMRANK (X, N, EPS, IR, R, RANK, S, T) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NMTIE.** Tie statistics, given a sample of observations. *Proprietary Fortran subroutine in IMSL.* **Classes:** L1a1e **Usage:** CALL NMTIE (X, M, EPS, TIES) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NNBRD.** K nearest neighbor discrimination. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L12 **Usage:** CALL NNBRD (*22 parameters*) **Precision:** Single (Double: DNNBRD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NONPAR CORR.** Produces a matrix of either Spearman's rho or Kendall's tau-b rank-order correlation coefficients with one- or two-tailed significance levels. Options: random sampling, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L4c1b **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**NORCDF.** Computes the cumulative distribution function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1n **Usage:** CALL NORCDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NORMAL PROBABILITY PLOT.** Generates a probability plot for the standard normal (Gaussian) distribution. Alternate name: GAUSSIAN PROBABILITY PLOT. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2n **Usage:** NORMAL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**NOROUT.** Performs a normal outlier analysis on the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a6 **Usage:** CALL NOROUT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NORPDF.** Computes the probability density function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a1n **Usage:** CALL NORPDF(X, PDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NORPLT.** Generates a normal (Gaussian) probability plot with mean 0 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2n **Usage:** CALL NORPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NORPPF.** Computes the percent point function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2n **Usage:** CALL NORPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NORRAN.** Generates a random sample of size N from the normal (Gaussian) distribution with mean 0 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L6a14 **Usage:** CALL NORRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NORSF.** Computes the sparsity function value for the normal (Gaussian) distribution with mean 0 and standard deviation 1. *Fortran subroutine in DATAPAC.* **Classes:** L5a2n **Usage:** CALL NORSF(P, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**NPAR TESTS.** Nonparametric tests including chi-square, runs, binomial, McNemar, sign, Wilcoxon, Kolmogorov-Smirnov, Kendall's coefficient of concordance, Cochran Q median, Mann-Whitney, Wald-Wolfowitz, Moses, Kruskal-Wallis, Friedman's two-way ANOVA. Options: univariate statistics, random sampling, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L4a1c, L4a1d, L4b1b, L4b1c, L4b3, L4b5, L4c1b, L7a2 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**NPAR1WAY.** Performs nonparametric one-way analysis of variance on ranks and four rank scores (Wilcoxon, median, van der Waerden, and Savage). *Proprietary stand-alone program using SAS command language.* **Classes:** L7a2 **Precision:** Single **Availability:** CAMVAX.

**NR1CB.** Compute the 1-norm of a complex band matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b2 **Usage:** CALL NR1CB (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NR1RB.** Compute the 1-norm of a real band matrix in band storage mode. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b2 **Usage:** CALL NR1RB (*6 parameters*) **Precision:** Single (Double: DNR1RB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NR1RR.** Compute the 1-norm of a real matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:**

D1b2 **Usage:** CALL NR1RR (NRA, NCA, A, LDA, ANORM) **Precision:** Single (Double: DNR1RR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NR2RR.** Compute the Frobenius norm of a real rectangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b2 **Usage:** CALL NR2RR (NRA, NCA, A, LDA, ANORM) **Precision:** Single (Double: DNR2RR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NRAND.** Generate a vector of normal pseudo-random numbers with zero mean and unit standard deviation. *Portable Fortran subroutine in STARPAC.* **Classes:** L6a14 **Usage:** CALL NRAND (Y, N, ISEED) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NRANDC.** Generate a vector of normal pseudo-random numbers with mean YMEAN and standard deviation SIGMA. *Portable Fortran subroutine in STARPAC.* **Classes:** L6a14 **Usage:** CALL NRANDC (Y, N, ISEED, YMEAN, SIGMA) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**NRANDOM.** Generates K pseudo-random numbers from the normal distribution with specified mean and standard deviation. *Command in MINITAB proprietary interactive system.* **Classes:** L6a14 **Usage:** NRANdom K observations with mu = K, sigma = K, put into C **Precision:** Single **Availability:** 855NOS.

**NRBHA.** Bhapkar v test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4c1b **Usage:** CALL NRBHA (X, NPS, IC, IR, W, V, Q, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NRCES.** Compute maximum likelihood estimates of the mean and variance from grouped and/or censored normal data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a3 **Usage:** CALL NRCES (*14 parameters*) **Precision:** Single (Double: DNRCES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NRIRR.** Compute the infinity norm of a real matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b2 **Usage:** CALL NRIRR (NRA, NCA, A, LDA, ANORM) **Precision:** Single (Double: DNRIRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NRWMD.** Wilcoxon signed rank test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b3 **Usage:** CALL NRWMD (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NRWRST.** Wilcoxon's rank-sum test. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b1b **Usage:** CALL NRWRST (X, M, N, EPS, IR, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**NSBJF.** Compute Box-Jenkins forecasts and their associated probability limits for a nonseasonal ARMA model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2d3 **Usage:** CALL NSBJF (*17 parameters*) **Precision:** Single (Double: DNSBJF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NSCORES.** Calculates normal scores. *Command in MINITAB proprietary interactive system.* **Classes:** L2a **Usage:** NSCOres of C, put into C **Precision:** Single **Availability:** 855NOS.

**NSLSE.** Compute least squares estimates of parameters for a nonseasonal ARMA model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2d2 **Usage:** CALL NSLSE (*20 parameters*) **Precision:** Single (Double: DNSLSE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NSPE.** Compute preliminary estimates of the autoregressive and moving average parameters of an ARMA model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2d1 **Usage:** CALL NSPE (*13 parameters*) **Precision:** Single (Double: DNSPE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NTIES.** Compute tie statistics for a sample of observations. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L1a1e **Usage:** CALL NTIES (NOBS, X, FUZZ, TIES) **Precision:** Single (Double: DNTIES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**NUMXER.** Get most current message number. *Portable Fortran function in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** I = NUMXER(NERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

# O

**OCDIS.** Pairwise Euclidean distances between the columns of a matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L14d **Usage:** CALL OCDIS (X, IX, N, M, WK, XDIS) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OCLINK.** Perform a single-linkage or complete-linkage hierarchical cluster analysis given a similarity matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L14a1a1 **Usage:** CALL OCLINK (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ODEQ.** Finds the integral of a set of functions over the same interval by using the differential equation solver ODES1. For smooth functions. *Proprietary Fortran subroutine in PORT.* **Classes:** H2a1a1 **Usage:** CALL ODEQ (N, F, A, B, EPS, ANS) **Precision:** Single (Double: DODEQ) **Availability:** 855NOS, 205.

**ODES.** Solves an initial value problem for a system of ordinary differential equations. Easy to use. *Proprietary Fortran subroutine in PORT.* **Classes:** I1a1c **Usage:** CALL ODES (*8 parameters*) **Precision:** Single (Double: DODES) **Availability:** 855NOS, 205.

**ODES1.** Solves an initial value problem for a system of ordinary differential equations. Allows great flexibility and user control. *Proprietary Fortran subroutine in PORT.* **Classes:** I1a1c **Usage:** CALL ODES1 (*11 parameters*) **Also see:** ODESH ODESE **Precision:** Single (Double: DODES1) **Availability:** 855NOS, 205.

**ODESE.** Standard error subprogram for the routine ODES1. *Proprietary Fortran function in PORT.* **Classes:** I1c **Usage:** L = ODESE(*7 parameters*) **Also see:** ODES1 **Precision:** Single (Double: DODESE) **Availability:** 855NOS, 205.

**ODESH.** Default HANDLE routine for ODES. Used to access the results at the end of each integration time step. *Proprietary Fortran subroutine in PORT.* **Classes:** I1c **Usage:** CALL ODESH (T0, X0, T1, X1, NX, DT, TSTOP, E) **Also see:** ODES **Precision:** Single (Double: DODESH) **Availability:** 855NOS, 205.

**ODFISH.** Linear discriminant analysis method of Fisher for reducing the number of variables. *Proprietary Fortran subroutine in IMSL.* **Classes:** L12 **Usage:** CALL ODFISH (*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ODNORM.** Multivariate normal linear discriminant analysis among several known groups. *Proprietary Fortran subroutine in IMSL.* **Classes:** L12 **Usage:** CALL ODNORM (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFCOEF.** Compute a matrix of factor score coefficients for input to IMSL routine OFSCOR. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFCOEF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFCOMM.** Compute an unrotated factor loading matrix according to a common factor model by unweighted or generalized least squares, or by maximum likelihood procedures. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFCOMM (R, NV, NF, IND, NT, IV, MAXIT, MAXTRY, EPS, EPSE, ALPHA, V, A, IA, RI, Y, S, G, IS, WK **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFHARR.** Transformation of unrotated factor loading matrix to oblique axes by Harris-Kaiser method. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFHARR (*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFIMA3.** Least squares solution to the matrix equation AT = B. *Proprietary Fortran subroutine in IMSL.* **Classes:** D9a1, L8f **Usage:** CALL OFIMA3 (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFIMAG.** Compute an unrotated factor loading matrix according to an image model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFIMAG (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFPRI.** Compute an unrotated factor loading matrix according to a principal component model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13b **Usage:** CALL OFPRI (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFPROT.** Oblique transformation of the factor loading matrix using a target matrix, including pivot and power vector options. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFPROT (*21 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFRESI.** Communalities and normalized factor residual correlation matrix calculation. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFRESI (R, NV, NF, A, IA, Y, S, WK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFROTA.** Orthogonal rotation of a factor loading matrix using a generalized orthomax criterion, including quartimax, varimax, and equamax. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFROTA (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFSCHN.** Orthogonal transformation of the factor loading matrix using a target matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFSCHN (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OFSCOR.** Compute a set of factor scores given the factor score coefficient matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13a **Usage:** CALL OFSCOR (*13 parameters*) **Also see:** OFCOEF **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OIND.** Wilks test for the independence of k sets of multi-normal variates. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4e **Usage:** CALL OIND (S, N, IP, K, STAT, WKAREA, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ONEWAY.** Performs one-way analysis of variance, prints standard results, and optionally saves results. *Command in MINITAB proprietary interactive system.* **Classes:** L7a1 **Usage:** ONEWay analysis of variance for data in C, subscripts in C [put residuals into C [fits into C]] **Precision:** Single **Availability:** 855NOS.

**ONEWAY.** Performs one-way ANOVA. Options: descriptive statistics, homogeneity of variance tests, fixed- and random-effects measures, test for trends, a priori contrasts, seven range tests for pairwise comparison, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L7a1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**OPOLY.** Generate orthogonal polynomials with respect to a specified interval and weights. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** C3a, L2a, L8b1b2 **Usage:** CALL OPOLY (*12 parameters*) **Precision:** Single (Double: DOPOLY) **Availability:** 855NOS, 855VE, 205, 840NOS.

**OPRINC.** Principal components of a multivariate sample of observations. *Proprietary Fortran subroutine in IMSL.* **Classes:** L13b **Usage:** CALL OPRINC (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OPTIF0.** Solves the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. Easy to use version of OPTIF9. User has no control of options. *Portable Fortran subroutine in CMLIB (UNCMIN sublibrary).* **Classes:** G1b1a **Usage:** CALL OPTIF0(*10 parameters*) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**OPTIF9.** Solves the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. User has full control of options. *Portable Fortran subroutine in CMLIB (UNCMIN sublibrary).* **Classes:** G1b1a, G1b1b, G1b1c **Usage:** CALL OPTIF9(*22 parameters*) **Precision:** Double **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ORDER.** Sorts in ascending order the values in each of one or more vectors. *Command in MINITAB proprietary interactive system.* **Classes:** N6a2b **Usage:** ORDEr C [and C, ..., C] put into C [and into C, ..., C] **Precision:** Single **Availability:** 855NOS.

**ORTBAK.** Forms eigenvectors of general real matrix from eigenvectors of upper Hessenberg matrix output from ORTHES. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL ORTBAK(NM, LOW, IGH, A, ORT, M, Z) **Also see:** ORTHES **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ORTHES.** Reduces real general matrix to upper Hessenberg form using orthogonal similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b2 **Usage:** CALL ORTHES(NM, N, LOW, IHI, A, ORT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ORTHP.** Evaluates a polynomial expressed as a sum of general orthogonal polynomials. *Proprietary Fortran function in PORT.* **Classes:** C3a **Usage:** R = ORTHP(N, ALPHA, X, A, B, C) **Precision:** Single (Double: DORTHP) **Availability:** 855NOS, 205.

**ORTRAN.** Accumulates orthogonal similarity transformations in reduction of real general matrix by ORTHES. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL ORTRAN(NM, N, LOW, IHI, A, ORT, Z) **Also see:** ORTHES **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**OTMLNR.** Maximum likelihood estimation from grouped and/or censored normal data. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a3 **Usage:** CALL OTMLNR (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**OUT.** Prints a data matrix with the appropriate labels for the cases and variables which is output from many of the subroutines from the CLUSTER sublibrary. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** CALL OUT(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**OWFRQ.** Tally observations into a one-way frequency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L2b **Usage:** CALL OWFRQ (*9 parameters*) **Precision:** Single (Double: DOWFRQ) **Availability:** 855NOS, 855VE, 205, 840NOS.

# P

**P01AAF.** Returns the value of IERROR or terminates the program, printing a failure message. *Proprietary Fortran function in NAG.* **Classes:** R3c **Usage:** I = P01AAF(IFAIL, IERROR, SRNAME) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**P01ABF.** Either returns the value of IERROR (soft failure), or terminates execution of the program (hard failure). Diagnostic messages may be output. *Proprietary Fortran function in NAG.* **Classes:** R3c **Usage:** I = P01ABF(*5 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**P1D.** Prints univariate statistics (mean, standard deviation, standard error of mean, coefficient of variation, extreme values, extreme z-scores, range) for each variable. Options: statistics for each level of each grouping variable, sorting, printing all cases OR only cases with values missing or values outside specified limits. *Proprietary stand-alone program using BMDP command language.* **Classes:** L1a1, L1c1 **Precision:** Single **Availability:** 855NOS.

**P1L.** Estimates survival (time-to-response) distribution of patients who have been observed over varying periods of time by product-limit (Kaplan-Meier) or actuarial life table (Cutler-Ederer) method. Options: three forms of input, Mantel-Cox and Breslow test of equality of survival curves, five plots. *Proprietary stand-alone program using BMDP command language.* **Classes:** L15 **Precision:** Single **Availability:** 855NOS.

**P1M.** Stepwise cluster analysis of variables using one of four measures of similarity, three criteria for combining clusters, with printing of a summary table of clusters, shaded distance measure display, and a tree showing cluster formation. Options: form of input, additional printing and display. *Proprietary stand-alone program using BMDP command language.* **Classes:** L14a1a1 **Precision:** Single **Availability:** 855NOS.

**P1R.** Performs multiple linear regression and prints standard results. Options: weights, form of input, regression on subsets or groups and test of equality of regression lines, intercept term present or absent, more printing, five plots, save predicted values and residuals. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8a1a, L8c1b, L8c2 **Precision:** Single **Availability:** 855NOS.

**P1S.** At each pass through the data, computes univariate statistics (choose means, standard deviations, geometric means, harmonic means, extreme values), and transforms or edits the data using statistics computed in the previous pass. Options: printing, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L2a **Precision:** Single **Availability:** 855NOS.

**P1T.** Interactive or batch spectral analysis of one or two time series, with estimates of spectral density and coherence between variables. Options: print, plot (variable vs. time, lagged plots, complex demodulation, periodogram), handle missing values, remove seasonal means and linear trend, filtering, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L3a6, L3a7, L3b3c, L3e3c, L10a3, L10b3 **Precision:** Single **Availability:** 855NOS.

**P1V.** Performs one-way ANOVA or ANCOVA with standard results. For ANCOVA, tests 1) equality of slopes, 2) zero slope, and 3) equality of adjusted cell means; plots the covariate for each group. Tests equality of pairs of means (or adjusted means). Options: linear contrasts, within-group correlations and statistics. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7a1 **Precision:** Single **Availability:** 855NOS.

**P2D.** For each variable, prints frequency and percent for each distinct value; mean, median, mode, standard deviation, standard errors of mean and median, skewness, kurtosis, half interquartile range; histogram, and stem-and-leaf plot. Options: initially round or truncate, three robust location estimates. *Proprietary stand-alone program using BMDP command language.* **Classes:** L1a1, L1c1 **Precision:** Single **Availability:** 855NOS.

**P2L.** Analyzes survival data with covariates using Cox proportional hazard regression model. Options: two forms of input, stepwise selection of covariates, time-dependent covariates, stratification, significance tests, three plots, print survival functions and residuals. *Proprietary stand-alone program using BMDP command language.* **Classes:** L15 **Precision:** Single **Availability:** 855NOS.

**P2M.** Stepwise cluster analysis of cases (observations) using one of four distance measures (including Euclidean and one for data that are frequency counts) and three linkage algorithms (single, centroid, k nearest neighbors), with a summary table of clusters and a cluster tree. Options: weights, standardized data. *Proprietary stand-alone program using BMDP command language.* **Classes:** L14a1a1 **Precision:** Single **Availability:** 855NOS.

**P2R.** Multiple linear regression, with standard results. Options: weights, form of input, forward or backward stepping, interactive stepping, stepping sets of variables (e.g. design variables), forcing variables into the model, eleven diagnostics (including Cook and AP statistics) available for printing, plotting, and saving. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8c1a1, L8c1a2 **Precision:** Single **Availability:** 855NOS.

**P2T.** Interactive or batch Box-Jenkins time series analysis for univariate time domain models (including ARIMA, regression, intervention, and transfer function models) – model identification, parameter estimation, testing, forecasting. Options: print, plot, differencing and filtering, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L10a2, L10b2 **Precision:** Single **Availability:** 855NOS.

**P2V.** ANOVA and ANCOVA for unbalanced fixed-effect models (including full and fractional factorial designs, Latin squares), and repeated measure models, or a combination of models, with Greenhouse-Geisser and Huynh-Feldt degree of freedom adjustment. Options: orthogonal decomposition of within-effects, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7d **Precision:** Single **Availability:** 855NOS.

**P3D.** One-sample t-test to test if one group mean is zero (e.g., matched pairs); two-sample t test with and without equal variances assumption, Levene's test for equal variances, histograms. Options: trimmed t test, Hotelling's T-squared and Mahalanobis' D-squared, within-group correlations, data listing. *Proprietary stand-alone program using BMDP command language.* **Classes:** L4a1a4n, L4b1a4, L4c1a **Precision:** Single **Availability:** 855NOS.

**P3M.** Forms blocks (submatrices of the data matrix) where a subset of the cases (for a subset of the variables) cluster together, with printing of the blocks and tree diagrams for cases and for variables – appropriate for categorical data with few levels. *Proprietary stand-alone program using BMDP command language.* **Classes:** L14b **Precision:** Single **Availability:** 855NOS.

**P3R.** Performs nonlinear least squares regression with standard results. Six functions are built in; others can be specified. Options: weights, evaluates functions of parameters (with standard errors), upper and lower limits on parameters, ridging, exact linear constraints, maximum likelihood estimates, and five plots. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8e1b2, L8e1b4, L8e2 **Precision:** Single **Availability:** 855NOS.

**P3S.** Computes and prints results from one or more of the following: sign test, Wilcoxon signed-rank test, Mann-Whitney rank-sum test, Kruskall-Wallis one-way ANOVA, Friedman two-way ANOVA, Kendall's coefficient of concordance, Kendall and Spearman rank-correlation coefficients. *Proprietary stand-alone program using BMDP command language.* **Classes:** L4a1b1, L4b1b, L4b3, L4c1b, L7a2 **Precision:** Single **Availability:** 855NOS.

**P3V.** Uses maximum likelihood (ML) and restricted ML approaches to balanced and unbalanced fixed and random coefficient models of quite arbitrary form (including having covariates), with parameter estimation, hypothesis testing, and printing. Weights optional. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7d **Precision:** Single **Availability:** 855NOS.

**P4D.** Counts frequency of each number, letter, or symbol in single-column fields (A1 format). Options: input case label variables in A4 format, diagnostic printing useful in preliminary data screening. Specified characters may be replaced by blanks or symbols. *Proprietary stand-alone program using BMDP command language.* **Classes:** L1a1d, L1c1d, N3 **Precision:** Single **Availability:** 855NOS.

**P4F.** Forms two- or multi-way frequency tables. Options: percents; 25 tests and measures for two-way tables; fits and tests log-linear models, tests of marginal and partial association, stepwise models, three forms of input, structural zeros, cell and strata deletion, residuals. *Proprietary stand-alone program using BMDP command language.* **Classes:** L9 **Precision:** Single **Availability:** 855NOS.

**P4M.** Provides four methods of initial factor extraction from a correlation or covariance matrix, and several methods of rotation, prints shaded correlations, factor loadings, factor score coefficients, factor scores, Mahalanobis distances, and plots. Options: weights, form of input, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L13a **Precision:** Single **Availability:** 855NOS.

**P4MED.** Computes the geometric median of a finite set S of points X belonging to n dimensional space. The geometric median is that point G such that the sum of $(X\text{-}G)^2$ over all points is less than the sum of $(X\text{-}Y)^2$ for all Y in the space. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** P **Usage:** CALL P4MED(*12 parameters*) **Precision:** Double **Availability:** PC.

**P4R.** Regression analysis for a dependent variable on a set of principal components computed from the independent variables in a stepwise manner determined either by magnitude of eigenvalue or correlations between dependent variable and components, with printing. Options: form of input, more printing, four plots. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8c1a3 **Precision:** Single **Availability:** 855NOS.

**P4V.** Interactive or batch univariate and multivariate ANOVA and ANCOVA, including nested, repeated measures, split-plot, and changeover designs, and model building features. Options: cell weights for hypothesis testing, contrasts, tests of simple effects, save cell means. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7d, L7e **Precision:** Single **Availability:** 855NOS.

**P5D.** Prints histograms with frequencies and percentages, normal and detrended normal probability plots, halfnormal plots, cumulative frequency distribution plots, and cumulative histograms for ungrouped data or for grouped data – either separately or combined in one plot. Plot options. *Proprietary stand-alone program using BMDP command language.* **Classes:** L3a1, L3a2, L3b1, L3b2, L3e1, L3e2, L4a1a2h, L4a1a2n **Precision:** Single **Availability:** 855NOS.

**P5R.** Least squares fit of a polynomial in one independent variable to the dependent variable. Prints standard results and goodness-of-fit statistics for each polynomial degree. Computations use orthogonal polynomials. Options: weights, additional printing, and three plots. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8b1b2 **Precision:** Single **Availability:** 855NOS.

**P6D.** Bivariate (scatter) plots. Options: several variables, or subsets of one variable (symbols identify group membership), on the same plot; prints correlation and linear regression statistics (line is marked on plot frame); user control for plot size, scales, and symbols. *Proprietary stand-alone program using BMDP command language.* **Classes:** L3b3a, L3e3a **Precision:** Single **Availability:** 855NOS.

**P6M.** Computes canonical correlation analysis for two sets of variables and Bartlett's test for the significance of the remaining eigenvalues, with printing. Options: weights, form of input, additional printing and plotting, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L13c **Precision:** Single **Availability:** 855NOS.

**P6R.** Computes the partial correlations of a set of variables after removing the linear effects of a second set of variables. Can be used for regression, especially if multiple dependent variables are present. Prints standard results. Options: weights, form of input, additional printing and plots. *Proprietary stand-alone program using BMDP command language.* **Classes:** L11, L8f **Precision:** Single **Availability:** 855NOS.

**P7D.** Side-by-side histograms for each cell in one-way or two-way ANOVA, within-group summary statistics and ANOVA table (with equality of variance test and tests that do not assume equal variances). Options: trimmed mean analysis, ANOVA diagnostics, tests of pairwise mean comparisons, correlations, Winsorized means. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7a, L7b **Precision:** Single **Availability:** 855NOS.

**P7M.** Stepwise forward or backward discriminant analysis (including jackknifed classifications, percent correctly classified, Mahalanobis' distances, canonical variable coefficients, eigenvalues, scores, and plot of first two canonical variables). Options: interactive stepping, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L12 **Precision:** Single **Availability:** 855NOS.

**P8D.** Four methods to compute covariance and correlation matrices when data contain missing values or values out of range. Options: weights, summary statistics, save results, pairwise t-tests based on the pattern of incomplete data. *Proprietary stand-alone program using BMDP command language.* **Classes:** L1c2 **Precision:** Single **Availability:** 855NOS.

**P8M.** Boolean factor analysis of binary (dichotomous) data. Options: initial estimates of the loading matrix, printing, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L13a **Precision:** Single **Availability:** 855NOS.

**P8V.** ANOVA for complete designs with equal cell sizes – nested, crossed, partially nested, partially crossed designs for fixed-effect models, mixed models (including repeated measures), and random-effect models, with parameter estimation and printing. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7d1 **Precision:** Single **Availability:** 855NOS.

**P9D.** Provides descriptive statistics (means, standard deviations, frequencies, one-way ANOVA table) of groups (cells) for data classified into cells using one or more grouping variables. Options: miniplots of cell means (eight per page), plot frames are defined by combinations of levels of grouping variables. *Proprietary stand-alone program using BMDP command language.* **Classes:** L7g **Precision:** Single **Availability:** 855NOS.

**P9M.** Scoring based on preference pairs – for each observation construct score as linear combination of variables with coefficients based on expert preference, in stepwise manner. Options: printing, plots, compare results when analysis is repeated for different judges. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8c1a3 **Precision:** Single **Availability:** 855NOS.

**P9R.** Estimates regression equations for "best" (by R-squared, adjusted R-squared, or Mallows' C(p) criterion) subset of predictor variables by Furnival-Wilson algorithm. Options: weights, form of input, Durbin-Watson statistic. Cook's distance and several types of residuals may be printed, plotted, or saved. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8c1a1, L8c1a2 **Precision:** Single **Availability:** 855NOS.

**PACF.** Compute the sample partial autocorrelation function of a stationary time series. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2a2 **Usage:** CALL PACF (MAXLAG, AC, PAC) **Precision:** Single (Double: DPACF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PACF.** Computes and graphs partial autocorrelations of a time series and optionally saves results. *Command in MINITAB proprietary interactive system.* **Classes:** L10a2a2 **Usage:** PACF [up to K lags] for series in C [put into C] **Precision:** Single **Availability:** 855NOS.

**PAGE.** Set or retrieve page width and length for printing. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N1 **Usage:** CALL PAGE (IOPT, IPAGE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**PAIRS.** Perform a pairs test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1d **Usage:** CALL PAIRS (*11 parameters*) **Precision:** Single (Double: DPAIRS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PAM.** Describes pattern of invalid values (missing or out of range) for multivariate data. Options: weights, grouping, estimates covariance and correlation matrices by one of three methods (including maximum likelihood), replace invalid values using means or one of several regression procedures, plots, save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L1c2 **Precision:** Single **Availability:** 855NOS.

**PAR.** Performs nonlinear regression using pseudo-Gauss-Newton algorithm. Derivatives are NOT specified. Options: weights, linear inequality constraints, maximum likelihood, functions of parameters, ridging, four plots, fitting models defined by differential equations. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8e1b1, L8e1b3, L8e2 **Precision:** Single **Availability:** 855NOS.

**PARCDF.** Computes the cumulative distribution function value for the Pareto distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1p **Usage:** CALL PARCDF(X, GAMMA, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PARETO PLOT.** Generate a Pareto plot with horizontal axis an index variable and vertical axis typically plotted as a bar or a spike with a labelled character, and ordered from the largest to the smallest value. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a4 **Usage:** PARETO PLOT <VARIABLE> **Also see:** CHARACTERS, CHARACTER OFFSET, CHARACTER ANGLE **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**PARETO PPCC PLOT.** Generates a probability plot correlation coefficient plot for the Pareto distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging from 1 to 100, or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3p **Usage:** PARETO PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PARETO PROBABILITY PLOT.** Generates a probability plot for the Pareto distribution with tail length parameter $\gamma$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2p **Usage:** PARETO PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PARPLT.** Generates a Pareto probability plot with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2p **Usage:** CALL PARPLT(X, N, GAMMA) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PARPPF.** Computes the percent point function value for the Pareto distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a2p **Usage:** CALL PARPPF(P, GAMMA, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PARRAN.** Generates a random sample of size N from the Pareto distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L6a16 **Usage:** CALL PARRAN(N, GAMMA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PARTIAL CORR.** Produces a matrix of partial correlation coefficients describing the relationship between two variables, adjusted for effects of additional variables. Options: one- or two-tailed significance levels, univariate statistics, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L11 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**PCGRC.** Solve a real symmetric definite linear system using a preconditioned conjugate gradient method with reverse communication. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D2b4 **Usage:** CALL PCGRC (*8 parameters*) **Precision:** Single (Double: DPCGRC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PCHEV.** Evaluates piecewise cubic polynomial and its derivative at NVAL points in array XVAL given N-arrays X,F,D; results are put in arrays FVAL and DVAL. Usually used following a call to PCHEZ, but can be used independently. *Portable Fortran software in NMS library.* **Classes:** E3a1, E3a2 **Usage:** CALL PCHEV(*9 parameters*) **Also see:** PCHEZ **Precision:** Single (Double: DPCHEV) **Availability:** PC.

**PCHEZ.** Finds either spline or visually-pleasing piecewise cubic interpolant to input N-arrays X,F. Evaluate resulting function with PCHEV; integrate it with PCHQA. *Portable Fortran software in NMS library.* **Classes:** E1a **Usage:** CALL PCHEZ(N, X, F, D, SPLINE, W, LW, IERR) **Also see:** PCHEV **Precision:** Single (Double: DPCHEZ) **Availability:** PC.

**PCHFD.** Evaluates a piecewise cubic Hermite function and its first derivative at an array of points. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E3a1, E3a2 **Usage:** CALL PCHFD(*11 parameters*) **Precision:** Single (Double: DPCHFD) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHFE.** Evaluates a piecewise cubic Hermite function at an array of points. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E3a1 **Usage:** CALL PCHFE(*10 parameters*) **Precision:** Single (Double: DPCHFE) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHIA.** Evaluates the definite integral of a piecewise cubic Hermite function over an arbitrary interval. *Portable Fortran function in CMLIB (PCHIPS sublibrary).* **Classes:** E3a3 **Usage:** R = PCHIA(*9 parameters*) **Precision:** Single (Double: DPCHIA) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHIC.** Determines a piecewise monotone, piecewise cubic Hermite interpolant to given data. User control is available over boundary conditions and/or treatment of points where monotonicity switches direction. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E1a **Usage:** CALL PCHIC(*11 parameters*) **Also see:** PCHFE for evaluation. See package documentation for other facilities. **Precision:** Single (Double: DPCHIC) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHID.** Evaluates the definite integral of a piecewise cubic Hermite function over an interval whose endpoints are data points. *Portable Fortran function in CMLIB (PCHIPS sublibrary).* **Classes:** E3a3 **Usage:** R = PCHID(*9 parameters*) **Precision:** Single (Double: DPCHID) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHIM.** Determines a monotone piecewise cubic Hermite interpolant to given data. Default boundary values are provided which are compatible with monotonicity. The interpolant will have an extremum at each point where monotonicity switches direction. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E1a

**Usage:** CALL PCHIM(N, X, F, D, INFD, IERR) **Also see:** PCHFE for evaluation. See package documentation for other facilities. **Precision:** Single (Double: DPCHIM) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHMC.** Checks a cubic Hermite function for monotonicity. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E3d **Usage:** CALL PCHMC(*8 parameters*) **Precision:** Single (Double: DPCHMC) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PCHQA.** Integrates piecewise cubic from A to B given N-arrays X,F,D. Usually used in conjunction with PCHEZ to form cubic, but can be used independently, especially if the abscissas are equally spaced. *Portable Fortran software in NMS library.* **Classes:** E3a3, H2a1b2, K6a3 **Usage:** CALL PCHQA(N, X, F, D, A, B, IERR) **Also see:** PCHEZ **Precision:** Single (Double: DPCHQA) **Availability:** PC.

**PCHSP.** Determines the cubic spline interpolant to given data. User has control over boundary conditions. *Portable Fortran subroutine in CMLIB (PCHIPS sublibrary).* **Classes:** E1a **Usage:** CALL PCHSP(*10 parameters*) **Also see:** PCHFE for evaluation. See package documentation for other facilities. **Precision:** Single (Double: DPCHSP) **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PDECOL.** Solves general nonlinear systems of initial-boundary-value problems in one space dimension with general boundary conditions. Spatial derivatives may be of at most second order. Uses method of lines based on collocation of B-spline basis functions. *Fortran subroutine in PDELIB (PDECOL sublibrary).* **Classes:** I2a1a **Usage:** CALL PDECOL(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 205.

**PDETWO.** Solves general nonlinear systems of initial-boundary-value problems in two spatial dimensions with quasi-linear boundary conditions. Uses the method of lines based upon finite differences on a user-specified rectangular mesh. *Fortran subroutine in PDELIB (PDETWO sublibrary).* **Classes:** I2a1b **Usage:** CALL DRIVEP(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 205.

**PDM.** An interactive data manipulation system which transforms (re-expresses data, selects cases, edits data, and computes various results), sorts, checks data (for minimum, maximum, missing, etc.), reports summary information including histograms, prints graphical displays, constructs new files from existing files in several ways, and constructs new records from existing records in several ways. The Data Manager can be used with both BMDP files and raw data files. *Proprietary stand-alone program using BMDP command language.* **Classes:** L2 **Precision:** Single **Availability:** 855NOS.

**PEARSON CORR.** Produces matrices of Pearson product-moment correlation coefficients with one- or two-tailed significance levels. Options: univariate statistics, covariances, cross-product deviations, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1c1b, L4c1a **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**PERARS.** Fits periodic autoregressive models by the method of least squares realized through the Householder transformation. The outputs are the estimates of the regression coefficients and innovation variance of the periodic AR-model for each instant. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2c **Precision:** Single **Availability:** 855NOS.

**PERCENT POINT PLOT.** Generates a percent point plot (cumulative distribution plot with axes interchanged). Optional control of upper and lower limits and/or class width. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a2 **Usage:** PERCENT POINT PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PERIODOGRAM.** Generates a frequency vs. power plot for equi-spaced univariate time series data. Alternate names: HARMONIC PLOT, FOURIER PLOT. *Command(s) in DATAPLOT interactive system.* **Classes:** L10a3a2 **Usage:** PERIODOGRAM <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PERMA.** Permute the rows or columns of a matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N8 **Usage:** CALL PERMA (*8 parameters*) **Precision:** Single (Double: DPERMA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PERMU.** Rearrange the elements of an array as specified by a permutation. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N8 **Usage:** CALL PERMU (N, X, IPERMU, IPATH, XPERMU) **Precision:** Single (Double: DPERMU) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PFFT.** Compute the periodogram of a stationary time series using a fast Fourier transform. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a3a2 **Usage:** CALL PFFT (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**PFQAD.** Computes integral on (X1,X2) of product of function and the ID-th derivative of B-spline which is in piecewise polynomial representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a3, H2a2a1, K6a3 **Usage:** CALL PFQAD(*12 parameters*) **Precision:** Single (Double: DPFQAD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**PGM.** Compute and print a periodogram analysis of a series; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL PGM (YFFT, N, LYFFT, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PGMS.** Compute and optionally print a periodogram analysis of a series; use FFT for computations; return periodogram and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a2 **Usage:** CALL PGMS (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PIE CHART.** Generates a pie chart for presentation rather that analysis purposes. Optional control of upper and lower limits and/or class widths. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a5 **Usage:** PIE CHART [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PKM.** By k-means procedure, partitions a set of cases (observations) into k clusters – beginning with user-specified initial clusters or one cluster, proceeding in divisive stepwise manner, then doing iterative reallocation – prints cluster profile and plot. Options: weights, standardize data (four ways), save results. *Proprietary stand-alone program using BMDP command language.* **Classes:** L14a1b **Precision:** Single **Availability:** 855NOS.

**PLAN.** Generates random permutations of positive integers for experimental plans (e.g., completely random, split-plot, and hierarchical designs) given a specification of the randomized plan including number of levels of nesting. Option: seed of first permutation. *Proprietary stand-alone program using SAS command language.* **Classes:** L6b5, L7f **Precision:** Single **Availability:** CAMVAX.

**PLOD.** An easy to use interactive system on a personal computer for the solution of initial value problems for ordinary differential equations. The user can change initial conditions, interval, parameters etc., and examine various plots on the terminal. Little programming needed. *PLOD is an interactive system.* **Classes:** I1a **Precision:** Single **Availability:** PC.

**PLOT.** Produces a scatter plot of several Y variables vs. one X variable. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e3a **Usage:** CALL PLOT(*5 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PLOT.** Yields a one-page printer plot of $Y_i$ versus $X_i$. *Fortran subroutine in DATAPAC.* **Classes:** L3b3a **Usage:** CALL PLOT (Y, X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOT.** Plots a single or multi-trace plot of data, functions, or both. Built-in function library includes elementary, trigonometric, logarithmic, and probability functions. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a6, L3b3a, L3e3a, L3e3c **Usage:** PLOT <RESPONSE VARIABLE> [<TRACE VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PLOT.** Prints a scatter diagram, with optional scale specification. *Command in MINITAB proprietary interactive system.* **Classes:** L3b3a **Usage:** PLOT y in C [from K to K] vs x in C [from K to K] **Precision:** Single **Availability:** 855NOS.

**PLOT.** Produces a Y vs. X line printer scatterplot, a superimposed plot, or a contour plot. Options: missing values, user control of plot features. *Proprietary stand-alone program using SAS command language.* **Classes:** L3b3a, L3c, L3e3a **Precision:** Single **Availability:** CAMVAX.

**PLOT.** Produces two-dimensional line-printer plots. Options: format (bivariate scatterplot, contour, overlay, regression), labels, scales, size, symbols, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L3b3a, L3c, L3e3a **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**PLOT10.** Yields a one-page printer plot of $Y_i$ versus $X_i$ for a subset of the data, with special plot characters, and with specified axis limits and labels. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOT10(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOT10.** A set of Fortran subprograms for two-dimensional general purpose computer graphics. Output is supported only on Tektronix terminals. Capabilities include: 1) low-level plotting calls such as move and draw (TCS), 2) higher level calls such as drawing bar plots (AGII), and 3) Calcomp preview routines. The latter are standard Calcomp library calls which allow users to preview pen plotter output on Tektronix terminals. *PLOT10 proprietary Fortran subprogram library.* **Classes:** Q **Precision:** Single **Availability:** 855NOS, 855VE.

**PLOT6.** Yields a one-page printer plot of $Y_i$ versus $X_i$ for specified axis limits. *Fortran subroutine in DATAPAC.* **Classes:** L3b3a **Usage:** CALL PLOT6(Y, X, N, YMIN, YMAX, XMIN, XMAX) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOT7.** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plot characters and for specified axis limits. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOT7(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOT8.** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plot characters for a subset of the data with specified axis limits. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOT8(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOT9.** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plot characters and for specified axis limits and axis labels. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOT9(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTC.** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special plotting characters. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOTC(Y, X, CHAR, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTCT.** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$ with special plotting characters. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOTCT(Y, X, CHAR, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTP.** Print a plot of up to ten sets of points. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** Q **Usage:** CALL PLOTP (*11 parameters*) **Precision:** Single (Double: DPLOTP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PLOTP.** Print a plot of up to ten sets of points. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3e3a **Usage:** CALL PLOTP (*11 parameters*) **Precision:** Single (Double: DPLOTP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PLOTS.** Yields a one-page printer plot of $Y_i$ versus $X_i$ for a subset of the data. *Fortran subroutine in DATAPAC.* **Classes:** L3b3a **Usage:** CALL PLOTS(Y, X, N, D, DMIN, DMAX) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTSC.** Yields a one-page printer plot of $Y_i$ versus $X_i$ with special characters for a subset of the data. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLOTSC(Y, X, CHAR, N, D, DMIN, DMAX) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTST.** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$ for a subset of the data. *Fortran subroutine in DATAPAC.* **Classes:** L3b3a **Usage:** CALL PLOTST(Y, X, N, D, DMIN, DMAX) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTT.** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$. *Fortran subroutine in DATAPAC.* **Classes:** L3b3a **Usage:** CALL PLOTT(Y, X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTU.** Produces 4 plots: data plot ($X_i$ vs. $i$), autoregression plot ($X_i$ vs. $X_{i-1}$), histogram, and normal probability plot. *Fortran subroutine in DATAPAC.* **Classes:** L3a1, L3a6, L3a7, L4a1a2n **Usage:** CALL PLOTU(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTX.** Yields a one-page printer plot of $X_i$ vs. $i$. *Fortran subroutine in DATAPAC.* **Classes:** L3a6 **Usage:** CALL PLOTX(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTXT.** Yields a narrow-width (71-character) plot of $X_i$ vs. $i$. *Fortran subroutine in DATAPAC.* **Classes:** L3a6 **Usage:** CALL PLOTXT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLOTXX.** Yields a one-page printer plot of $X_i$ versus $X_{i-1}$ for testing autocorrelation. *Fortran subroutine in DATAPAC.* **Classes:** L3a7 **Usage:** CALL PLOTXX(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLR.** Stepwise logistic regression for binary dependent variable and categorical (design variables are formed) and continuous independent variables, using either maximum likelihood or approximate asymptotic estimates for stepping. Three options for generating design variables, plots, interactive stepping. *Proprietary stand-alone program using BMDP command language.* **Classes:** L8e1a **Precision:** Single **Availability:** 855NOS.

**PLTSCT.** Yields a narrow-width (71-character) plot of $Y_i$ versus $X_i$ with special plot characters and a a subset of the data. *Fortran subroutine in DATAPAC.* **Classes:** L3b3b **Usage:** CALL PLTSCT(Y, X, CHAR, N, D, DMIN, DMAX) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PLTXXT.** Yields a narrow-width (71-character) plot of $X_i$ versus $X_{i-1}$ for testing autocorrelation. *Fortran subroutine in DATAPAC.* **Classes:** L3a7 **Usage:** CALL PLTXXT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PMANY.** Produces a matrix of scatter plots between $Y_i$ and $Y_j$ for for i and j between 1 and n where n is the number of variables. For i=j, a histogram for variable i is produced. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e3d **Usage:** CALL PMANY(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**POCH.** Pochhammer's symbol. Input and output are real. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** R = POCH(A, X) **Precision:** Single (Double: DPOCH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**POCH.** Generalization of Pochhammer's symbol. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** R = POCH(A, X) **Precision:** Single (Double: DPOCH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**POCH1.** Pochhammer's symbol from first order. Input and output are real. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7a **Usage:** R = POCH1(A, X) **Precision:** Single (Double: DPOCH1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**POCH1.** Generalization of Pochhammer's symbol starting from first order. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7a **Usage:** R = POCH1(A, X) **Precision:** Single (Double: DPOCH1) **Availability:** 855NOS, 855VE, 205, 840NOS.

**POICDF.** Computes the cumulative distribution function value at X for the Poisson distribution with tail length parameter ALAMBA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1p **Usage:** CALL POICDF(X, ALAMBA, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**POIDF.** Evaluate the Poisson distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and*

*IMSL SFUN/LIBRARY.* **Classes:** L5a1p **Usage:** R = POIDF(K, THETA) **Precision:** Single (Double: DPOIDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**POIES.** Estimate the parameter of the Poisson distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1a4p **Usage:** CALL POIES (*6 parameters*) **Precision:** Single (Double: DPOIES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**POIPLT.** Generates a Poisson probability plot with tail length parameter ALAMBA, mean ALAMBA and standard deviation $\sqrt{\text{ALAMBA}}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2p **Usage:** CALL POIPLT(X, N, ALAMBA) **Precision:** Single **Availability:** 855NOS, 840NOS.

**POIPPF.** Computes the percent point function value at P for the Poisson distribution with mean ALAMBA and standard deviation $\sqrt{\text{ALAMBA}}$. *Fortran subroutine in DATAPAC.* **Classes:** L5a2p **Usage:** CALL POIPPF(P, ALAMBA, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**POIPR.** Evaluate the Poisson probability function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1p **Usage:** R = POIPR(K, THETA) **Precision:** Single (Double: DPOIPR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**POIRAN.** Generates a random sample of size N from the Poisson distribution with mean ALAMBA and standard deviation $\sqrt{\text{ALAMBA}}$. *Fortran subroutine in DATAPAC.* **Classes:** L6a16 **Usage:** CALL POIRAN(N, ALAMBA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**POIS3D.** Solves block tridiagonal linear systems of algebraic equations arising from the discretization of separable elliptic partial differential equations in 3D. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** D2a4, I2b4b **Usage:** CALL POIS3D(*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**POISSON.** Prints table of Poisson probabilities and cumulative distribution function. *Command in MINITAB proprietary interactive system.* **Classes:** L5a1p **Usage:** POISson probabilities for mean K **Precision:** Single **Availability:** 855NOS.

**POISSON PPCC PLOT.** Generates a probability plot correlation coefficient plot for the Poisson distribution (plot of probability plot correlation coefficient vs. the tail parameter $\lambda$ for $\lambda$ ranging from 1 to 100 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3p **Usage:** POISSON PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**POISSON PROBABILITY PLOT.** Generates a probability plot for the Poisson distribution with tail length parameter $\lambda$, mean $\lambda$ and standard deviation $\sqrt{\lambda}$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2p **Usage:** POISSON PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**POISTG.** Solves block tridiagonal linear systems of algebraic equations arising from the discretization of separable elliptic partial differential equations. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** D2a4, I2b4b **Usage:** CALL POISTG(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**POLRG.** Evaluate a real general matrix polynomial. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b7 **Usage:** CALL POLRG (*7 parameters*) **Precision:** Single (Double: DPOLRG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**POLY.** Computes a least squares polynomial fit (of degree IDEG) of the response variable data in the vector Y as a function of vector X and with optional weights. *Fortran subroutine in DATAPAC.* **Classes:** L8b1b1 **Usage:** CALL POLY(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PP.** Print Y versus X scatterplot; linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3a **Usage:** CALL PP (Y, X, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PPC.** Print Y versus X scatterplot; log or linear axes; user-supplied control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3a **Usage:** CALL PPC (*10 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PPDER.** Evaluate the derivative of a piecewise polynomial. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a2, K6a2 **Usage:** R = PPDER(*6 parameters*) **Precision:** Single (Double: DPPDER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PPITG.** Evaluate the integral of a piecewise polynomial. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a3, K6a3 **Usage:** R = PPITG(*6 parameters*) **Precision:** Single (Double: DPPITG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PPL.** Print Y versus X scatterplot; log or linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3a **Usage:** CALL PPL (Y, X, N, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PPM.** Print Y versus X scatterplot; linear axes; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3a **Usage:** CALL PPM (Y, YMISS, X, XMISS, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PPMC.** Print Y versus X scatterplot; log or linear axes; user-supplied control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3a **Usage:** CALL PPMC (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PPML.** Print Y versus X scatterplot; log or linear axes; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3a **Usage:** CALL PPML (Y, YMISS, X, XMISS, N, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**PPQAD.** Computes the integral of a B-spline from X1 to X2. The B-spline must be in piecewise polynomial representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a3, H2a2b1, K6a3 **Usage:** CALL PPQAD(LDC, C, XI, LXI, K, X1, X2, PQUAD) **Precision:** Single (Double: DPPQAD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**PPVAL.** Calculates (at X) the value of the IDERIV-th derivative of the B-spline from its piecewise polynomial representation. *Portable Fortran subroutine in CMLIB (BSPLINE sublibrary).* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL PPVAL(*8 parameters*) **Precision:** Single (Double: DPPVAL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**PPVAL.** Evaluate a piecewise polynomial. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E3a1, K6a1 **Usage:** R = PPVAL(*5 parameters*) **Precision:** Single (Double: DPPVAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PRANDOM.** Generates K pseudo-random numbers from the Poisson distribution with specified population mean K. *Command in MINITAB proprietary interactive system.* **Classes:** L6a16 **Usage:** PRANdom K Poisson observations with population mean = K, put into C **Precision:** Single **Availability:** 855NOS.

**PRE-FIT.** Fits a nonlinear model (including linear and polynomial models) to data over a specified discrete lattice of parameter values by least squares, least absolute value, and, optionally, other $L_p$ criteria. For the best and worst set parameter values for each criterion, output includes coefficients and their standard deviations, predicted values, residuals, residual standard deviation, residual degrees of freedom, and lack-of-fit analysis if there is replication. *Command(s) in DATAPLOT interactive system.* **Classes:** L8c3, L8e1b1 **Usage:** PRE-FIT <RESPONSE VARIABLE> = <FUNCTIONAL EXPRESSION OR FUNCTION NAME> FOR <QUALIFICATION FOR EACH PARAMETER> (OR FOR POLYNOMIAL MODELS, BEGIN COMMAND WITH DEGREE OF POLYNOMIAL, E.G., QUADRATIC PRE-FIT <RESPONSE VARIABLE>=...) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PRIME.** Decompose an integer into its prime factors. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** B **Usage:** CALL PRIME (N, NPF, IPF, IEXP, IPW) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**PRINC.** Compute principal components from a variance-covariance matrix or a correlation matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L13b **Usage:** CALL PRINC (*12 parameters*) **Precision:** Single (Double: DPRINC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PRINCOMP.** Performs principle component analysis on raw data, a correlation matrix, or a covariance matrix. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L13b **Precision:** Single **Availability:** CAMVAX.

**PROBIT.** Calculates maximum-likelihood estimates of the intercept, slope and natural (threshold) response rate for biological assay data using a modified Gauss-Newton algorithm. *Proprietary stand-alone program using SAS command language.* **Classes:** L8e1a **Precision:** Single **Availability:** CAMVAX.

**PROBIT.** Estimates the effects of one or more independent variables on a dichotomous dependent variable by maximum likelihood. Options: form of response model (probit and/or logit), logarithmic transformation, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L8e1a **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**PROBP.** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1a2e, L4a1a2h, L4a1a2l, L4a1a2n, L4a1a2w **Usage:** CALL PROBP (NOBS, N1, N2, X, IDIST) **Precision:** Single (Double: DPROBP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PROF.** Computes and outputs profiles of the variables such that the linear regression lines through each case have the smallest total error. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e4 **Usage:** CALL PROF(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**PROFILE PLOT.** Generate a profile plot. *Command(s) in DATAPLOT interactive system.* **Classes:** L3e4 **Usage:**

PROFILE PLOT <VARIABLE 1> ... <VARIABLE K> SUBSET <VARIABLE> <SUBSET VALUE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**PROPOR.** Computes the sample proportion which is the proportion of data between XMIN and XMAX (inclusively) in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1d **Usage:** CALL PROPOR(X, N, XMIN, XMAX, IWRITE, XPROP) **Precision:** Single **Availability:** 855NOS, 840NOS.

**PRPFT.** Perform iterative proportional fitting of a contingency table using a loglinear model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L9c **Usage:** CALL PRPFT (*9 parameters*) **Precision:** Single (Double: DPRPFT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PSI.** Psi (digamma) of real argument. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C7c **Usage:** R = PSI(X) **Precision:** Single (Double: DPSI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**PSI.** Logarithmic derivative of the gamma function. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C7c **Usage:** R = PSI(X) **Precision:** Single (Double: DPSI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**PSIFN.** Derivatives of the Psi function. *Portable Fortran subroutine in CMLIB (AMOSLIB sublibrary).* **Classes:** C7c **Usage:** CALL PSIFN(X, N, KODE, M, ANS, NZ, IERR) **Precision:** Single (Double: DPSIFN) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**PST3DI.** Initialization routine for PSTG3D. *Portable Fortran subroutine in CMLIB (VHS3 sublibrary).* **Usage:** CALL PST3DI(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**PSTG3D.** Solves certain block tridiagonal systems of linear algebraic equations that arise in finite difference approximations to three-dimensional Helmholtz equations on a staggered grid. *Portable Fortran subroutine in CMLIB (VHS3 sublibrary).* **Classes:** I2b4b **Usage:** CALL PSTG3D(LDIMF, MDIMF, F, W) **Also see:** PST3DI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**PSVD.** Computes in an efficient and reliable way a basis for the left and/ or right singular subspace of a matrix corresponding to its smallest singular values. The dimension of the desired subspace may be given or may depend on a given upper bound for those smallest singular values. From: "An Efficient and Reliable Algorithm for Computing the Singular Subspace of a Matrix Associated with its Smallest Singular Values", by S. Van Huffel, J. Vandewalle, and A. Haegemans, J. Comp. Appl. Math. 19, (1987) 313-330. *Portable Fortran software in JCAM library.* **Classes:** D6 **Usage:** CALL PSVD(*18 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**PTLS.** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution, and is more efficient than the author's routine DTLS. From: "The Partial Total Least Squares Algorithm", by S. Van Huffel and J. Vandewalle, J. Comp. Appl. Math. 21, (1988) 333-341. *Portable Fortran software in JCAM library.* **Classes:** D9a1, D9a4 **Usage:** CALL PTLS(*18 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**PUMB.** Given a basic mesh, this subdivides each interval into a uniform but variable number of points. Multiplicities can occur. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL PUMB (XB, NXB, NA, K, X, NX) **Precision:** Single (Double: DPUMB) **Availability:** 855NOS, 205.

**PUMD.** Given a basic mesh, this subdivides each interval into a uniform but variable number of points. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL PUMD (XB, NXB, NA, X, NX) **Precision:** Single (Double: DPUMD) **Availability:** 855NOS, 205.

# Q

**Q1DA.** Automatic integration of a user-defined function of one variable. Special features include randomization and singularity weakening. *Portable Fortran subroutine in CMLIB (Q1DA sublibrary).* **Classes:** H2a1a1 **Usage:** CALL Q1DA(A, B, EPS, R, E, KF, IFLAG) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**Q1DA.** Automatic integration of a user-defined function of one variable. Special features include randomization and singularity weakening. *Portable Fortran software in NMS library.* **Classes:** H2a1a1 **Usage:** CALL Q1DA(A, B, EPS, RESULT, E, KF, IFLAG) **Precision:** Single (Double: DQ1DA) **Availability:** PC.

**Q1DAX.** Flexible subroutine for the automatic integration of a user-defined function of one variable. Special features include randomization, singularity weakening, restarting, specification of an initial mesh (optional), and output of smallest and largest integrand values. *Portable Fortran subroutine in CMLIB (Q1DA sublibrary).* **Classes:** H2a1a1 **Usage:** CALL Q1DAX(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**Q1DB.** Automatic integration of a user-defined function of one variable. Integrand must be a Fortran function but user may select name. Special features include randomization and singularity weakening. Intermediate in usage difficulty between Q1DA and Q1DAX. *Portable Fortran subroutine in CMLIB (Q1DA sublibrary).* **Classes:** H2a1a1 **Usage:** CALL Q1DB(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAG.** Automatic adaptive integrator, will handle many non-smooth integrands using Gauss Kronrod formulas. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a1 **Usage:** CALL QAG(*15 parameters*) **Precision:** Single (Double: DQAG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGE.** Automatic adaptive integrator, can handle most non-smooth functions, also provides more information than QAG. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a1 **Usage:** CALL QAGE(*17 parameters*) **Precision:** Single (Double: DQAGE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGI.** Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a3a1, H2a4a1 **Usage:** CALL QAGI(*14 parameters*) **Precision:** Single (Double: DQAGI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGI.** Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation. *Portable Fortran software in NMS library.* **Classes:** H2a3a1, H2a4a1 **Usage:** CALL QAGI(*14 parameters*) **Precision:** Single (Double: DQAGI) **Availability:** PC.

**QAGIE.** Automatic integrator for semi-infinite or infinite intervals and general integrands, provides more information than QAGI. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a3a1, H2a4a1 **Usage:** CALL QAGIE(*16 parameters*) **Precision:** Single (Double: DQAGIE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGP.** Automatic adaptive integrator, allows user to specify location of singularities or difficulties of integrand, uses extrapolation. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1 **Usage:** CALL QAGP(*15 parameters*) **Precision:** Single (Double: DQAGP) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGPE.** Automatic adaptive integrator for function with user specified endpoint singularities, provides more information that QAGP. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1 **Usage:** CALL QAGPE(*20 parameters*) **Precision:** Single (Double: DQAGPE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGS.** Automatic adaptive integrator, will handle most non-smooth integrands including those with endpoint singularities, uses extrapolation. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a1 **Usage:** CALL QAGS(*14 parameters*) **Precision:** Single (Double: DQAGS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAGSE.** Automatic adaptive integrator, can handle integrands with endpoint singularities, provides more information than QAGS. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a1 **Usage:** CALL QAGSE(*16 parameters*) **Precision:** Single (Double: DQAGSE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAND.** Integrate a function on a hyper-rectangle. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2b1a1 **Usage:** CALL QAND (*9 parameters*) **Precision:** Single (Double: DQAND) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QAWC.** Cauchy principal value integrator, using adaptive Clenshaw Curtis method (real Hilbert transform). *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1, J4 **Usage:** CALL QAWC(*15 parameters*)

**Precision:** Single (Double: DQAWC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWCE.** Cauchy principal value integrator, provides more information than QAWC (real Hilbert transform). *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1, J4 **Usage:** CALL QAWCE(*17 parameters*) **Precision:** Single (Double: DQAWCE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWF.** Automatic integrator for Fourier integrals on (a, $\infty$) with factors $\sin(\omega x)$, $\cos(\omega x)$ by integrating between zeros. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a3a1 **Usage:** CALL QAWF(*16 parameters*) **Precision:** Single (Double: DQAWF) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWFE.** Automatic integrator for Fourier integrals, with $\sin(\omega x)$ factor on (a, $\infty$), provides more information than QAWF. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a3a1 **Usage:** CALL QAWFE(*23 parameters*) **Precision:** Single (Double: DQAWFE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWO.** Automatic adaptive integrator for integrands with oscillatory sine or cosine factor. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1 **Usage:** CALL QAWO(*17 parameters*) **Precision:** Single (Double: DQAWO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWOE.** Automatic integrator for integrands with explicit oscillatory sine or cosine factor, provides more information than QAWO. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1 **Usage:** CALL QAWOE(*22 parameters*) **Precision:** Single (Double: DQAWOE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWS.** Automatic integrator for functions with explicit algebraic and/or logarithmic endpoint singularities. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1 **Usage:** CALL QAWS(*17 parameters*) **Precision:** Single (Double: DQAWS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QAWSE.** Automatic integrator for integrands with explicit algebraic and/or logarithmic endpoint singularities, provides more information than QAWS. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1 **Usage:** CALL QAWSE(*19 parameters*) **Precision:** Single (Double: DQAWSE) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QC25C.** Uses 25 point Clenshaw-Curtis formula to estimate integral of f·w where w=1/(x-c). *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a2, J4 **Usage:** CALL QC25C(*8 parameters*) **Precision:** Single (Double: DQC25C) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QC25F.** Clenshaw-Curtis integration rule for function with cos or sin factor, also uses Gauss Kronrod formula. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a2 **Usage:** CALL QC25F(*15 parameters*) **Precision:** Single (Double: DQC25F) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QC25S.** Estimates integral of function with algebraic-logarithmic singularities using 25 point Clenshaw-Curtis formula and gives error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a2 **Usage:** CALL QC25S(*14 parameters*) INTEGR, NEV) **Precision:** Single (Double: DQC25S) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QCOSB.** Compute a sequence from its cosine Fourier coefficients with only odd wave numbers. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a3 **Usage:** CALL QCOSB (N, COEF, SEQ) **Also see:** QCOSI **Precision:** Single (Double: DQCOSB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QCOSF.** Compute the coefficients of the cosine Fourier transform with only odd wave numbers. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a3 **Usage:** CALL QCOSF (N, SEQ, COEF) **Also see:** QCOSI **Precision:** Single (Double: DQCOSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QD2DR.** Evaluate the derivative of a function defined on a rectangular grid using quadratic interpolation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E2a **Usage:** R = QD2DR(*11 parameters*) **Precision:** Single (Double: DQD2DR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QD2VL.** Evaluate a function defined on a rectangular grid using quadratic interpolation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E2a **Usage:** R = QD2VL(*9 parameters*) **Precision:** Single (Double: DQD2VL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QD3DR.** Evaluate the derivative of a function defined on a rectangular three-dimensional grid using quadratic interpolation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E2a **Usage:** R = QD3DR(*16 parameters*) **Precision:** Single (Double: DQD3DR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QD3VL.** Evaluate a function defined on a rectangular three-dimensional grid using quadratic interpolation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E2a **Usage:** R = QD3VL(*13 parameters*) **Precision:** Single (Double: DQD3VL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAG.** Integrate a function using a globally adaptive scheme based on Gauss-Kronrod rules. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a1a1 **Usage:** CALL QDAG (*8 parameters*) **Precision:** Single (Double: DQDAG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAGI.** Integrate a function over an infinite or semi-infinite interval. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a3a1 **Usage:** CALL QDAGI (*7 parameters*) **Precision:** Single (Double: DQDAGI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAGP.** Integrate a function with singularity points given. *Proprietary Fortran subroutine in IMSL*

*MATH/LIBRARY.* **Classes:** H2a2a1 **Usage:** CALL QDAGP (*9 parameters*) **Precision:** Single (Double: DQDAGP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAGS.** Integrate a function (which may have endpoint singularities). *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** H2a1a1 **Usage:** CALL QDAGS (*7 parameters*) **Precision:** Single (Double: DQDAGS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAWC.** Integrate a function f(x)/(x-c) in the Cauchy principal value sense. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a2a1 **Usage:** CALL QDAWC (*8 parameters*) **Precision:** Single (Double: DQDAWC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAWF.** Compute a Fourier integral. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a3a1 **Usage:** CALL QDAWF (*7 parameters*) **Precision:** Single (Double: DQDAWF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAWO.** Integrate a function containing a sine or a cosine. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a2a1 **Usage:** CALL QDAWO (*9 parameters*) **Precision:** Single (Double: DQDAWO) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDAWS.** Integrate a function with algebraic-logarithmic singularities. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a2a1 **Usage:** CALL QDAWS (*10 parameters*) **Precision:** Single (Double: DQDAWS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDDER.** Evaluate the derivative of a function defined on a set of points using quadratic interpolation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** R = QDDER(*6 parameters*) **Precision:** Single (Double: DQDDER) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDNG.** Integrate a smooth function using a nonadaptive rule. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2a1a1 **Usage:** CALL QDNG (*7 parameters*) **Precision:** Single (Double: DQDNG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QDVAL.** Evaluate a function defined on a set of points using quadratic interpolation. *Proprietary Fortran function in IMSL MATH/LIBRARY.* **Classes:** E1a **Usage:** R = QDVAL(X, NDATA, XDATA, FDATA, CHECK) **Precision:** Single (Double: DQDVAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QK15.** Evaluates integral of given function on an interval with a 15 point Gauss Kronrod formula and returns error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a2 **Usage:** CALL QK15(*7 parameters*) **Precision:** Single (Double: DQK15) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK15.** Evaluates integral of given function on an interval with a 15 point Gauss Kronrod formula and returns error estimate. *Portable Fortran software in NMS library.* **Classes:** H2a1a2 **Usage:** CALL QK15(*7 parameters*) **Precision:** Single (Double: DQK15) **Availability:** PC.

**QK15I.** Evaluates integral of given function on semi-infinite or infinite interval with a transformed 15 point Gauss Kronrod formula and gives error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a3a2, H2a4a2 **Usage:** CALL QK15I(*9 parameters*) **Precision:** Single (Double: DQK15I) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK15W.** Evaluates integral of given function times arbitrary weight function on interval with 15 point Gauss Kronrod formula and gives error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a2 **Usage:** CALL QK15W(*13 parameters*) **Precision:** Single (Double: DQK15W) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK21.** Evaluates integral of given function on an interval with a 21 point Gauss Kronrod formula and returns error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a2 **Usage:** CALL QK21(*7 parameters*) **Precision:** Single (Double: DQK21) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK31.** Evaluates integral of given function on an interval with a 31 point Gauss Kronrod formula and returns error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a2 **Usage:** CALL QK31(*7 parameters*) **Precision:** Single (Double: DQK31) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK41.** Evaluates integral of given function on an interval with a 41 point Gauss Kronrod formula and returns error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a2 **Usage:** CALL QK41(*7 parameters*) **Precision:** Single (Double: DQK41) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK51.** Evaluates integral of given function on an interval with a 51 point Gauss Kronrod formula and returns error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a2 **Usage:** CALL QK51(*7 parameters*) **Precision:** Single (Double: DQK51) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QK61.** Evaluates integral of given function on an interval with a 61 point Gauss Kronrod formula and returns error estimate. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a2 **Usage:** CALL QK61(*7 parameters*) **Precision:** Single (Double: DQK61) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QMOMO.** Computes integral of k-th degree Chebyshev polynomial times one of a selection of functions with various singularities. *Portable Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a2a1, C3a2 **Usage:** CALL QMOMO(*7 parameters*) **Precision:** Single (Double: DQMOMO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QNG.** Automatic non-adaptive integrator for smooth functions, using Gauss Kronrod Patterson formulas. *Portable*

*Fortran subroutine in CMLIB (QUADPKS sublibrary).* **Classes:** H2a1a1 **Usage:** CALL QNG(*9 parameters*) **Precision:** Single (Double: DQNG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**QPROG.** Solve a quadratic programming problem subject to linear equality/ inequality constraints. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G2e1 **Usage:** CALL QPROG (*14 parameters*) **Precision:** Single (Double: DQPROG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QSINB.** Compute a sequence from its sine Fourier coefficients with only odd wave numbers. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a3 **Usage:** CALL QSINB (N, COEF, SEQ) **Also see:** QSINI **Precision:** Single (Double: DQSINB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QSINF.** Compute the coefficients of the sine Fourier transform with only odd wave numbers. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** J1a3 **Usage:** CALL QSINF (N, SEQ, COEF) **Also see:** QSINI **Precision:** Single (Double: DQSINF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QSORT.** To sort the elements of a real or integer array in a non-decreasing, or ascending, order with respect to algebraic value. *Fortran/meta subroutine in MAGEV.* **Classes:** N6a2a, N6a2b **Usage:** CALL QSORT(N, X, IX) **Precision:** Single **Availability:** 205 (vectorized)

**QTEST.** Perform a Cochran Q test for related observations. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4c1b **Usage:** CALL QTEST (NOBS, NVAR, X, LDX, Q, PQ) **Precision:** Single (Double: DQTEST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**QUAD.** Finds the integral of a general user defined EXTERNAL function by an adaptive technique to given absolute accuracy. *Proprietary Fortran subroutine in PORT.* **Classes:** H2a1a1 **Usage:** CALL QUAD (F, A, B, EPS, ANS, ERREST) **Precision:** Single (Double: DQUAD) **Availability:** 855NOS, 205.

**QUADIT.** To provide a printed report about the utilization of system resources. *Fortran/meta subroutine in MAGEV.* **Classes:** R, S3 **Usage:** CALL QUADIT(NUNIT, ID) **Precision:** Single **Availability:** 205 (vectorized)

**QUANTILE-QUANTILE PLOT.** Generate a quantile-quantile plot. Horizontal axis contains the estimated quantiles from data set one and vertical axis the estimated quantiles from data set two. *Command(s) in DATAPLOT interactive system.* **Classes:** L4b1a1 **Usage:** QUANTILE-QUANTILE PLOT <VARIABLE 1> <VARIABLE 2> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**QUICK.** Finds a quick partition of the cases by comparing, to a user-defined threshold, the Euclidean distances to the existing cluster leaders. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL QUICK(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**QUICK CLUSTER.** Produces clusters by finding cluster centers based on values of cluster variables and by assigning cases to centers that are nearest. Default is McQueen's k-means clustering method. Options: specify the number of clusters, select initial cluster centers, update centers, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L14a1b **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**QZHES.** The first step of the QZ algorithm for solving generalized matrix eigenproblems. Accepts a pair of real general matrices and reduces one of them to upper Hessenberg form and the other to upper triangular form using orthogonal transformations. Usually followed by QZIT, QZVAL, QZVEC. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b3 **Usage:** CALL QZHES(NM, N, A, B, MATZ, Z) **Also see:** QZIT QZVAL QZVEC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**QZIT.** The second step of the QZ algorithm for generalized eigenproblems. Accepts an upper Hessenberg and an upper triangular matrix and reduces the former to quasi-triangular form while preserving the form of the latter. Usually preceded by QZHES and followed by QZVAL and QZVEC. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b3 **Usage:** CALL QZIT(NM, N, A, B, EPS1, MATZ, Z, IERR) **Also see:** QZHES QZVAL QZVEC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**QZVAL.** The third step of the QZ algorithm for generalized eigenproblems. Accepts a pair of real matrices, one in quasi-triangular form and the other in upper triangular form and computes the eigenvalues of the associated eigenproblem. Usually preceded by QZHES, QZIT, and followed by QZVEC. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c2c **Usage:** CALL QZVAL(*9 parameters*) **Also see:** QZHES QZIT QZVEC **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**QZVEC.** The optional fourth step of the QZ algorithm for generalized eigenproblems. Accepts a matrix in quasi-triangular form and another in upper triangular form and computes the eigenvectors of the triangular problem and transforms them back to the original coordinates. Usually preceded by QZHES, QZIT, QZVAL. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c3 **Usage:** CALL QZVEC(NM, N, A, B, ALFR, ALFI, BETA, Z) **Also see:** QZHES QZIT QZVAL **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

# R

**R1MAC.** Returns single precision machine-dependent constants (like largest allowed floating-point number) for the local machine environment. *Proprietary Fortran function in the Scientific Desk.* **Classes:** R1 **Usage:** R = R1MAC(I) **Precision:** Single (Double: R1MACD) **Availability:** PC.

**R1MACD.** *Double precision version of R1MAC.*

**R1MACH.** Returns single precision machine-dependent constants (like largest allowed floating-point number) for the local machine environment. *Portable Fortran function in CMLIB (MACHCON sublibrary).* **Classes:** R1 **Usage:** R = R1MACH(I) **Precision:** Single (Double: D1MACH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**R1MACH.** Provides the single precision machine-dependent constants required to adapt PORT library programs to individual computers. *Proprietary Fortran function in PORT.* **Classes:** R1 **Usage:** R = R1MACH(I) **Precision:** Single (Double: D1MACH) **Availability:** 855NOS, 205.

**R1MACI.** Returns integer machine-dependent constants (like largest allowed integer) for the local machine environment. *Proprietary Fortran function in the Scientific Desk.* **Classes:** R1 **Usage:** I = R1MACI(K) **Precision:** Single **Availability:** PC.

**R3A.** Accesses or changes an indicator which causes specific types of error messages to be printed. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** R3a **Usage:** CALL R3A(IOPT, ITYPE) **Availability:** PC.

**R3B.** Accesses the current input and output unit reference numbers for error messages. Can also be used to reset the unit reference numbers. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** R3b **Usage:** CALL R3B(IOPT, IN, IOUT) **Availability:** PC.

**R3C.** Prints an error message on the current output device, that is, the device associated with the output reference number supplied by R3B. The type of error message to print is determined by a call to R3A. *Proprietary Fortran subroutine in the Scientific Desk.* **Classes:** R3c **Usage:** CALL R3C(KE, CODE) **Availability:** PC.

**RANBYT.** Returns the real random variate generated by UNI, together with its bit pattern presented in four 8-bit bytes. *Proprietary Fortran subroutine in PORT.* **Classes:** L6a21 **Usage:** CALL RANBYT (UNI, IBYTE) **Also see:** UNI RANSET **Precision:** Single **Availability:** 855NOS, 205.

**RAND.** Uniform random number on [0,1]. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** L6a21 **Usage:** R = RAND(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RANGE.** Computes the sample range of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1b **Usage:** CALL RANGE (X, N, IWRITE, XRANGE) **Precision:** Single **Availability:** 855NOS, 840NOS.

**RANK.** Produces rank profiles by ranking cases for each variable and reordering variables to minimize crossings. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L3e4 **Usage:** CALL RANK(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**RANK.** Ranks (in ascending order) the N elements of the single precision vector X, and puts the resulting N ranks into the vector XR. *Fortran subroutine in DATAPAC.* **Classes:** N6a1b **Usage:** CALL RANK(X, N, XR) **Precision:** Single **Availability:** 855NOS, 840NOS.

**RANK.** Ranks the values in a vector. Ties are assigned the average rank. *Command in MINITAB proprietary interactive system.* **Classes:** L2a **Usage:** RANK the values in C, put ranks into C **Precision:** Single **Availability:** 855NOS.

**RANK.** Computes ranks for one or more numeric variables across the observations of a data set. Options: group continuous data into ranges, fractional ranks, normal scores (Blom, Tukey, or van der Waerden), Savage (exponential) scores. *Proprietary stand-alone program using SAS command language.* **Classes:** L2a **Precision:** Single **Availability:** CAMVAX.

**RANKS.** Compute the ranks, normal scores, or exponential scores for a vector of observations. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L2a **Usage:** CALL RANKS (*6 parameters*) **Precision:** Single (Double: DRANKS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RANPER.** Generates a random permutation of size N of the values 1.0, 2.0, 3.0, ..., N-1, N. *Fortran subroutine in DATAPAC.* **Classes:** L6a16 **Usage:** CALL RANPER(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**RANSET.** Initializes the uniform random number generator, UNI, to other than the default initial values. *Proprietary Fortran subroutine in PORT.* **Classes:** L6c **Usage:** CALL RANSET (ICSEED, ITSEED) **Also see:** UNI RANBYT **Precision:** Single **Availability:** 855NOS, 205.

**RATCH.** Compute a rational weighted Chebyshev approximation to a continuous function on an interval. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K2 **Usage:** CALL RATCH (*10 parameters*) **Precision:** Single (Double: DRATCH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RATQR.** Computes largest or smallest eigenvalues of symmetric tridiagonal matrix using rational QR method with

Newton correction. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL RATQR(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RBEST.** Select the best multiple linear regression models. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1a2 **Usage:** CALL RBEST (*15 parameters*) **Precision:** Single (Double: DRBEST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RC.** Carlson's incomplete elliptic integral RC(x,y). *Portable Fortran function in CMLIB (FCNPAK sublibrary).* **Classes:** C14 **Usage:** R = RC(X, Y, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RCASE.** Compute case statistics and diagnostics given data points, coefficient estimates, and the R matrix for a fitted general linear model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1c, L8c1d **Usage:** CALL RCASE (*30 parameters*) **Precision:** Single (Double: DRCASE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RCASP.** Compute case statistics for a polynomial regression model given the fit based on orthogonal polynomials. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8b1c, L8b1d **Usage:** CALL RCASP (*23 parameters*) **Precision:** Single (Double: DRCASP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RCOMP.** Generate an orthogonal central composite design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8d **Usage:** CALL RCOMP (*8 parameters*) **Precision:** Single (Double: DRCOMP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RCOV.** Fit a multiple linear regression model given the variance-covariance matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1b2, L8f **Usage:** CALL RCOV (*15 parameters*) **Precision:** Single (Double: DRCOV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RCOVB.** Compute the estimated variance-covariance matrix of the estimated regression coefficients given the R matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1c **Usage:** CALL RCOVB (*6 parameters*) **Precision:** Single (Double: DRCOVB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RCURV.** Fit a polynomial curve using least squares. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8b1b2 **Usage:** CALL RCURV (*7 parameters*) **Precision:** Single (Double: DRCURV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RD.** Carlson's incomplete elliptic integral RD(x,y,z). *Portable Fortran function in CMLIB (FCNPAK sublibrary).* **Classes:** C14 **Usage:** R = RD(X, Y, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**READ.** Performs a format-free read. *Fortran subroutine in DATAPAC.* **Classes:** N1 **Usage:** CALL READ(ICOL1, ICOL2, X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**READG.** Performs a format-free read of data from input unit IRD. *Fortran subroutine in DATAPAC.* **Classes:** N1 **Usage:** CALL READG(IRD, ICOL1, ICOL2, X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**REBAK.** Forms eigenvectors of generalized symmetric eigensystem from eigenvectors of derived matrix output from REDUC or REDUC2. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL REBAK(NM, N, B, DL, M, Z) **Also see:** REDUC REDUC2 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**REBAKB.** Forms eigenvectors of generalized symmetric eigensystem from eigenvectors of derived matrix output from REDUC2. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL REBAKB(NM, N, B, DL, M, Z) **Also see:** REDUC2 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RECCF.** Compute recurrence coefficients for various monic polynomials. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2c **Usage:** CALL RECCF (*6 parameters*) **Precision:** Single (Double: DRECCF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RECQR.** Compute recurrence coefficients for monic polynomials given a quadrature rule. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2c **Usage:** CALL RECQR (N, QX, QW, NTERM, B, C) **Precision:** Single (Double: DRECQR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**REDUC.** Reduces generalized symmetric eigenproblem $Ax=\lambda Bx$, to standard symmetric eigenproblem, using Cholesky factorization. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1c **Usage:** CALL REDUC(NM, N, A, B, DL, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**REDUC2.** Reduces certain generalized symmetric eigenproblems to standard symmetric eigenproblem, using Cholesky factorization. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1c **Usage:** CALL REDUC2(NM, N, A, B, DL, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**REG.** Fits least-squares estimates to linear regression models. Options: weights; parameter estimates, predicted values, residuals, Studentized residuals, confidence limits, hypothesis tests; collinearity diagnostics; influence diagnostics including partial regression leverage plots; Durbin-Watson statistic; hypothesis tests involving multiple dependent variables; parameter estimates subject to linear restriction. *Proprietary stand-alone program using SAS* *command language.* **Classes:** L8a1a, L8c1b **Precision:** Single **Availability:** CAMVAX.

**REGRESS.** Performs simple or multiple linear regression, prints standard results. Options: amount of output, save results, weights, handle missing values, through the origin, compute and save regression diagnostics, lack of fit tests. *Command in MINITAB proprietary interactive system.* **Classes:** L8a1a, L8c1b1 **Usage:** REGRess C on K predictors C, . . ., C [put standardized residuals in C [fits in C]] [; subcommands NOCONSTANT; WEIGHTS in C; MSE into K; COEF into C; XPXINV into M; RMATRIX into M; HI into C; RESIDS into C; TRESIDS into C; COOKD into C; DFITS into C; VIF; PURE error lack of fit test; XLOF experimental lack of fit.] **Precision:** Single **Availability:** 855NOS.

**REGRESSION.** Calculates multiple regression equation and associated statistics and plots. Options: backward, forward, stepwise, or subset of variables selection, residual diagnostics and analysis, partial residual plots, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L8a1a1, L8c1a1, L8c1a2 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**RELOC.** Sets each cluster center equal to the cluster mean. Can be used with CLUSTER subroutine ASSIGN to form a user-defined K-means package. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL RELOC(*9 parameters*) **Also see:** ASSIGN **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**RELSD.** Computes the sample relative standard deviation of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1b **Usage:** CALL RELSD(X, N, IWRITE, XRELSD) **Precision:** Single **Availability:** 855NOS, 840NOS.

**REPLAC.** Replaces (with the value XNEW) all observations in the vector X which are inside the interval [XMIN, XMAX]. *Fortran subroutine in DATAPAC.* **Classes:** L2a **Usage:** CALL REPLAC(X, N, XMIN, XMAX, XNEW) **Precision:** Single **Availability:** 855NOS, 840NOS.

**REPORT.** Produces case listings, univariate summary statistics, absolute and relative frequencies for subpopulations with user control of output appearance. Options: page lengths, column widths, margins, titles, footnotes, labels, concatenation of variables and literals for display, functions of summary statistics, missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L1c1 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**RETAIN.** Retains all observations in the vector X which are inside the interval [XMIN, XMAX]. *Fortran subroutine in DATAPAC.* **Classes:** L2c **Usage:** CALL RETAIN(X, N, XMIN, XMAX, NEWN) **Precision:** Single **Availability:** 855NOS, 840NOS.

**RETSRC.** Test and reset error recovery mode for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3a **Usage:** CALL RETSRC (IROLD) **Precision:** Single **Availability:** 855NOS, 205.

**RF.** Carlson's incomplete elliptic integral RF(x,y,z). *Portable Fortran function in CMLIB (FCNPAK sublibrary).* **Classes:** C14 **Usage:** R = RF(X, Y, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RFFTB.** Computes real periodic sequence from real Fourier coefficients. Performs Fourier synthesis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a1 **Usage:** CALL RFFTB(N, R, WSAVE) **Also see:** RFFTF RFFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RFFTF.** Computes Fourier coefficients of real periodic sequence (fast). Performs Fourier analysis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a1 **Usage:** CALL RFFTF(N, R, WSAVE) **Also see:** RFFTB RFFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RFFTI.** Initialize WSAVE array for SUBROUTINE RFFTF and RFFTB. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a1 **Usage:** CALL RFTI(N, WSAVE) **Also see:** RFFTF RFFTB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RFORP.** Fit an orthogonal polynomial regression model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8b1b2 **Usage:** CALL RFORP (*24 parameters*) **Precision:** Single (Double: DRFORP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RG.** Computes eigenvalues and, optionally, eigenvectors of a real general matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a2 **Usage:** CALL RG(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RGAUSS.** Normal random number. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** L6a14 **Usage:** R = RGAUSS(XMEAN, SD) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RGG.** Computes eigenvalues and eigenvectors for real generalized eigenproblem: $Ax=\lambda Bx$. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4b2 **Usage:** CALL RGG(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RGIVN.** Fit a multivariate linear regression model via fast Givens transformations. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1b1, L8f **Usage:** CALL RGIVN (*26 parameters*) **Precision:** Single (Double: DRGIVN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RGLM.** Fit a multivariate general linear model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8f **Usage:** CALL RGLM (*34 parameters*) **Precision:** Single (Double: DRGLM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RGM.** Computes estimates of simple linear regression parameters for a geometric mean regression. *Portable Fortran subroutine in CMLIB (SLRPACK sublibrary).* **Classes:** L8a4 **Usage:** CALL RGM(N, X, Y, OUTPUT, IER) **Also see:** RYORK RWILL **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RINCF.** Perform response control given a fitted simple linear regression model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8a1d **Usage:** CALL RINCF (*13 parameters*) **Precision:** Single (Double: DRINCF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RINPF.** Perform inverse prediction given a fitted simple linear regression model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8a1d **Usage:** CALL RINPF (*14 parameters*) **Precision:** Single (Double: DRINPF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RJ.** Carlson's incomplete elliptic integral RJ(x,y,z,p). *Portable Fortran function in CMLIB (FCNPAK sublibrary).* **Classes:** C14 **Usage:** R = RJ(X, Y, Z, P, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RKF45.** Runge-Kutta Felberg method for the integration of a first order system of ordinary differential equations. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** I1a1a **Precision:** Single **Availability:** 855NOS.

**RLAV.** Fit a multiple linear regression model using the least absolute values criterion. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8a2, L8c3, K3 **Usage:** CALL RLAV (*13 parameters*) **Precision:** Single (Double: DRLAV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLCOMP.** Generation of an orthogonal central composite design. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8d **Usage:** CALL RLCOMP (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLDCQM.** Decoding of a quadratic regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLDCQM (XBAR, M, BN) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLDCVA.** Variance estimates for decoded orthogonal polynomial regression coefficients. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLDCVA (V, ID, A, B, SC, T, IT, IER) **Also see:** RLDCW **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLDCW.** Variances of coded orthogonal polynomial regression coefficients for usage in conjunction with IMSL routines RLFOTH and RLFOTW, and provided to prepare input for IMSL routine RLDCVA. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLDCW (*12 parameters*) **Also see:** RLFOTH RLFOTW RLDCVA **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLDOPM.** Coefficient decoder for an orthogonal polynomial regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLDOPM (C, ID, A, B, T) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLEAP.** Leaps and bounds algorithm for determining a number of best regression subsets from a full regression model. USLEAP is a special purpose output routine designed to be used only in conjunction with RLEAP. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1a2 **Usage:** CALL RLEAP (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLEQU.** Fit a multivariate linear regression model with linear equality restrictions $H\beta = G$ imposed on the regression parameters given results from IMSL routine RGIVN after IDO = 1 and IDO = 2 and prior to IDO = 3. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** K1a2a, L8f **Usage:** CALL RLEQU (*20 parameters*) **Precision:** Single (Double: DRLEQU) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLFITI.** Pure replication error degrees of freedom and sum of squares – in-core version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1c **Usage:** CALL RLFITI (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLFITO.** Pure replication error degrees of freedom and sum of squares – out-of-core version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1c **Usage:** CALL RLFITO (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLFOR.** Fit a univariate curvilinear regression model using orthogonal polynomials with optional weighting and prediction analysis – easy-to-use version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLFOR (*13 parameters*) **Also see:** BDTRGI **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLFOTH.** Fit a univariate curvilinear regression model using orthogonal polynomials. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLFOTH (*12 parameters*) **Also see:** RLDOPM **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLFOTW.** Fit a univariate curvilinear regression model using orthogonal polynomials with weighting. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1b2 **Usage:** CALL RLFOTW (*13 parameters*) **Also see:** RLDOPM **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLGQMI.** Centering of independent variable settings and generation of centered square and cross product terms – in-core version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2a **Usage:** CALL RLGQMI (X, N, M, IX, XBAR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLGQMO.** Centering of independent variable settings and generation of uncentered square and cross product terms

– out-of-core version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L2a **Usage:** CALL RLGQMO (X, N, M, I, XBAR, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLINCF.** Response control using a fitted simple linear regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8a1d **Usage:** CALL RLINCF (CRIT, IOP, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLINE.** Fit a line to a set of data points using least squares. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** K1a1a2, L8a1a1 **Usage:** CALL RLINE (*6 parameters*) **Precision:** Single (Double: DRLINE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLINE.** Fit a line to a set of data points using least squares. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8a1a1 **Usage:** CALL RLINE (*6 parameters*) **Precision:** Single (Double: DRLINE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLINE.** Fits straight line to x-y data by resistant line procedure – partitions data by x-value into three groups and uses an iterative procedure to find the line that makes the median residual in the left and the right partitions equal. *Command in MINITAB proprietary interactive system.* **Classes:** L8h **Usage:** RLINe y in C, x in C [put residuals into C [predicted values in C [coefficients into C]]] [; subcommand MAXITERATION = K] **Precision:** Single **Availability:** 855NOS.

**RLINPF.** Inverse prediction using a fitted simple linear regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8a1d **Usage:** CALL RLINPF (CRIT, IOP, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLLAV.** Perform linear regression using the least absolute values criterion. *Proprietary Fortran subroutine in IMSL.* **Classes:** K3, L8a2, L8c3 **Usage:** CALL RLLAV (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLLMV.** Perform linear regression using the minimax criterion. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8a2, K2, L8c3 **Usage:** CALL RLLMV (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLMUL.** Multiple linear regression analysis. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1b2 **Usage:** CALL RLMUL (*10 parameters*) **Also see:** BECOVM RLSUM **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLMV.** Fit a multiple linear regression model using the minimax criterion. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** K2, L8a2, L8c3 **Usage:** CALL RLMV (*13 parameters*) **Precision:** Single (Double: DRLMV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLONE.** Analysis of a simple linear regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8a1a1 **Usage:** CALL RLONE (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLOPDC.** Response prediction using an orthogonal polynomial regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8b1d **Usage:** CALL RLOPDC (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLPOL.** Generate orthogonal polynomials with associated constants AA and BB. *Proprietary Fortran subroutine in IMSL.* **Classes:** C3a, L2a, L8b1b2 **Usage:** CALL RLPOL (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLPRDI.** Confidence intervals for the true response and for the average of a set of future observations on the response – in-core version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1d **Usage:** CALL RLPRDI (Y, V, N, TS, NR, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLPRDO.** Confidence intervals for the true response and for the average of a set of future observations on the response – out-of-core version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1d **Usage:** CALL RLPRDO (Y, V, TS, NR, C) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLRES.** Perform a residual analysis for a fitted regression model. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1c **Usage:** CALL RLRES (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLSEP.** Selection of a regression model using a forward stepwise algorithm, and computation of the usual analysis of variance table entries – easy-to-use version. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1a1 **Usage:** CALL RLSEP (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLSTP.** Regression model selection using a forward stepwise algorithm with results available after each step. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8c1a2 **Usage:** CALL RLSTP (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLSUBM.** Retrieval of a symmetric submatrix from a matrix stored in symmetric storage mode by RLSTP. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8i **Usage:** CALL RLSUBM (A, M, IH, S, N) **Also see:** RLSTP **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLSUM.** Reordering of the rows and corresponding columns of a symmetric matrix stored in symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8i **Usage:** CALL RLSUM (AA, MM, IH, M, A, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RLTR.** An auxiliary routine for use together with FFT to transform 2N real data points. Uses less storage than

FFTR. *Proprietary Fortran subroutine in PORT.* **Classes:** J1a1 **Usage:** CALL RLTR (A, B, N, ISN) **Also see:** FFT **Precision:** Single (Double: DRLTR) **Availability:** 855NOS, 205.

**RNARM.** Generate a time series from a specified ARMA model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a20 **Usage:** CALL RNARM (*13 parameters*) **Precision:** Single (Double: DRNARM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNBET.** Generate pseudorandom numbers from a beta distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a2 **Usage:** CALL RNBET (NR, PIN, QIN, R) **Precision:** Single (Double: DRNBET) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNBIN.** Generate pseudorandom numbers from a binomial distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a2 **Usage:** CALL RNBIN (NR, N, P, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNCHI.** Generate pseudorandom numbers from a chi-squared distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a3 **Usage:** CALL RNCHI (NR, DF, R) **Precision:** Single (Double: DRNCHI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNCHY.** Generate pseudorandom numbers from a Cauchy distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a3 **Usage:** CALL RNCHY (NR, R) **Precision:** Single (Double: DRNCHY) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNCOR.** Generate a pseudorandom orthogonal matrix or a correlation matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6b3, L6b15 **Usage:** CALL RNCOR (*7 parameters*) **Precision:** Single (Double: DRNCOR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNEXP.** Generate pseudorandom numbers from a standard exponential distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a5 **Usage:** CALL RNEXP (NR, R) **Precision:** Single (Double: DRNEXP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNEXT.** Generate pseudorandom numbers from a mixture of two exponential distributions. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a5 **Usage:** CALL RNEXT (NR, THETA1, THETA2, P, R) **Precision:** Single (Double: DRNEXT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGAM.** Generate pseudorandom numbers from a standard gamma distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGAM (NR, A, R) **Precision:** Single (Double: DRNGAM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGCS.** Set up table to generate pseudorandom numbers from a general continuous distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGCS (*5 parameters*) **Precision:** Single (Double: DRNGCS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGCT.** Generate pseudorandom numbers from a general continuous distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGCT (NR, NINT, TABLE, LDTABL, R) **Precision:** Single (Double: DRNGCT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGDA.** Generate pseudorandom numbers from a general discrete distribution using an alias method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGDA (*8 parameters*) **Precision:** Single (Double: DRNGDA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGDS.** Set up table to generate pseudorandom numbers from a general discrete distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGDS (*8 parameters*) **Precision:** Single (Double: DRNGDS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGDT.** Generate pseudorandom numbers from a general discrete distribution using a table lookup method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGDT (NR, IMIN, NMASS, CUMPR, IR) **Precision:** Single (Double: DRNGDT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGEO.** Generate pseudorandom numbers from a geometric distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a7 **Usage:** CALL RNGEO (NR, P, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGES.** Retrieve the current value of the table in the IMSL random number generators that use shuffling. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6c **Usage:** CALL RNGES (TABLE) **Precision:** Single (Double: DRNGES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNGET.** Retrieve the current value of the seed used in the IMSL random number generators. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** L6c **Usage:** CALL RNGET (ISEED) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNHYP.** Generate pseudorandom numbers from a hypergeometric distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a8 **Usage:** CALL RNHYP (NR, N, M, L, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNKSM.** Perform the Wilcoxon rank sum test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b1b **Usage:** CALL RNKSM (*8 parameters*) **Precision:** Single (Double: DRNKSM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNLIN.** Fit a nonlinear regression model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:**

K1b1a1, K1b1a2, L8e1 **Usage:** CALL RNLIN (*9 parameters*) **Precision:** Single (Double: DRNLIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNLNL.** Generate pseudorandom numbers from a lognormal distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a12 **Usage:** CALL RNLNL (NR, XM, S, R) **Precision:** Single (Double: DRNLNL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNMTN.** Generate pseudorandom numbers from a multinomial distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6b13 **Usage:** CALL RNMTN (NR, N, K, P, IR, LDIR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNMVN.** Generate pseudorandom numbers from a multivariate normal distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6b14 **Usage:** CALL RNMVN (NR, K, RSIG, LDRSIG, R, LDR) **Precision:** Single (Double: DRNMVN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNNBN.** Generate pseudorandom numbers from a negative binomial distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a14 **Usage:** CALL RNNBN (NR, RK, P, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNNOA.** Generate pseudorandom numbers from a standard normal distribution using an acceptance/rejection method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a14 **Usage:** CALL RNNOA (NR, R) **Precision:** Single (Double: DRNNOA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNNOF.** Generate a pseudorandom number from a standard normal distribution. *Proprietary Fortran function in IMSL STAT/LIBRARY.* **Classes:** L6a14 **Usage:** R = RNNOF () **Precision:** Single (Double: DRNNOF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNNOR.** Generate pseudorandom numbers from a standard normal distribution using an inverse CDF method. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a14 **Usage:** CALL RNNOR (NR, R) **Precision:** Single (Double: DRNNOR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNNOS.** Generate pseudorandom order statistics from a standard normal distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a14 **Usage:** CALL RNNOS (IFIRST, ILAST, N, R) **Precision:** Single (Double: DRNNOS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNNPP.** Generate pseudorandom numbers from a nonhomogeneous Poisson process. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a16 **Usage:** CALL RNNPP (*8 parameters*) **Precision:** Single (Double: DRNNPP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNOPG.** Obtain the indicator of the uniform (0,1) multiplicative congruential pseudorandom number generator. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6c **Usage:** CALL RNOPG (IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNOPT.** Select the uniform (0,1) multiplicative congruential pseudorandom number generator. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** L6c **Usage:** CALL RNOPT (IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNOR.** Generates quasi normal random numbers with zero mean and unit standard deviation. *Portable Fortran function in CMLIB (RV sublibrary).* **Classes:** L6a14 **Usage:** R = RNOR(JD) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RNOR.** Standard normal generator with zero mean and unit standard deviation. Uses ziggaraut algorithm. Fast, excellent statistical properties and portable. *Portable Fortran software in NMS library.* **Classes:** L6a14 **Usage:** R = RNOR() **Precision:** Single **Availability:** PC.

**RNPER.** Generate a pseudorandom permutation. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a16 **Usage:** CALL RNPER (K, IPER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNPOI.** Generate pseudorandom numbers from a Poisson distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a16 **Usage:** CALL RNPOI (NR, THETA, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNSES.** Initialize the table in the IMSL random number generators that use shuffling. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6c **Usage:** CALL RNSES (TABLE) **Precision:** Single (Double: DRNSES) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNSET.** Initialize a random seed for use in the IMSL random number generators. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** L6c **Usage:** CALL RNSET (ISEED) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNSPH.** Generate pseudorandom points on a unit circle or K-dimensional sphere. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6b21 **Usage:** CALL RNSPH (NR, K, Z, LDZ) **Precision:** Single (Double: DRNSPH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNSRI.** Generate a simple pseudorandom sample of indices. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a19 **Usage:** CALL RNSRI (NSAMP, NPOP, INDEX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNSRS.** Generate a simple pseudorandom sample from a finite population. *Proprietary Fortran subroutine in IMSL*

*STAT/LIBRARY.* **Classes:** L6a19 **Usage:** CALL RNSRS (*10 parameters*) **Precision:** Single (Double: DRNSRS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNSTA.** Generate pseudorandom numbers from a stable distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a19 **Usage:** CALL RNSTA (NR, ALPHA, BPRIME, R) **Precision:** Single (Double: DRNSTA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNTAB.** Generate a pseudorandom two-way table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6b3 **Usage:** CALL RNTAB (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNTRI.** Generate pseudorandom numbers from a triangular distribution on the interval (0,1). *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a20 **Usage:** CALL RNTRI (NR, R) **Precision:** Single (Double: DRNTRI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNUN.** Generate pseudorandom numbers from a uniform (0,1) distribution. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** L6a21 **Usage:** CALL RNUN (NR, R) **Precision:** Single (Double: DRNUN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNUND.** Generate pseudorandom numbers from a discrete uniform distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a21 **Usage:** CALL RNUND (NR, K, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNUNF.** Generate a pseudorandom number from a uniform (0,1) distribution. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** L6a21 **Usage:** R = RNUNF() **Precision:** Single (Double: DRNUNF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNUNO.** Generate pseudorandom order statistics from a uniform (0,1) distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a21 **Usage:** CALL RNUNO (IFIRST, ILAST, N, R) **Precision:** Single (Double: DRNUNO) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNVMS.** Generate pseudorandom numbers from a von Mises distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a22 **Usage:** CALL RNVMS (NR, C, R) **Precision:** Single (Double: DRNVMS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RNWIB.** Generate pseudorandom numbers from a Weibull distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L6a23 **Usage:** CALL RNWIB (NR, A, R) **Precision:** Single (Double: DRNWIB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RONE.** Analyze a simple linear regression model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8a1a **Usage:** CALL RONE (*22 parameters*) **Precision:** Single (Double: DRONE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ROOTOGRAM.** Generate a rootogram plot, i.e. a histogram based on the square roots of the counts of data values in each bin. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a1 **Usage:** ROOTOGRAM <VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**ROOTOGRAM.** Prints a suspended rootogram, i.e. a histogram which has been fit with a Gaussian distribution based on square roots of the counts of data values in each bin and which uses medians and hinges. Options: specify Gaussian mean and standard deviation, save results. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1a1 **Usage:** ROOTogram data in C [use bin boundaries in C] [; subcommands BOUNDARIES into C; DRRS into C; FITTED values into C; COUNTS into C; FREQUENCIES are in C [bin boundaries are in C]; MEAN = K; STDEV = K.] **Precision:** Single **Availability:** 855NOS.

**RORDM.** Reorder rows and columns of a symmetric matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8i **Usage:** CALL RORDM (*7 parameters*) **Precision:** Single (Double: DRORDM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ROREX.** Reorder the responses from a balanced complete experimental design. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7d1 **Usage:** CALL ROREX (*6 parameters*) **Precision:** Single (Double: DROREX) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ROTIN.** Compute diagnostics for detection of outliers and influential data points given residuals and the R matrix for a fitted general linear model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1c **Usage:** CALL ROTIN (*16 parameters*) **Precision:** Single (Double: DROTIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RPOLY.** Analyze a polynomial regression model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** K1a1a2, L8b1a **Usage:** CALL RPOLY (*27 parameters*) **Precision:** Single (Double: DRPOLY) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RPOLY.** Finds zeros of a polynomial with real coefficients. Output zeros are in a pair of arrays, for real and imaginary part. *Proprietary Fortran subroutine in PORT.* **Classes:** F1a1 **Usage:** CALL RPOLY (DEGREE, COEFF, ZEROR, ZEROI) **Precision:** Single (Double: DRPOLY) **Availability:** 855NOS, 205.

**RPQR79.** Computes all the zeros of a general real polynomial using eigenvalue methods, requiring n-by-n storage for nth degree polynomial. *Portable Fortran subroutine in CMLIB (CPQR79 sublibrary).* **Classes:** F1a1 **Usage:** CALL RPQR79(NDEG, COEFF, ROOT, IERR, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205,

CAMVAX, 840NOS.

**RPZERO.** Computes all the zeros of a polynomial with real coefficients. Error bounds are also computed. Uses Newton's Method for systems. *Portable Fortran subroutine in CMLIB (CPZERO sublibrary).* **Classes:** F1a1 **Usage:** CALL RPZERO(N, A, R, T, IFLAG, S) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RQUAD.** Finds the integral of a general user defined EXTERNAL function by an adaptive technique. Combined absolute and relative error control. *Proprietary Fortran subroutine in PORT.* **Classes:** H2a1a1 **Usage:** CALL RQUAD (*7 parameters*) **Precision:** Single (Double: DRQUAD) **Availability:** 855NOS, 205.

**RS.** Computes eigenvalues and, optionally, eigenvectors of a real symmetric matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a1 **Usage:** CALL RS(NM, N, A, W, MATZ, Z, FV1, FV2, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSB.** Computes eigenvalues and, optionally, eigenvectors of real symmetric band matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a6 **Usage:** CALL RSB(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSG.** Computes eigenvalues and, optionally, eigenvectors of real symmetric generalized eigenproblem: Ax=λBx. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4b1 **Usage:** CALL RSG(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSGAB.** Computes eigenvalues and, optionally, eigenvectors of real symmetric generalized eigenproblem: ABx=λx. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4b1 **Usage:** CALL RSGAB(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSGBA.** Computes eigenvalues and, optionally, eigenvectors of real symmetric generalized eigenproblem: BAx=λx. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4b1 **Usage:** CALL RSGBA(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSMITZ.** Least squares fit of the non-linear regression model $y_i = \alpha + \beta\gamma^{x_i} + e_i$. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8e1b1 **Usage:** CALL RSMITZ (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**RSMOOTH.** Computes resistant smoother by 4253H, twice (or 3RSSH, twice), i.e. successive application of running medians and Hanning (running weighted averages), and save results. *Command in MINITAB proprietary interactive system.* **Classes:** L8h **Usage:** RSMOoth C, put rough into C, smooth into C [; subcommand SMOOTH 3RSSH, twice.] **Precision:** Single **Availability:** 855NOS.

**RSP.** Compute eigenvalues and, optionally, eigenvectors of a real symmetric matrix packed into a one dimensional array. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a1 **Usage:** CALL RSP(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSQUARE.** Uses the R-squared statistic to select optimal subsets of independent variables for multiple regression. Can specify largest and smallest number of independent variables for a subset and number of subsets of each size. Options: weights; statistics for each model selected including Akaike's information criterion, Mallows' C-p, and others. *Proprietary stand-alone program using SAS command language.* **Classes:** L8c1a1, L8c1a2 **Precision:** Single **Availability:** CAMVAX.

**RSREG.** Estimates a quadratic response surface using least-squares regression and determines critical values to optimize the response. Options: weights, lack of fit test, surface plotting, eigenvalues of the associated quadratic form. *Proprietary stand-alone program using SAS command language.* **Classes:** L8d **Precision:** Single **Availability:** CAMVAX.

**RST.** Compute eigenvalues and, optionally, eigenvectors of a real symmetric tridiagonal matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5 **Usage:** CALL RST(NM, N, W, E, MATZ, Z, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RSTAP.** Compute summary statistics for a polynomial regression model given the fit based on orthogonal polynomials. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8b1c, L8b1d **Usage:** CALL RSTAP (*20 parameters*) **Precision:** Single (Double: DRSTAP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RSTAT.** Compute statistics related to a regression fit given the coefficient estimates and the R matrix. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1d **Usage:** CALL RSTAT (*16 parameters*) **Precision:** Single (Double: DRSTAT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RSTEP.** Build multiple linear regression models using forward selection, backward selection, or stepwise selection. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8c1a2 **Usage:** CALL RSTEP (*21 parameters*) **Precision:** Single (Double: DRSTEP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RT.** Compute eigenvalues and eigenvectors of a special real tridiagonal matrix. *Portable Fortran subroutine in CMLIB*

*(EISPACK sublibrary).* **Classes:** D4a5 **Usage:** CALL RT(NM, N, A, W, MATZ, Z, FV1, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**RUN SEQUENCE PLOT.** Generates a run sequence plot (variable vs. dummy index). *Command(s) in DATAPLOT interactive system.* **Classes:** L3a6 **Usage:** RUN SEQUENCE PLOT <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**RUNIF.** Sequence of uniform random numbers on [0,1]. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** L6a21 **Usage:** R = RUNIF(T, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RUNS.** Performs a run analysis of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L4a1d **Usage:** CALL RUNS(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**RUNS.** Perform a runs analysis. Generated output includes the number of runs up, the number of runs down, the number of runs up and down, and the total number of runs. A z score is generated for the number of runs equal to I for I equal 1 through 10. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1d **Usage:** RUNS <VARIABLE NAME> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**RUNS.** Perform a runs up test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1d **Usage:** CALL RUNS (*11 parameters*) **Precision:** Single (Double: DRUNS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**RUNS.** Performs a two-sided runs test. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1d **Usage:** RUNS above and below K for data in column C **Precision:** Single **Availability:** 855NOS.

**RWILL.** Estimates simple linear regression coefficients when both variables are subject to errors which are not necessarily homogeneous in variance. (Method due to Williamson.) *Portable Fortran subroutine in CMLIB (SLRPACK sublibrary).* **Classes:** L8a4 **Usage:** CALL RWILL(*13 parameters*) **Also see:** RGM RYORK **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**RYORK.** Estimates simple linear regression coefficients when both variables are subject to errors which are not necessarily homogeneous in variance. *Portable Fortran subroutine in CMLIB (SLRPACK sublibrary).* **Classes:** L8a4 **Usage:** CALL RYORK(*14 parameters*) **Also see:** RGM RWILL **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

# S

**S07AAF.** tan(x). *Proprietary Fortran function in NAG.* **Classes:** C4a **Usage:** R = S07AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S09AAF.** arcsin(x). *Proprietary Fortran function in NAG.* **Classes:** C4a **Usage:** R = S09AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S09ABF.** arccos(x). *Proprietary Fortran function in NAG.* **Classes:** C4a **Usage:** R = S09ABF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S10AAF.** tanh(x). *Proprietary Fortran function in NAG.* **Classes:** C4c **Usage:** R = S10AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S10ABF.** sinh(x). *Proprietary Fortran function in NAG.* **Classes:** C4c **Usage:** R = S10ABF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S10ACF.** cosh(x). *Proprietary Fortran function in NAG.* **Classes:** C4c **Usage:** R = S10ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S11AAF.** arctanh(x). *Proprietary Fortran function in NAG.* **Classes:** C4c **Usage:** R = S11AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S11ABF.** arcsinh(x). *Proprietary Fortran function in NAG.* **Classes:** C4c **Usage:** R = S11ABF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S11ACF.** arccosh(x). *Proprietary Fortran function in NAG.* **Classes:** C4c **Usage:** R = S11ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S13AAF.** Exponential integral, ei(x). *Proprietary Fortran function in NAG.* **Classes:** C5 **Usage:** R = S13AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S13ACF.** Cosine integral, ci(x). *Proprietary Fortran function in NAG.* **Classes:** C6 **Usage:** R = S13ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S13ADF.** Sine integral, si(x). *Proprietary Fortran function in NAG.* **Classes:** C6 **Usage:** R = S13ADF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S14AAF.** $\Gamma(x)$. *Proprietary Fortran function in NAG.* **Classes:** C7a **Usage:** R = S14AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S14ABF.** $\ln(\Gamma(x))$. *Proprietary Fortran function in NAG.* **Classes:** C7a **Usage:** R = S14ABF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S15ABF.** Cumulative normal distribution function. *Proprietary Fortran function in NAG.* **Classes:** C8a, L5a1n **Usage:** R = S15ABF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S15ACF.** Complement of cumulative normal distribution function. *Proprietary Fortran function in NAG.* **Classes:** C8a, L5a1n **Usage:** R = S15ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S15ADF.** Complement of error function, erfc(x). *Proprietary Fortran function in NAG.* **Classes:** C8a, L5a1e **Usage:** R = S15ADF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S15AEF.** Error function, erf(x). *Proprietary Fortran function in NAG.* **Classes:** C8a, L5a1e **Usage:** R = S15AEF(X, IFAIL) **Also see:** S15ADF **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S15AFF.** Dawson's integral. *Proprietary Fortran function in NAG.* **Classes:** C8c **Usage:** R = S15AFF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17ACF.** Bessel functions $Y_0(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10a1 **Usage:** R = S17ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17ADF.** Bessel functions $Y_1(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10a1 **Usage:** R = S17ADF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17AEF.** Bessel functions $J_0(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10a1 **Usage:** R = S17AEF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17AFF.** Bessel functions $J_1(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10a1 **Usage:** R = S17AFF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17AGF.** Airy function Ai(x). *Proprietary Fortran function in NAG.* **Classes:** C10d **Usage:** R = S17AGF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17AHF.** Airy function Bi(x). *Proprietary Fortran function in NAG.* **Classes:** C10d **Usage:** R = S17AHF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17AJF.** Derivative of the Airy function Ai(x). *Proprietary Fortran function in NAG.* **Classes:** C10d **Usage:** R = S17AJF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S17AKF.** Derivative of the Airy function Bi(x). *Proprietary Fortran function in NAG.* **Classes:** C10d **Usage:** R = S17AKF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18ACF.** Modified Bessel function $K_0(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R =

S18ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18ADF.** Modified Bessel function $K_1(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18ADF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18AEF.** Modified Bessel function $I_0(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18AEF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18AFF.** Modified Bessel function $I_1(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18AFF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18CCF.** Exponentially scaled modified Bessel function $\exp(x)K_0(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18CCF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18CDF.** Exponentially scaled modified Bessel function $\exp(x)K_1(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18CDF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18CEF.** Exponentially scaled modified Bessel function $\exp(-|x|)I_0(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18CEF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S18CFF.** Exponentially scaled modified Bessel function $\exp(-|x|)I_1(x)$. *Proprietary Fortran function in NAG.* **Classes:** C10b1 **Usage:** R = S18CFF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S19AAF.** Kelvin function ber(x). *Proprietary Fortran function in NAG.* **Classes:** C10c **Usage:** R = S19AAF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S19ABF.** Kelvin function bei(x). *Proprietary Fortran function in NAG.* **Classes:** C10c **Usage:** R = S19ABF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S19ACF.** Kelvin function ker(x). *Proprietary Fortran function in NAG.* **Classes:** C10c **Usage:** R = S19ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S19ADF.** Kelvin function kei(x). *Proprietary Fortran function in NAG.* **Classes:** C10c **Usage:** R = S19ADF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S20ACF.** Fresnel integrals, s(x). *Proprietary Fortran function in NAG.* **Classes:** C8b **Usage:** R = S20ACF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S20ADF.** Fresnel integrals, c(x). *Proprietary Fortran function in NAG.* **Classes:** C8b **Usage:** R = S20ADF(X, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S21BAF.** Carlson's incomplete elliptic integral RC(x,y). *Proprietary Fortran function in NAG.* **Classes:** C14 **Usage:** R = S21BAF(X, Y, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S21BBF.** Carlson's incomplete elliptic integral RF(x,y,z). *Proprietary Fortran function in NAG.* **Classes:** C14 **Usage:** R = S21BBF(X, Y, Z, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S21BCF.** Carlson's incomplete elliptic integral RD(x,y,z). *Proprietary Fortran function in NAG.* **Classes:** C14 **Usage:** R = S21BCF(X, Y, Z, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**S21BDF.** Carlson's incomplete elliptic integral RJ(x,y,z,p). *Proprietary Fortran function in NAG.* **Classes:** C14 **Usage:** R = S21BDF(X, Y, Z, R, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**SADD.** Add a scalar to each component of a vector, $x = x + a$, all single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL SADD (N, SA, SX, INCX) **Precision:** Single (Double: DADD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SAMPLE.** Randomly selects without replacement values from one or more vectors, optionally carrying along other vectors. *Command in MINITAB proprietary interactive system.* **Classes:** L6a19 **Usage:** SAMPle K rows from C [..., C], put into C [..., C] **Precision:** Single **Availability:** 855NOS.

**SAMPLE.** Sample (extract) every NSth observation from a series; return sampled series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L2c, L10a1a **Usage:** CALL SAMPLE (Y, N, NS, YS, NYS) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SAMPP.** Computes the sample 100P percent point (where P is between 0.0 and 1.0, exclusively) of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1d **Usage:** CALL SAMPP(X, N, P, IWRITE, PP) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SASUM.** Compute single precision sum of absolute values of components of vector. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a3a **Usage:** R = SASUM(N, SX, INCX) **Precision:** Single (Double: DASUM) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SASUM.** Sum the absolute values of the components of a single precision vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a3a **Usage:** R = SASUM(N, SX, INCX) **Precision:** Single (Double: DASUM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SAXPY.** Compute a constant times a vector plus a vector, all single precision. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a7 **Usage:** CALL SAXPY(N, SA, SX, INCX, SY, INCY) **Precision:** Single (Double: DAXPY) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SAXPY.** Compute the scalar times a vector plus a vector, $y = ax + y$, all single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a7 **Usage:** CALL SAXPY (N, SA, SX, INCX, SY, INCY) **Precision:** Single (Double: DAXPY) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SCALE.** Discretizes the data into classes. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d, L2a **Usage:** CALL SCALE(MM, M, N, A, KL, KK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SCALE.** Computes 3 estimates of the scale (variation, scatter, dispersion) of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1b **Usage:** CALL SCALE(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SCASUM.** Compute complex sum of absolute values of components of vector. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a3a **Usage:** R = SCASUM(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCASUM.** Sum the absolute values of the real part together with the absolute values of the imaginary part of the components of a complex vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a3a **Usage:** R = SCASUM(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SCHDC.** Compute Cholesky decomposition of real positive definite matrix with optional pivoting. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SCHDC(A, LDA, P, WORK, JPVT, JOB, INFO) **Precision:** Single (Double: DCHDC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCHDD.** Downdates Cholesky factorization of real positive definite matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D7b **Usage:** CALL SCHDD(*12 parameters*) **Also see:** SCHDC **Precision:** Single (Double: DCHDD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCHEX.** Updates Cholesky factorization of real positive definite matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D7b **Usage:** CALL SCHEX(*11 parameters*) **Also see:** SCHDC **Precision:** Single (Double: DCHEX) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCHUD.** Updates Cholesky factorization of real positive definite matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D7b **Usage:** CALL SCHUD(*11 parameters*) **Also see:** SCHDC **Precision:** Single (Double: DCHUD) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCNRM2.** Compute the Euclidean length or $L_2$ norm of a complex vector. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a3b **Usage:** R = SCNRM2(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCNRM2.** Compute the Euclidean norm of a complex vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a3b **Usage:** R = SCNRM2(N, CX, INCX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SCOLR.** Sort columns of a real rectangular matrix using keys in rows. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** N6a1b, N6a2b **Usage:** CALL SCOLR (*12 parameters*) **Precision:** Single (Double: DSCOLR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SCOPY.** Copy a vector X to a vector Y, both single precision. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a5 **Usage:** CALL SCOPY(N, SX, INCX, SY, INCY) **Precision:** Single (Double: DCOPY) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SCOPY.** Copy a vector X to a vector Y, both single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a5 **Usage:** CALL SCOPY (N, SX, INCX, SY, INCY) **Precision:** Single (Double: DCOPY) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SCOPYM.** Copies negative of array SX into array SY, with corresponding increments INCX and INCY. *Portable Fortran subroutine in CMLIB (XBLAS sublibrary).* **Classes:** D1a5 **Usage:** CALL SCOPYM(N, SX, INCX, SY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SCOV.** Calculates covariance matrix for a nonlinear data fitting problem. This subprogram is intended to be used after a successful return from either of the subprograms SNLS1 or SNLS1E. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Usage:** CALL SCOV(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SCTP.** Print a scatterplot of several groups of data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3e3a **Usage:** CALL SCTP (*10 parameters*) **Precision:** Single (Double: DSCTP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SD.** Computes the sample standard deviation (with denominator N-1) of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1b **Usage:** CALL SD(X, N, IWRITE, XSD) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SDASSL.** Solves the system of differential/algebraic equations of the form $g(t, y, y') = 0$, with given initial values. *Portable Fortran subroutine in CMLIB (SDASSL sublibrary).* **Classes:** I1a1b **Usage:** CALL SDASSL(*17 parameters*) **Precision:** Single (Double: DDASSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SDDOTA.** Compute the sum of a single-precision scalar, a single-precision dot product and the double-precision accumulator, which is set to the result ACC = ACC + a + x·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** R = SDDOTA(*7 parameters*) **Precision:** Single (Double: DQDOTA) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SDDOTI.** Compute the sum of a single-precision scalar plus a single-precision dot product using a double-precision accumulator, which is set to the result ACC = a + x·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** R = SDDOTI(*7 parameters*) **Precision:** Single (Double: DQDOTI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SDOT.** Compute single precision dot product. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a4 **Usage:** R = SDOT(N, SX, INCX, SY, INCY) **Precision:** Single (Double: DDOT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SDOT.** Compute the single-precision dot product x·y. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** R = SDOT(N, SX, INCX, SY, INCY) **Precision:** Single (Double: DDOT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SDPLC.** Perform the Cox and Stuart sign test for trends in dispersion and location. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1d **Usage:** CALL SDPLC (*9 parameters*) **Precision:** Single (Double: DSDPLC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SDRIV1.** Numerical integration, initial value problems, ordinary differential equations, Gear stiff formulas. Easy to use. *Portable Fortran subroutine in CMLIB (SDRIV sublibrary).* **Classes:** I1a2, I1a1b **Usage:** CALL SDRIV1(*8 parameters*) **Precision:** Single (Double: DDRIV1) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SDRIV2.** Numerical integration, initial value problems, ordinary differential equations, Gear/Adams formulas. *Portable Fortran subroutine in CMLIB (SDRIV sublibrary).* **Classes:** I1a2, I1a1b **Usage:** CALL SDRIV2(*15 parameters*) **Precision:** Single (Double: DDRIV2) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SDRIV2.** Numerical integration, initial value problems, ordinary differential equations, Gear/Adams formulas. *Portable Fortran software in NMS library.* **Classes:** I1a1b, I1a2 **Usage:** CALL SDRIV2(*15 parameters*) **Precision:** Single (Double: DDRIV2) **Availability:** PC.

**SDRIV3.** Numerical integration, initial value problems, ordinary differential equations, implicit equations, sparse Jacobians. *Portable Fortran subroutine in CMLIB (SDRIV sublibrary).* **Classes:** I1a2, I1a1b **Usage:** CALL SDRIV3(*27 parameters*) **Precision:** Single (Double: DDRIV3) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SDSDOT.** Compute the sum of a single-precision scalar and a single-precision dot product, a + x·y, using a double-precision accumulator. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a4 **Usage:** R = SDSDOT(N, SB, SX, INCX, SY, INCY) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SDSDOT.** Compute the sum of a single-precision scalar and a single-precision dot product, a + x·y, using a double-precision accumulator. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a4 **Usage:** R = SDSDOT(N, SB, SX, INCX, SY, INCY) **Precision:** Single (Double: DQDDOT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SEARCH.** Produces and outputs tree which maximizes ALPHA×TRUE - (1.0-ALPHA)×FALSE where TRUE is the number of triads correctly predicted by the tree and FALSE is the number of triads incorrectly predicted by the tree and ALPHA is a user-defined probability. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1b **Usage:** CALL SEARCH(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SEMI-CIRCULAR PROBABILITY PLOT.** Generates a probability plot for the semicircular distribution (with density $f(x) = (2/\pi)(1+x^2)$ for x in the interval [-1,1]). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2s **Usage:** SEMI-CIRCULAR PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SEPELI.** Solves separable elliptic boundary value problems on a rectangle. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a2 **Usage:** CALL SEPELI(*26 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SEPX4.** Solves separable elliptic boundary value problems on a rectangle with constant coefficients in one direction. *Portable Fortran subroutine in CMLIB (FISHPAK sublibrary).* **Classes:** I2b1a2 **Usage:** CALL SEPX4(*22 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SET.** Create a constant vector or a vector of integers in increments of 1 or more or with other patterns. *Command in MINITAB proprietary interactive system.* **Classes:** L2e **Usage:** SET [K repetitions of] (K, ..., K) [repeated K times] into column C **Precision:** Single **Availability:** 855NOS.

**SETC.** Set a specified number of values in a complex array equal to a constant. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a1 **Usage:** CALL SETC (N, V, B) **Precision:** Single **Availability:** 855NOS, 205.

**SETD.** *Double precision version of SETR.*

**SETERR.** Sets the error indicator and depending on options prints a message and provides a dump for PORT library programs. *Proprietary Fortran subroutine in PORT.* **Classes:** R3c **Usage:** CALL SETERR (MESSG, NMESSG, NERR, IOPT) **Precision:** Single **Availability:** 855NOS, 205.

**SETI.** Set a specified number of values in an integer array equal to a constant. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a1 **Usage:** CALL SETI (N, V, B) **Precision:** Single **Availability:** 855NOS, 205.

**SETL.** Set a specified number of values in a logical array equal to a constant. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a1 **Usage:** CALL SETL (N, V, B) **Precision:** Single **Availability:** 855NOS, 205.

**SETR.** Set a specified number of values in a real array equal to a constant. *Proprietary Fortran subroutine in PORT.* **Classes:** D1a1 **Usage:** CALL SETR (N, V, B) **Precision:** Single (Double: SETD) **Availability:** 855NOS, 205.

**SGBCO.** Computes LU factorization of real band matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a2 **Usage:** CALL SGBCO(*8 parameters*) **Precision:** Single (Double: DGBCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGBDI.** Uses LU factorization of real band matrix to compute its determinant. (No provision for computing matrix inverse.) *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D3a2 **Usage:** CALL SGBDI(ABD, LDA, N, ML, MU, IPVT, DET) **Also see:** SGBCO SGBFA **Precision:** Single (Double: DGBDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGBFA.** Computes LU factorization of real band matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a2 **Usage:** CALL SGBFA(ABD, LDA, N, ML, MU, IPVT, INFO) **Precision:** Single (Double: DGBFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGBSL.** Uses LU factorization of real band matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a2 **Usage:** CALL SGBSL(ABD, LDA, N, ML, MU, IPVT, B, JOB) **Also see:** SGBCO SGBFA **Precision:** Single (Double: DGBSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGECO.** Computes LU factorization of real general matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a1 **Usage:** CALL SGECO(A, LDA, N, IPVT, RCOND, Z) **Precision:** Single (Double: DGECO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGEDI.** Uses LU factorization of real general matrix to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a1, D3a1 **Usage:** CALL SGEDI(A, LDA, N, IPVT, DET, WORK, JOB) **Also see:** SGECO SGEFA **Precision:** Single (Double: DGEDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGEEV.** Computes the eigenvalues and, optionally, the eigenvectors of a general real matrix. *Portable Fortran subroutine in CMLIB (LICEPAK sublibrary).* **Classes:** D4a2 **Usage:** CALL SGEEV(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGEFA.** Computes LU factorization of real general matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a1 **Usage:** CALL SGEFA(A, LDA, N, IPVT, INFO) **Precision:** Single (Double: DGEFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGEFS.** Factors and solves a general n-by-n single precision system of linear equations. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a1 **Usage:** CALL SGEFS(*8 parameters*) **Precision:** Single (Double: DGEFS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGEFS.** Solves n-by-n system AX=B; returns the solutions in B, the number of correct digits, and 1/cond(A). *Portable Fortran software in NMS library.* **Classes:** D2a1 **Usage:** CALL SGEFS(*9 parameters*) **Precision:** Single (Double: DGEFS) **Availability:** PC.

**SGEIR.** Factors and solves a general single precision system of linear equations and estimates solution accuracy. (Requires n-by-n extra storage.) *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a1 **Usage:** CALL SGEIR(*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGESL.** Uses LU factorization of real general matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a1 **Usage:** CALL SGESL(A, LDA, N, IPVT, B, JOB) **Also see:** SGECO SGEFA **Precision:** Single (Double: DGESL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SGLSS.** Solves linear least squares problems. Emphasis is put on detecting possible rank deficiency. Performs QR factorization using Householder transformations. Easy-to-use driver for LLSIA and ULSIA. *Portable Fortran subroutine in CMLIB (SGLSS sublibrary).* **Classes:** D9a1 **Usage:** CALL SGLSS(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SGTSL.** Factors a real tridiagonal matrix and simultaneously solves a system. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a2a **Usage:** CALL SGTSL(N, C, D, E, B, INFO) **Precision:** Single (Double: DGTSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SHI.** Hyperbolic sine integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = SHI(X) **Precision:** Single (Double: DSHI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SHOUAP.** Apply a Householder transformation in single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1b11 **Usage:** CALL SHOUAP (*9 parameters*) **Precision:** Single (Double:

DHOUAP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SHOUAP.** Apply a Householder transformation in single precision. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** D1a9 **Usage:** CALL SHOUAP (*9 parameters*) **Precision:** Single (Double: DHOUAP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SHOUTR.** Construct a Householder transformation in single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a9 **Usage:** CALL SHOUTR (N, X, K, J, V, BETA) **Precision:** Single (Double: DHOUTR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SHPROD.** Compute the Hadamard product of two single precision vectors. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL SHPROD (*7 parameters*) **Precision:** Single (Double: DHPROD) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SI.** Sine integral. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = SI(X) **Precision:** Single (Double: DSI) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SICIEI.** Sine, cosine, exponential, scaled exponential, hyperbolic sine and hyperbolic cosine integrals. *Portable Fortran software in SPECFN library.* **Classes:** C5, C6 **Usage:** CALL SICIEI(*11 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**SIGNT.** Perform a sign test of the hypothesis that a given value is a specified quantile of a distribution. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1b1 **Usage:** CALL SIGNT (*8 parameters*) **Precision:** Single (Double: DSIGNT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SIMP.** Calculates an estimate of the definite integral of a user supplied function by adaptive quadrature. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** H2a1a1 **Precision:** Single **Availability:** 855NOS.

**SIN.** sin(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = SIN(X) **Precision:** Single (Double: DSIN) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SIN.** sin(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = SIN(X) **Precision:** Single (Double: DSIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SINDG.** sin(x), x in degrees. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = SINDG(X) **Precision:** Single (Double: DSINDG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SINDG.** sin(x), x in degrees. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = SINDG(X) **Precision:** Single (Double: DSINDG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SINH.** sinh(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** R = SINH(X) **Precision:** Single (Double: DSINH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SINH.** sinh(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = SINH(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SINH.** sinh(x). *Proprietary Fortran function in PORT.* **Classes:** C4c **Usage:** R = SINH(X) **Precision:** Single (Double: DSINH) **Availability:** 855NOS, 205.

**SINQB.** Fast Fourier transform of quarter wave data. Backward fast sine transform. Performs Fourier synthesis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a3 **Usage:** CALL SINQB(N, X, WSAVE) **Also see:** SINQF SINQI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SINQF.** Computes fast Fourier transform of quarter wave data. Forward fast sine transform. Performs Fourier analysis. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a3 **Usage:** CALL SINQF(N, X, WSAVE) **Also see:** SINQB SINQI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SINQI.** Initialize array WSAVE for SUBROUTINE SINQF and SINQB. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Usage:** CALL SINQI(N, WSAVE) **Also see:** SINQF SINQB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SINT.** Computes fast Fourier sine transform of an odd sequence $X_i$. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Classes:** J1a3 **Usage:** CALL SINT(N, X, WSAVE) **Also see:** SINTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SINTI.** Initialize array WSAVE for SUBROUTINE SINT. *Portable Fortran subroutine in CMLIB (FFTPACK sublibrary).* **Usage:** CALL SINTI(N, WSAVE) **Also see:** SINT **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SKIPR.** Reads through (skips over) NLHEAD lines from input unit 5. *Fortran subroutine in DATAPAC.* **Classes:** N1 **Usage:** CALL SKIPR(NLHEAD) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SLFLT.** Perform symmetric linear filter operation with user-supplied filter coefficients; return filtered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1c2 **Usage:** CALL SLFLT (Y, N, K, H, YF, NYF) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SLICE4.** To provide the user with high-speed I/O, taking advantage of the efficiency of the system-provided subroutines Q7BUFIN and Q7BUFOUT, and of the concept of parallelism, achievable by distributing the data transfer over 4 different I/O-paths, or channels. *Fortran/meta subroutine in MAGEV.* **Classes:** N1 **Usage:** CALL SLICE4R(*9 parameters*) CALL SLICE4W(. . .) **Availability:** 205.

**SLINK.** Utilizes the single-linkage clustering algorithm to construct a tree from a user-specified distance matrix.

*Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a1 **Usage:** CALL SLINK(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SLVBLK.** Solves Ax=b where A is an almost block diagonal matrix. These arise in finite element or piecewise polynomial approximation. *Portable Fortran subroutine in CMLIB (SLVBLK sublibrary).* **Classes:** D2a2 **Usage:** CALL SLVBLK(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SMONOD.** *Double precision version of SMONOR.*

**SMONOI.** Test if an integer vector is strictly monotone increasing or decreasing. *Proprietary Fortran function in PORT.* **Classes:** R2 **Usage:** L = SMONOI(X, N, INC) **Precision:** Single **Availability:** 855NOS, 205.

**SMONOR.** Test if a real vector is strictly monotone increasing or decreasing. *Proprietary Fortran function in PORT.* **Classes:** R2 **Usage:** L = SMONOR(X, N, INC) **Precision:** Single (Double: SMONOD) **Availability:** 855NOS, 205.

**SMOOTH.** Perform either a least squares smooth or a robust smooth (Tukey's 3RSR smooth). Both the degree (zero to ten) and the width (the odd number of equally spaced points to use at each step) of the smooth can be specified. A moving average smooth is a degree one smooth. Output includes the predicted and residual values. *Command(s) in DATAPLOT interactive system.* **Classes:** L8h, L10a1c2 **Usage:** [QUADRATIC/MEDIAN/MIDMEAN/53/53H/53HT/MOVING AVERAGE] SMOOTH <RESPONSE VARIABLE> *or (for polynomial models only)* [DEGREE] <INTEGER> SMOOTH <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SMPPR.** Compute statistics for inferences regarding the population proportion and total given proportion data from a simple random sample. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a4 **Usage:** CALL SMPPR (*5 parameters*) **Precision:** Single (Double: DSMPPR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPPS.** Compute statistics for inferences regarding the population proportion and total given proportion data from a stratified random sample. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a4 **Usage:** CALL SMPPS (*7 parameters*) **Precision:** Single (Double: DSMPPS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPRR.** Compute statistics for inferences regarding the population mean and total using ratio or regression estimation, or inferences regarding the population ratio given a simple random sample. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b5 **Usage:** CALL SMPRR (*10 parameters*) **Precision:** Single (Double: DSMPRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPRS.** Compute statistics for inferences regarding the population mean and total using ratio or regression estimation given continuous data from a stratified random sample. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b5 **Usage:** CALL SMPRS (*20 parameters*) **Precision:** Single (Double: DSMPRS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPSC.** Compute statistics for inferences regarding the population mean and total using single stage cluster sampling with continuous data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a4 **Usage:** CALL SMPSC (*14 parameters*) **Precision:** Single (Double: DSMPSC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPSR.** Compute statistics for inferences regarding the population mean and total, given data from a simple random sample. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a4 **Usage:** CALL SMPSR (*8 parameters*) **Precision:** Single (Double: DSMPSR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPSS.** Compute statistics for inferences regarding the population mean and total, given data from a stratified random sample. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a4 **Usage:** CALL SMPSS (*11 parameters*) **Precision:** Single (Double: DSMPSS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMPST.** Compute statistics for inferences regarding the population mean and total given continuous data from a two-stage sample with equisized primary units. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a4 **Usage:** CALL SMPST (*11 parameters*) **Precision:** Single (Double: DSMPST) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SMSNO.** Minimize a general unconstrained objective function using finite difference gradients and secant Hessian approximations. *Portable Fortran subroutine in CMLIB (NL2SN sublibrary).* **Classes:** G1b1a **Usage:** CALL SMSNO(*11 parameters*) **Precision:** Single (Double: DSMSNO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SNBCO.** Factors a real band matrix by Gaussian elimination and estimates condition of the matrix. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a2 **Usage:** CALL SNBCO(*8 parameters*) **Precision:** Single (Double: DNBCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNBDI.** Computes the determinant of a single precision band matrix using factors previously computed. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D3a2 **Usage:** CALL SNBDI(ABE, LDA, N, ML, MU, IPVT, DET) **Also see:** SNBCO SNBFA **Precision:** Single (Double: DNBDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNBFA.** Factors a single precision band matrix by elimination. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a2 **Usage:** CALL SNBFA(ABE, LDA, N, ML, MU, IPVT, INFO) **Precision:** Single (Double:

DNBFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNBFS.** Factors and solves a general nonsymmetric single precision banded system of linear equations. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a2 **Usage:** CALL SNBFS(*10 parameters*) **Precision:** Single (Double: DNBFS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNBIR.** Factors and solves a general nonsymmetric single precision banded system of equations and estimates solution accuracy (needs N*(2*ML+MU) extra storage). *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a2 **Usage:** CALL SNBIR(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNBSL.** Solves a general nonsymmetric single precision banded system of linear equations using factors computed previously. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2a2 **Usage:** CALL SNBSL(ABE, LDA, N, ML, MU, IPVT, B, JOB) **Also see:** SNBCO SNBFA **Precision:** Single (Double: DNBSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNKMC.** Perform Student-Newman-Keuls multiple comparison test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L7a1 **Usage:** CALL SNKMC (*7 parameters*) **Precision:** Single (Double: DSNKMC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SNLS1.** Minimizes the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. Flexible usage, including various options for providing Jacobian. Covariance matrix is available via the subroutine SCOV. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Classes:** K1b1a1, K1b1a2 **Usage:** CALL SNLS1(*26 parameters*) **Also see:** SCOV, CHKDER checks user's Jacobian routine if desired **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SNLS1E.** Minimizes the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. An easy to use driver for SNLS1. The covariance matrix is available by calling the subroutine SCOV. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Classes:** K1b1a1, K1b1a2 **Usage:** CALL SNLS1E(*12 parameters*) **Also see:** SCOV, CHKDER checks user's Jacobian routine if desired **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SNRM2.** Compute the Euclidean length or $L_2$ norm of a single precision vector, without underflow or overflow. *Portable Fortran function in CMLIB (BLAS sublibrary).* **Classes:** D1a3b **Usage:** R = SNRM2(N, SX, INCX) **Precision:** Single (Double: DNRM2) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SNRM2.** Compute the Euclidean length or $L_2$ norm of a single precision vector, without underflow or overflow. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a3b **Usage:** R = SNRM2(N, SX, INCX) **Precision:** Single (Double: DNRM2) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SNRM2.** Finds the length (Euclidean norm) of a vector, without underflow or overflow. *Proprietary Fortran function in PORT.* **Classes:** D1a3b **Usage:** R = SNRM2(N, X, INCX) **Precision:** Single (Double: DNRM2) **Availability:** 855NOS, 205.

**SNRNK.** Perform a Wilcoxon signed rank test. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4a1b1 **Usage:** CALL SNRNK (NOBS, Y, FUZZ, STAT, NMISS) **Precision:** Single (Double: DSNRNK) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SNSQ.** Finds a zero of a system of N nonlinear equations in N variables by a modification of the Powell hybrid method. Flexible usage. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Classes:** F2 **Usage:** CALL SNSQ(*27 parameters*) **Also see:** CHKDER checks user's Jacobian routine if desired. **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SNSQE.** Finds a zero of a system of N nonlinear equations in N variables by a modification of Powell's hybrid method. An easy to use driver for SNSQ. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Classes:** F2 **Usage:** CALL SNSQE(*11 parameters*) **Also see:** CHKDER if user supplies Jacobian this checks for consistency. **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SNSQE.** Solves N nonlinear equations given in SUBROUTINE FCN(N,X,FVEC, IFLAG) to accuracy TOL. If IOPT=2 then Jacobian subroutine JAC need not be supplied. Needs EXTERNAL statement for user selected subroutine name. NMPINT=0 for simple use. Returns solution in X(N), residuals in FVEC(N). *Portable Fortran software in NMS library.* **Classes:** F2 **Usage:** CALL SNSQE(*11 parameters*) **Precision:** Single (Double: DNSQE) **Availability:** PC.

**SODR.** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are preset, and a two-part report of the results is automatically generated. *Portable Fortran subroutine in CMLIB (ODRPACK sublibrary).* **Classes:** K1b1a2, L8a4, L8c5, L8e5 **Usage:** CALL SODR(*16 parameters*) **Precision:** Single (Double: DODR) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SODRC.** Computes the weighted orthogonal distance regression or ordinary linear or nonlinear least squares solution. Derivatives are either supplied by the user or numerically approximated. Control values are supplied by the user, and a three-part report of the results is optionally generated. *Portable Fortran subroutine in CMLIB (ODR-*

*PACK sublibrary).* **Classes:** K1a1a3, K1b1a1, K1b1a2, L8a4, L8c5, L8e5 **Usage:** CALL SODRC(*31 parameters*) **Precision:** Single (Double: DODRC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SODS.** Solves an overdetermined system of linear equations. For full rank matrices the unique least squares solution is provided. The least squares solution of minimal length can be obtained in the rank deficient case. *Portable Fortran subroutine in CMLIB (SUDSODS sublibrary).* **Classes:** D9a1 **Usage:** CALL SODS(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SORT.** Sorts (in ascending order) the N elements of the vector X and puts the resulting N sorted values into the vector Y. *Fortran subroutine in DATAPAC.* **Classes:** N6a2b **Usage:** CALL SORT(X, N, Y) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SORT.** Sorts a vector in ascending order and optionally carries along other vectors. *Command in MINITAB proprietary interactive system.* **Classes:** N6a2b **Usage:** SORT the values in C [carry along corresponding rows of C, ..., C] put into C [corresponding rows into C, ..., C] **Precision:** Single **Availability:** 855NOS.

**SORT.** Sorts observations by one or more variables in ascending or descending order. Options: handle duplicate records, maintain order of observations with identical values of the sorting variables, several collating sequences. *Proprietary stand-alone program using SAS command language.* **Classes:** N6a2a, N6a2b, N6a2c **Precision:** Single **Availability:** CAMVAX.

**SORT CASES.** Sorts numeric or alphanumeric data in either ascending or descending order according to one or more variables. *Proprietary stand-alone program using SPSS command language.* **Classes:** N6a2 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**SORTC.** Sorts (in ascending order) the N elements of the vector X and rearranges the elements of the vector Y. *Fortran subroutine in DATAPAC.* **Classes:** N6a2b **Usage:** CALL SORTC(X, Y, N, XS, YC) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SORTP.** Sorts (in ascending order) the N elements of the vector X, puts the resulting N sorted values into the vector Y, and puts the position (in the original vector X) of each of the sorted values into the single precision vector XPOS. *Fortran subroutine in DATAPAC.* **Classes:** N6a2b **Usage:** CALL SORTP(X, N, Y, XPOS) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SORTR.** To sort a linearly stored set of fixed length records, or equivalently, the columns of a matrix, into a nondecreasing (ascending) order with respect to the algebraic values of the elements of a separate real or integer array known as the key. *Fortran/meta subroutine in MAGEV.* **Classes:** N6a2a, N6a2b **Usage:** CALL SORTR(X, LR, NR, KEY, NRS) **Precision:** Single **Availability:** 205 (vectorized)

**SOS.** Finds a zero of a system of N nonlinear equations in N unknowns using Brown's method. *Portable Fortran subroutine in CMLIB (SNLS1E sublibrary).* **Classes:** F2 **Usage:** CALL SOS(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SPARSE.** Approximates the covariance matrix by BB(trans) where B contains many zeros and corresponds to the loading matrix for factor analysis where clusters of variables are the factors. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L13a, L14a1a2 **Usage:** CALL SPARSE(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SPBCO.** Uses Cholesky algorithm to compute factorization of real positive definite band matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b2 **Usage:** CALL SPBCO(ABD, LDA, N, M, RCOND, Z, INFO) **Precision:** Single (Double: DPBCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPBDI.** Uses factorization of real positive definite band matrix to compute its determinant. (No provision for matrix inverse.) *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D3b2 **Usage:** CALL SPBDI(ABD, LDA, N, M, DET) **Also see:** SPBCO SPBFA **Precision:** Single (Double: DPBDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPBFA.** Uses Cholesky algorithm to compute factorization of real positive definite band matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b2 **Usage:** CALL SPBFA(ABD, LDA, N, M, INFO) **Precision:** Single (Double: DPBFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPBSL.** Uses factorization of real positive definite band matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b2 **Usage:** CALL SPBSL(ABD, LDA, N, M, B) **Also see:** SPBCO SPBFA **Precision:** Single (Double: DPBSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPCORR.** Computes the Spearman rank correlation coefficient between the two sets of data in the input vectors X and Y. *Fortran subroutine in DATAPAC.* **Classes:** L1b **Usage:** CALL SPCORR(X, Y, N, IWRITE, SPC) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SPECTRUM.** Plots the spectrum (smoothed Fourier transform of the autocorrelation function) for univariate time series data which are equi-spaced (in time, for example), or, for bivariate time series, plots the cross- , co- , quadrature, coherency, phase, or gain spectrum plots. *Command(s) in DATAPLOT interactive system.* **Classes:** L10a3a4, L10b3a4, L10b3a6 **Usage:** [CROSS/CO -/QUADRATURE/COHERENCY/PHASE/GAIN] SPEC-

TRUM <RESPONSE VARIABLE> [<RESPONSE VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SPENC.** Integral from 0 to x of ln(|1-t|)/t. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C5 **Usage:** R = SPENC(X) **Precision:** Single (Double: DSPENC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SPENC.** Evaluate Spence's integral, see [AMS55 (27.7.1)]. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C5 **Usage:** R = SPENC(X) **Precision:** Single (Double: DSPENC) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SPISC1.** Computes a shape preserving polynomial spline of arbitrary degree to a given set of data. *Portable Fortran software in JCAM library.* **Classes:** E1a **Usage:** CALL SPISC1(*13 parameters*) **Precision:** Double **Availability:** 855NOS (In source form only.)

**SPLINE.** Calculates the coefficients of a spline interpolant to given data. Usually used in conjunction with SEVAL to subsequently evaluate the spline at arbitrary points. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** E1a **Precision:** Single **Availability:** 855NOS.

**SPLINE FIT.** Performs a least squares spline fit. The user can specify both the degree of the piecewise polynomial (linear through tenth degree, default is cubic) and the knot points (i.e., the regions where distinct polynomials are fit). Both the predicted and the residual values are available for further analysis and plotting. *Command(s) in DATAPLOT interactive system.* **Classes:** L8g **Usage:** <DEGREE> SPLINE FIT <RESPONSE VARIABLE> <INDEPENDENT VARIABLE> [<KNOTS VARIABLE >] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SPLIT1.** Splits cases in each variable until all within-cluster variances are smaller than a user-specified threshold. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a2 **Usage:** CALL SPLIT1(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SPLIT2.** Splits matrix of case-by-variable data values into blocks until all within-block variances are less than a given threshold. Includes user-controlled constraints. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14b **Usage:** CALL SPLIT2(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SPLN1.** Evaluates a function and derivatives described previously by an expansion in terms of B-splines. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL SPLN1 (K, T, N, A, X, NX, ID, NID, FX) **Precision:** Single (Double: DSPLN1) **Availability:** 855NOS, 205.

**SPLN2.** Evaluates a function described by a previously determined expansion in B-splines. More flexible than SPLN1. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL SPLN2 (*13 parameters*) **Precision:** Single (Double: DSPLN2) **Availability:** 855NOS, 205.

**SPLND.** Evaluates at a given set of points a function described by a previously determined expansion in terms of B-splines. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a1, E3a2, K6a1, K6a2 **Usage:** CALL SPLND (K, T, N, A, X, NX, MD, FX) **Precision:** Single (Double: DSPLND) **Availability:** 855NOS, 205.

**SPLNE.** Evaluates at a set of points, a function described by a previously determined expansion in terms of B-splines. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a1, K6a1 **Usage:** CALL SPLNE (K, T, N, A, X, NX, FX) **Precision:** Single (Double: DSPLNE) **Availability:** 855NOS, 205.

**SPLNI.** Integrates a function described previously by an expansion in terms of B-splines. Several integrations can be performed in one call. *Proprietary Fortran subroutine in PORT.* **Classes:** E3a3, H2a2b1, K6a3 **Usage:** CALL SPLNI (K, T, N, A, X, NX, FIX) **Precision:** Single (Double: DSPLNI) **Availability:** 855NOS, 205.

**SPLP.** Solves linear optimization problems, that is, it minimizes the linear function (Transpose of costs) × X subject to AX=W, where the entries of the vectors X and W may have simple upper or lower bounds. Uses a sparse storage mode for the matrix A and out-of-core scratch storage. *Portable Fortran subroutine in CMLIB (SPLP sublibrary).* **Classes:** G2a2 **Usage:** CALL SPLP(*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SPOCO.** Uses Cholesky algorithm to factor real positive definite matrix and estimate its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SPOCO(A, LDA, N, RCOND, Z, INFO) **Precision:** Single (Double: DPOCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPODI.** Uses factorization of real positive definite matrix to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b, D3b1b **Usage:** CALL SPODI(A, LDA, N, DET, JOB) **Also see:** SPOCO SPOFA **Precision:** Single (Double: DPODI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPOFA.** Uses Cholesky algorithm to factor real positive definite matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SPOFA(A, LDA, N, INFO) **Precision:** Single (Double: DPOFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPOFS.** Factors and solves a symmetric positive definite single precision system of linear equations. *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2b1b **Usage:** CALL SPOFS(A, LDA, N, V, ITASK, IND, WORK) **Precision:** Single (Double: DPOFS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPOIR.** Factors and solves a symmetric positive definite single precision system of equations and estimates solution accuracy. (Needs n-by-n extra storage.) *Portable Fortran subroutine in CMLIB (LINDRV sublibrary).* **Classes:** D2b1b **Usage:** CALL SPOIR(A, LDA, N, V, ITASK, IND, WORK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPOSL.** Uses factorization of real positive definite matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SPOSL(A, LDA, N, B) **Also see:** SPOCO SPOFA **Precision:** Single (Double: DPOSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPP.** Print Y versus X scatterplot with individual plot symbols specified by user; linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3b **Usage:** CALL SPP (Y, X, N, ISYM) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SPPC.** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axes; user-supplied control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3b **Usage:** CALL SPPC (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SPPCO.** Uses Cholesky algorithm to factor real positive definite matrix stored in packed form and estimate its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SPPCO(AP, N, RCOND, Z, INFO) **Precision:** Single (Double: DPPCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPPDI.** Uses factorization of real positive definite matrix stored in packed form to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b, D3b1b **Usage:** CALL SPPDI(AP, N, DET, JOB) **Also see:** SPPCO SPPFA **Precision:** Single (Double: DPPDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPPFA.** Uses Cholesky algorithm to factor real positive definite matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SPPFA(AP, N, INFO) **Precision:** Single (Double: DPPFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPPL.** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3b **Usage:** CALL SPPL (Y, X, N, ISYM, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SPPM.** Print Y versus X scatterplot with individual plot symbols specified by user; linear axes; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3b **Usage:** CALL SPPM (Y, YMISS, X, XMISS, N, ISYM) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SPPMC.** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axes; user-supplied control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3b **Usage:** CALL SPPMC (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SPPML.** Print Y versus X scatterplot with individual plot symbols specified by user; log or linear axis; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3b3b **Usage:** CALL SPPML (*7 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SPPSL.** Uses factorization of real positive definite matrix stored in packed form to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1b **Usage:** CALL SPPSL(AP, N, B) **Also see:** SPPCO SPPFA **Precision:** Single (Double: DPPSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPRDCT.** Multiply the components of a single precision vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** R = SPRDCT(N, SX, INCX) **Precision:** Single (Double: DPRDCT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SPTSL.** Decomposes real symmetric positive definite tridiagonal matrix and simultaneously solves a system. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b2a **Usage:** CALL SPTSL(N, D, E, B) **Precision:** Single (Double: DPTSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SPWF.** Compute the Wiener forecast operator for a stationary stochastic process. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a2c **Usage:** CALL SPWF (*9 parameters*) **Precision:** Single (Double: DSPWF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SQRANK.** For solving linear systems in least squares sense. Computes the QR decomposition of matrix using LINPACK subroutines. *Portable Fortran subroutine in CMLIB (SQRLSS sublibrary).* **Classes:** D5 **Usage:** CALL SQRANK(*9 parameters*) **Precision:** Single (Double: DQRANK) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SQRDC.** Computes QR decomposition of real general matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D5 **Usage:** CALL SQRDC(*8 parameters*) **Precision:** Single (Double: DQRDC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SQRLS.** Solves m-by-n system Ax=b in least squares sense. *Portable Fortran software in NMS library.* **Classes:** D9a1 **Usage:** CALL SQRLS(*14 parameters*) **Precision:** Single (Double: DQRLS) **Availability:** PC.

**SQRLSS.** For solving linear systems in least squares sense. Finds solution and residual after matrix factored by SQRANK. *Portable Fortran subroutine in CMLIB (SQRLSS sublibrary).* **Classes:** D9a1 **Usage:** CALL SQRLSS(*10 parameters*) **Also see:** SQRANK **Precision:** Single (Double: DQRLSS) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SQRSL.** Applies the output of SQRDC to compute coordinate transformations, projections, and least squares solutions (general real matrix). *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a1, D9a1 **Usage:** CALL SQRSL(*13 parameters*) **Also see:** SQRDC **Precision:** Single (Double: DQRSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SQRT.** Square root of a real argument. *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C2 **Usage:** R = SQRT(X) **Precision:** Single (Double: DSQRT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SQRT.** Square root of a real argument. *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C2 **Usage:** R = SQRT(X) **Precision:** Single (Double: DSQRT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SRCH.** Search a sorted vector for a given scalar and return its index. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N5c **Usage:** CALL SRCH (N, VALUE, X, INCX, INDEX) **Precision:** Single (Double: DSRCH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SROT.** Apply Givens plane rotation to a single precision vector. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a8 **Usage:** CALL SROT(N, SX, INCX, SY, INCY, SC, SS) **Precision:** Single (Double: DROT) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SROT.** Apply Givens plane rotation to a single precision vector. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a8 **Usage:** CALL SROT (N, SX, INCX, SY, INCY, C, S) **Precision:** Single (Double: DROT) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SROTG.** Construct Givens plane rotation of single precision matrix. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1b10 **Usage:** CALL SROTG(SA, SB, SC, SS) **Precision:** Single (Double: DROTG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SROTG.** Construct Givens plane rotation of a single precision matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a8 **Usage:** CALL SROTG (SA, SB, SC, SS) **Precision:** Single (Double: DROTG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SROTM.** Apply modified Givens plane rotation to single precision vector. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a8 **Usage:** CALL SROTM(N, SX, INCX, SY, INCY, SPARAM) **Precision:** Single (Double: DROTM) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SROTM.** Apply modified Givens plane rotation to single precision vector. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a8 **Usage:** CALL SROTM (*6 parameters*) **Precision:** Single (Double: DROTM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SROTMG.** Construct modified Givens plane rotation of single precision matrix. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1b10 **Usage:** CALL SROTMG(SD1, SD2, SB1, SB2, SPARAM) **Precision:** Single (Double: DROTMG) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SROTMG.** Construct modified Givens plane rotation of a single precision matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a8 **Usage:** CALL SROTMG (SD1, SD2, SX1, SY1, SPARAM) **Precision:** Single (Double: DROTMG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SRTAD.** *Double precision version of SRTAR.*

**SRTAH.** Actively sorts Hollerith data into ascending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a2c **Usage:** CALL SRTAH (A, L, INC, N) **Precision:** Single **Availability:** 855NOS, 205.

**SRTAI.** Actively sorts integer data into ascending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a2a **Usage:** CALL SRTAI (A, INT, N) **Precision:** Single **Availability:** 855NOS, 205.

**SRTAR.** Actively sorts real data into ascending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a2b **Usage:** CALL SRTAR (A, INT, N) **Precision:** Single (Double: SRTAD) **Availability:** 855NOS, 205.

**SRTDD.** *Double precision version of SRTDR.*

**SRTDH.** Actively sorts Hollerith data into descending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a2c **Usage:** CALL SRTDH (A, L, INC, N) **Precision:** Single **Availability:** 855NOS, 205.

**SRTDI.** Actively sorts integer data into descending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a2a

**Usage:** CALL SRTDI (A, INT, N) **Precision:** Single **Availability:** 855NOS, 205.

**SRTDR.** Actively sorts real data into descending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a2b **Usage:** CALL SRTDR (A, INT, N) **Precision:** Single (Double: SRTDD) **Availability:** 855NOS, 205.

**SRTPAD.** *Double precision version of SRTPAR.*

**SRTPAH.** Passively sorts Hollerith data into ascending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a1c **Usage:** CALL SRTPAH (A, L, INC, N) **Also see:** SRTRH **Precision:** Single **Availability:** 855NOS, 205.

**SRTPAI.** Passively sorts integer data into ascending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a1a **Usage:** CALL SRTPAI (A, INTA, IP, INTP, N) **Also see:** SRTRI **Precision:** Single **Availability:** 855NOS, 205.

**SRTPAR.** Passively sorts real data into ascending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a1b **Usage:** CALL SRTPAR (A, INTA, IP, INTP, N) **Also see:** SRTRR **Precision:** Single (Double: SRTPAD) **Availability:** 855NOS, 205.

**SRTPDD.** *Double precision version of SRTPDR.*

**SRTPDH.** Passively sorts Hollerith data into descending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a1c **Usage:** CALL SRTPDH (A, L, INC, N) **Also see:** SRTRH **Precision:** Single **Availability:** 855NOS, 205.

**SRTPDI.** Passively sorts integer data into descending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a1a **Usage:** CALL SRTPDI (A, INTA, IP, INTP, N) **Also see:** SRTRI **Precision:** Single **Availability:** 855NOS, 205.

**SRTPDR.** Passively sorts real data into descending order. *Proprietary Fortran subroutine in PORT.* **Classes:** N6a1b **Usage:** CALL SRTPDR (A, INTA, IP, INTP, N) **Also see:** SRTRR **Precision:** Single (Double: SRTPDD) **Availability:** 855NOS, 205.

**SRTRD.** *Double precision version of SRTRR.*

**SRTRH.** Rearranges Hollerith data according to permutation stored in IP. *Proprietary Fortran subroutine in PORT.* **Classes:** N8 **Usage:** CALL SRTRH (A, L, INTA, IP, INTP, N) **Precision:** Single **Availability:** 855NOS, 205.

**SRTRI.** Rearranges integer data according to permutation stored in IP. *Proprietary Fortran subroutine in PORT.* **Classes:** N8 **Usage:** CALL SRTRI (A, INTA, IP, INTP, N) **Precision:** Single **Availability:** 855NOS, 205.

**SRTRR.** Rearranges real data according to permutation stored in IP. *Proprietary Fortran subroutine in PORT.* **Classes:** N8 **Usage:** CALL SRTRR (A, INTA, IP, INTP, N) **Precision:** Single (Double: SRTRD) **Availability:** 855NOS, 205.

**SSCAL.** Multiply a vector by a scalar, y =ay, both single precision. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a6 **Usage:** CALL SSCAL(N, SA, SX, INCX) **Precision:** Single (Double: DSCAL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSCAL.** Multiply a vector by a scalar, y =ay, both single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a6 **Usage:** CALL SSCAL (N, SA, SX, INCX) **Precision:** Single (Double: DSCAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSET.** Set the components of a vector to a scalar, all single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a1 **Usage:** CALL SSET (N, SA, SX, INCX) **Precision:** Single (Double: DSET) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSICO.** Computes factorization of real symmetric indefinite matrix and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a **Usage:** CALL SSICO(A, LDA, N, KPVT, RCOND, Z) **Precision:** Single (Double: DSICO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSIDI.** Uses factorization of real symmetric indefinite matrix to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a, D3b1a **Usage:** CALL SSIDI(*8 parameters*) **Also see:** SSICO SSIFA **Precision:** Single (Double: DSIDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSIEV.** Computes the eigenvalues and, optionally, the eigenvectors of a real symmetric matrix. *Portable Fortran subroutine in CMLIB (LICEPAK sublibrary).* **Classes:** D4a1 **Usage:** CALL SSIEV(A, LDA, N, E, WORK, JOB, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSIFA.** Computes factorization of real symmetric indefinite matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a **Usage:** CALL SSIFA(A, LDA, N, KPVT, INFO) **Precision:** Single (Double: DSIFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSISL.** Uses factorization of real symmetric indefinite matrix to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a **Usage:** CALL SSISL(A, LDA, N, KPVT, B) **Also see:** SSICO SSIFA **Precision:** Single (Double: DSISL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSORT.** Sorts an array X (of N real numbers) into increasing or decreasing order. An optional array Y is carried along with X. *Portable Fortran subroutine in CMLIB (SSORT sublibrary).* **Classes:** N6a2b **Usage:** CALL SSORT(X, Y, N, KFLAG) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SSPAND.** Simple random sampling with proportion data – inferences regarding the population proportion and total. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a4 **Usage:** CALL SSPAND (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSPBLK.** Stratified random sampling with proportion data – inferences regarding the population proportion and total. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a4 **Usage:** CALL SSPBLK (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSPCO.** Computes factorization of real symmetric indefinite matrix stored in packed form and estimates its condition. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a **Usage:** CALL SSPCO(AP, N, KPVT, RCOND, Z) **Precision:** Single (Double: DSPCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSPDI.** Uses factorization of real symmetric indefinite matrix stored in packed form to compute its determinant and/or inverse. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a, D3b1a **Usage:** CALL SSPDI(*7 parameters*) **Also see:** SSPCO SSPFA **Precision:** Single (Double: DSPDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSPEV.** Computes eigenvalues and, optionally eigenvectors of real symmetric matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LICEPAK sublibrary).* **Classes:** D4a1 **Usage:** CALL SSPEV(A, N, E, V, LDV, WORK, JOB, INFO) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSPFA.** Computes factorization of real symmetric indefinite matrix stored in packed form. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a **Usage:** CALL SSPFA(AP, N, KPVT, INFO) **Precision:** Single (Double: DSPFA) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSPSL.** Uses factorization of real symmetric indefinite matrix stored in packed form to solve systems. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2b1a **Usage:** CALL SSPSL(AP, N, KPVT, B) **Also see:** SSPCO SSPFA **Precision:** Single (Double: DSPSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSRAND.** Simple random sampling with continuous data – inferences regarding the population mean and total using ratio or regression estimation. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b5 **Usage:** CALL SSRAND (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSRBLK.** Stratified random sampling with continuous data-inferences regarding the population mean and total using ratio or regression estimation. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4b5 **Usage:** CALL SSRBLK (*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSRCH.** Search a character vector, sorted in ascending ASCII order, for a given string and return its index. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N5c **Usage:** CALL SSRCH (N, STRING, CHX, INCX, INDEX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSSAND.** Simple random sampling with continuous data – inferences regarding the population mean and total. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a4 **Usage:** CALL SSSAND (Y, NBR, ALPHA, TEMP, STAT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSSBLK.** Stratified random sampling with continuous data – inferences regarding the population mean and total. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a4 **Usage:** CALL SSSBLK (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSSCAN.** Single stage cluster sampling with continuous data – inferences regarding the population mean and total. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a4 **Usage:** CALL SSSCAN (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSSEST.** Two-stage sampling with continuous data and equisized primary units – inferences regarding the population mean and total. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a4 **Usage:** CALL SSSEST (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSUB.** Subtract each component of a vector from a scalar, x = a − x, all single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** CALL SSUB (N, SA, SX, INCX) **Precision:** Single (Double: DSUB) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSUM.** Sum the values of a single precision vector. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** R = SSUM(N, SX, INCX) **Precision:** Single (Double: DSUM) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSVDC.** Computes the singular value decomposition of a real n-by-p matrix X, dimensioned X(LDX,P). Has options to allow computation of only the singular values, or singular values and associated decomposition matrices. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D6 **Usage:** CALL SSVDC(*13 parameters*) **Precision:** Single (Double: DSVDC) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSVDC.** Computes the singular value decomposition of a real n-by-p matrix X, dimensioned X(LDX,P). Has options to allow computation of only the singular values, or singular values and associated decomposition matrices. *Portable*

*Fortran software in NMS library.* **Classes:** D6 **Usage:** CALL SSVDC(*13 parameters*) **Precision:** Single (Double: DSVDC) **Availability:** PC.

**SSWAP.** Interchange vectors X and Y, both single precision. *Portable Fortran subroutine in CMLIB (BLAS sublibrary).* **Classes:** D1a5 **Usage:** CALL SSWAP(N, SX, INCX, SY, INCY) **Precision:** Single (Double: DSWAP) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SSWAP.** Interchange vectors X and Y, both single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a5 **Usage:** CALL SSWAP (N, SX, INCX, SY, INCY) **Precision:** Single (Double: DSWAP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSWD.** Estimate the nonnormalized spectral density of a stationary time series using a spectral window given the time series data. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a3a3 **Usage:** CALL SSWD (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SSWP.** Estimate the nonnormalized spectral density of a stationary time series using a spectral window given the periodogram. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10a3a3 **Usage:** CALL SSWP (N, PX, NF, F, ISWVER, M, SX) **Precision:** Single (Double: DSSWP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**STAND.** Standardizes a data matrix such that each variable has mean zero and unit variance. Missing values are ignored. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d, L2a **Usage:** CALL STAND(MM, M, N, A) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**STANDARD.** Standardizes some or all of the variables in a data set to a given mean and standard deviation. Options: weights, replace missing values with variable mean. *Proprietary stand-alone program using SAS command language.* **Classes:** L2a **Precision:** Single **Availability:** CAMVAX.

**STAR PLOT.** Generate a star plot. *Command(s) in DATAPLOT interactive system.* **Classes:** L3e4 **Usage:** STAR PLOT <VARIABLE 1> ... <VARIABLE K> SUBSET <VARIABLE> <SUBSET VALUE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**STAT.** Compute and print 53 statistics describing the input data. *Portable Fortran subroutine in STARPAC.* **Classes:** L4aL4a1d **Usage:** CALL STAT (Y, N, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**STATISTICS PLOT.** For one of over 30 location, dispersion, shape, EDA, correlation, regression, time series, and quality control statistics, generate a plot with the vertical axis containing the value of the statistic for each distinct subset of another variable while the horizontal axis is the subset identifier. Common statistics are the mean, the standard deviation, the linear-intercept, the linear-slope, the linear RESSD, and the linear-correlation. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a3, L3b4 **Usage:** <STATISTIC> PLOT <VARIABLE> <SUBSET VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**STATS.** Compute and optionally print 53 statistics describing input data; return statistics. *Portable Fortran subroutine in STARPAC.* **Classes:** L1a1, L4a1a4n, L4a1d **Usage:** CALL STATS (Y, N, LDSTAK, STS, NPRT) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**STATW.** Compute and print 53 statistics describing weighted input data. *Portable Fortran subroutine in STARPAC.* **Classes:** L1a1, L4a1a4n, L4a1d **Usage:** CALL STATW (Y, WT, N, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**STATWS.** Compute and optionally print 53 statistics describing weighted input data; return statistics. *Portable Fortran subroutine in STARPAC.* **Classes:** L1a1, L4a1a4n, L4a1d **Usage:** CALL STATWS (*6 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**STEM AND LEAF DIAGRAM.** Generates a stem and leaf plot of a one-dimensional data set. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a3 **Usage:** STEM AND LEAF PLOT <VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**STEM-&-LEAF.** Prints stem-and-leaf display(s), optionally with no trimming of outliers. *Command in MINITAB proprietary interactive system.* **Classes:** L3a3 **Usage:** STEM-and-leaf display of C, ..., C [; subcommand NOTRIM.] **Precision:** Single **Availability:** 855NOS.

**STEPDISC.** Performs a stepwise discriminant analysis by forward selection, backward elimination, or stepwise selection of variables. The classes are assumed to be multivariate normal with a common covariance matrix. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L12 **Precision:** Single **Availability:** CAMVAX.

**STEPWISE.** Performs stepwise linear regression using forward selection, backward elimination, conventional stepwise, or user intervention. Options available through subcommands: F-to-enter and F-to-remove, force and remove sets of variables, print next "best" (by the F-statistic) K alternatives. *Command in MINITAB proprietary interactive system.* **Classes:** L8c1a1 **Usage:** STEPwise regression of y in C, predictors in C, ...C [; subcommands FENTER = K; FREMOVE = K; FORCE C, ..., C; ENTER C, ..., C; REMOVE C, ..., C; BEST K; STEPS = K.] **Precision:** Single **Availability:** 855NOS.

**STEPWISE.** Provides five methods (forward selection, backward elimination, stepwise, maximum and minimum

R-squared improvements) for stepwise regression. Options: weights, fsignificance levels, Mallows' C-p statistic. *Proprietary stand-alone program using SAS command language.* **Classes:** L8c1a1 **Precision:** Single **Availability:** CAMVAX.

**STMLP.** Print a stem-and-leaf plot. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3a3 **Usage:** CALL STMLP (NOBS, X, UNIT, TITLE) **Precision:** Single (Double: DSTMLP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**STMOM3.** Computes the sample standardized third central moment of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1c **Usage:** CALL STMOM3(X, N, IWRITE, XSMOM3) **Precision:** Single **Availability:** 855NOS, 840NOS.

**STMOM4.** Computes the sample standardized fourth central moment of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1c **Usage:** CALL STMOM4(X, N, IWRITE, XSMOM4) **Precision:** Single **Availability:** 855NOS, 840NOS.

**STPLS.** Select optimum step size for computing finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by default values. *Portable Fortran subroutine in STARPAC.* **Classes:** L8i **Usage:** CALL STPLS (*9 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**STPLSC.** Select optimum step size for computing finite difference derivatives. Features detailed documentation, extensive error checking, and comprehensive output. Computations and reports controlled by user. *Portable Fortran subroutine in STARPAC.* **Classes:** L8i **Usage:** CALL STPLSC (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**STRCO.** Estimates the condition of real triangular matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a3 **Usage:** CALL STRCO(T, LDT, N, RCOND, Z, JOB) **Precision:** Single (Double: DTRCO) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**STRDI.** Computes determinant and/or inverse of real triangular matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a3, D3a3 **Usage:** CALL STRDI(T, LDT, N, DET, JOB, INFO) **Precision:** Single (Double: DTRDI) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**STRSL.** Solves systems with real triangular matrix. *Portable Fortran subroutine in CMLIB (LINPAKS sublibrary).* **Classes:** D2a3 **Usage:** CALL STRSL(T, LDT, N, B, JOB, INFO) **Precision:** Single (Double: DTRSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SUBSE1.** Carries over into Y all observations of vector X for which the corresponding elements in vector D are in the interval [DMIN,DMAX]. *Fortran subroutine in DATAPAC.* **Classes:** L2c **Usage:** CALL SUBSE1(X, N, D, DMIN, DMAX, Y, NY) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SUBSE2.** Carries over into Y all observations of vector X for which the corresponding elements in vector D1 are in the inclusive interval [D1MIN,D1MAX] and also for which the corresponding elements in D2 are in the interval [D2MIN,D2MAX]. *Fortran subroutine in DATAPAC.* **Classes:** L2c **Usage:** CALL SUBSE2(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SUBSET.** Retain all observations in vector X for which the corresponding elements in vector D are in the interval [DMIN,DMAX]. *Fortran subroutine in DATAPAC.* **Classes:** L2c **Usage:** CALL SUBSET (X, N, D, DMIN, DMAX, NEWN) **Precision:** Single **Availability:** 855NOS, 840NOS.

**SUDS.** Solves underdetermined systems of linear equations. For full rank matrices the minimum norm solution is returned, as well as an orthonormal basis for the null space of the matrix. If the system of equations is inconsistent only the least squares solution of minimal length is computed. *Portable Fortran subroutine in CMLIB (SUDSODS sublibrary).* **Classes:** D9a1, D9a4 **Usage:** CALL SUDS(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SUMMARY.** Generate summary statistics for a one-dimensional data set. Generated statistics include location statistics (mean, median, midmean, midrange), dispersion statistics (range, standard deviation, average absolute deviation, minimum and maximum, lower and upper quartiles, lower and upper hinges), randomness statistics (autocorrelation), and distributional statistics (8 types). *Command(s) in DATAPLOT interactive system.* **Classes:** L1a1 **Usage:** SUMMARY <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**SUMMARY.** Produces univariate descriptive statistics for numeric variables. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L1a1, L1c1 **Precision:** Single **Availability:** CAMVAX.

**SUMSL.** Minimizes a general unconstrained objective function using analytic gradient and a Hessian approximation from a secant update. *Portable Fortran subroutine in CMLIB (NL2SN sublibrary).* **Classes:** G1b1b **Usage:** CALL SUMSL(*12 parameters*) **Precision:** Single (Double: DSUMSL) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**SURF.** Compute a smooth bivariate interpolant to scattered data which is locally a quintic polynomial in two variables. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** E2b **Usage:** CALL SURF (*9 parameters*) **Precision:** Single (Double: DSURF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SURVIVAL.** Performs survival analysis. Options: subgroup comparison (using Lee and Desu algorithm calculated from statistic D), plots of the survival functions (cumulative distribution, probability density, hazard rate), missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L15 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**SVCAL.** Multiply a vector by a scalar and store the result in another vector, y = ax, all single precision. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a6 **Usage:** CALL SVCAL (N, SA, SX, INCX, SY, INCY) **Precision:** Single (Double: DVCAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVD.** Computes singular value decomposition of arbitrary real rectangular matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D6 **Usage:** CALL SVD(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**SVD.** Calculates the singular value decomposition (SVD) of a given rectangular matrix. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** D6 **Precision:** Single **Availability:** 855NOS.

**SVIBN.** Sort an integer array by absolute values. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N6a2a **Usage:** CALL SVIBN (N, IA, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVIBP.** Sort an integer array by absolute values and return a pointer array. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N6a1a, N6a2a **Usage:** CALL SVIBP (N, IA, IB, IPERM) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVIGN.** Sort an integer array by algebraic values. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N6a2a **Usage:** CALL SVIGN (N, IA, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVIGP.** Sort an integer array by algebraic values and return a pointer array. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N6a1a, N6a2a **Usage:** CALL SVIGP (N, IA, IB, IPERM) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVP.** Print vertical plot of Y versus input order with individual plot symbols specified by user; linear axis; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL SVP (Y, N, NS, ISYM) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SVPC.** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; user-supplied control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL SVPC (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SVPL.** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL SVPL (Y, N, NS, ISYM, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SVPM.** Print vertical plot of Y versus input order with individual plot symbols specified by user; linear axis; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL SVPM (Y, YMISS, N, NS, ISYM) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SVPMC.** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; user-supplied control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL SVPMC (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SVPML.** Print vertical plot of Y versus input order with individual plot symbols specified by user; log or linear horizontal (Y) axis; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL SVPML (Y, YMISS, N, NS, ISYM, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**SVRBN.** Sort a real array by absolute values. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N6a2b **Usage:** CALL SVRBN (N, RA, RB) **Precision:** Single (Double: DSVRBN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVRBP.** Sort a real array by absolute values and return a pointer array. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N6a1b, N6a2b **Usage:** CALL SVRBP (N, RA, RB, IPERM) **Precision:** Single (Double: DSVRBP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVRGN.** Sort a real array by algebraic values. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N6a2b **Usage:** CALL SVRGN (N, RA, RB) **Precision:** Single (Double: DSVRGN)

**Availability:** 855NOS, 855VE, 205, 840NOS.

**SVRGP.** Sort a real array by algebraic values and return a pointer array. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N6a1b, N6a2b **Usage:** CALL SVRGP (N, RA, RB, IPERM) **Precision:** Single (Double: DSVRGP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SVRGP.** Sort a real array by algebraic values and return a pointer array. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** N6a1b **Usage:** CALL SVRGP (N, RA, RB, IPERM) **Precision:** Single (Double: DSVRGP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SXYZ.** Compute a single precision XYZ product. *Proprietary Fortran function in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1a11 **Usage:** R = SXYZ(*7 parameters*) **Precision:** Single (Double: DXYZ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**SYMEIG.** Computes the eigenvalues and, optionally, the eigenvectors of a real symmertric matrix. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** D4a1 **Precision:** Single **Availability:** 855NOS.

**SYMMETRY PLOT.** Generate a symmetry plot. The horizontal axis contains the sorted $X_i$ for $i$ = N, N-1, ..., N/2 (or N/2+1) while the vertical axis contains the sorted $-X_i$ for $i$ = 1, 2, ..., N/2 (or N/2+1). Symmetric data sets should approximate a 45 degree line. *Command(s) in DATAPLOT interactive system.* **Classes:** L3a3 **Usage:** SYMMETRY PLOT <VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**SYMSTO.** To copy a symmetric matrix, stored in one of five different modes, to another symmetric matrix, thereby optionally changing the storage mode. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b8, D1b9 **Usage:** CALL SYMSTO(N, A, MA, ISA, B, MB, ISB) **Precision:** Single (Half: HSYMSTO) **Availability:** 205 (vectorized)

**SYMUUT.** To perform the Cholesky decomposition A = UL of the positive definite symmetric matrix A. U here denotes a non-singular upper triangular matrix, and L stands for its lower triangular transpose. Note that the order of the matrices U and L is reversed from that in the conventional LU decomposition. *Fortran/meta subroutine in MAGEV.* **Classes:** D2b1b **Usage:** CALL SYMUUT(N, A, KA, U, KU, D, NERR, NLP) **Precision:** Single (Half: HSYMUUT) **Availability:** 205 (vectorized)

# T

**3D-PLOT.** Generates a single or multi-surface 3-dimensional plot of data, functions, or both. *Command(s) in DATAPLOT interactive system.* **Classes:** L3c **Usage:** 3D-PLOT <RESPONSE VARIABLE> <INDEPENDENT VARIABLE 1> <INDEPENDENT VARIABLE 2> [<SURFACE VARIABLE>] **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**T PPCC PLOT.** Generates a probability plot correlation coefficient plot for the t distribution (plot of probability plot correlation coefficient vs. degrees of freedom parameter $\nu$ for $\nu$ ranging from 1 to 100 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3t **Usage:** T PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**T PROBABILITY PLOT.** Generates a probability plot for the Student's t distribution with degrees of freedom parameter $\nu$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2t **Usage:** T PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**T-TEST.** Performs a two sample t-test for the mean based on normal theory. The analysis is performed with and without equal variances assumption. The sample sizes do not have to be equal. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a4n, L4b1a4 **Usage:** T-TEST <VARIABLE NAME> <HYPOTHESIZED THEORETICAL MEAN VALUE> *or* T-TEST <VARIABLE NAME> <VARIABLE NAME> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**T-TEST.** Produces Student's t-statistic, degrees of freedom, and two-tailed probability for a comparison of two means for independent samples (using pooled- and separate -variance estimates with a test of homogeneity of variances) or for paired samples. Option: missing values. *Proprietary stand-alone program using SPSS command language.* **Classes:** L4b1a4 **Precision:** Single **Availability:** 855NOS, 855VE, 840NOS.

**TABLE.** Produces and prints one-way, two-way, and multi-way tables of counts with 20 optional subcommands for summarizing (e.g., cell mean, standard deviation), marginals, performing chi-square tests for each 2-way table, handling missing values, and selecting forms of input and output. *Command in MINITAB proprietary interactive system.* **Classes:** L1a1, L1c1, L9 **Usage:** TABLe the data classified by C, ..., [; subcommands MEANS for C, ..., C; MEDIANS ...; SUMS ...; MINIMUMS ...; MAXIMUMS ...; STDEV ...; STATS ...; DATA ...; NONMISSING ...; PROPORTION ...; COUNTS; ROWPERCENTS; COLPERCENTS; TOTPERCENTS; CHISQUARE ...; NOALL; ALL ...; MISSING ...; FREQUENCIES ...; LAYOUT ....] **Precision:** Single **Availability:** 855NOS.

**TABULATE.** Produce a table consisting of counts for each distinct value of a variable. Options include calculating either the counts, the means, the standard deviations, or the ranges of one variable for each distinct value of another variable. *Command(s) in DATAPLOT interactive system.* **Classes:** L2b **Usage:** TABULATE <VARIABLE> *or* TABULATE COUNT <VARIABLE> <IDENTIFIER VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**TABULATE.** Produces hierarchical tables of descriptive statistics from compositions of classification variables, analysis variables, and statistics keywords. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L1c1 **Precision:** Single **Availability:** CAMVAX.

**TAIL.** Performs a symmetric distribution tail length analysis on the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a6 **Usage:** CALL TAIL(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TAN.** tan(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4a **Usage:** R = TAN(X) **Precision:** Single (Double: DTAN) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**TAN.** tan(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = TAN(X) **Precision:** Single (Double: DTAN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TAN.** tan(x). *Proprietary Fortran function in PORT.* **Classes:** C4a **Usage:** R = TAN(X) **Precision:** Single (Double: DTAN) **Availability:** 855NOS, 205.

**TANH.** tanh(x). *Portable Fortran function in CMLIB (FNLIB sublibrary).* **Classes:** C4c **Usage:** R = TANH(X) **Precision:** Single (Double: DTANH) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**TANH.** tanh(x). *Proprietary Fortran function in IMSL SFUN/LIBRARY.* **Classes:** C4a **Usage:** R = TANH(X) **Precision:** Single (Double: DTANH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TANH.** tanh(x). *Proprietary Fortran function in PORT.* **Classes:** C4c **Usage:** R = TANH(X) **Precision:** Single (Double: DTANH) **Availability:** 855NOS, 205.

**TAPER.** Center a series about its mean and apply a split-cosine-bell taper; return the tapered series. (No printed output.) *Portable Fortran subroutine in STARPAC.* **Classes:** L10a1d **Usage:** CALL TAPER (Y, N, TAPERP, YT) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**TCDF.** Computes the cumulative distribution function value for Student's t distribution with degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L5a1t **Usage:** CALL TCDF(X, NU, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TCHBP.** Evaluates a polynomial expressed as a sum of Chebyshev polynomials. *Proprietary Fortran function in PORT.* **Classes:** C3a2 **Usage:** R = TCHBP(N, ALPHA, X, X0, X1) **Precision:** Single (Double: DTCHBP) **Availability:** 855NOS, 205.

**TCSCP.** Transform coefficients from a quadratic regression model generated from squares and crossproducts of centered variables to a model using uncentered variables. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L8b1b2 **Usage:** CALL TCSCP (NVAR, XMEAN, SCPM, BC, B) **Precision:** Single (Double: DTCSCP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TDATE.** Get today's date. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** R **Usage:** CALL TDATE (IDAY, MONTH, IYEAR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**TDF.** Evaluate the Student's t distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1t **Usage:** R = TDF(T, DF) **Precision:** Single (Double: DTDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TEMPLATE.** A device and computer-independent general-purpose, two and/or three dimensional Fortran-callable graphics system for static, interactive, and dynamic applications. Included are line drawing, text generation, coordinate system generation, windows, blanking regions, charting and graphing (e.g., contour and surface plots, business graphics), interactive input, graphics structures, display list operations, and color redefinition and selection. *TEMPLATE proprietary Fortran subprogram library.* **Classes:** Q **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX. *On 855NOS, 855VE, 205:* Output on Dicomed film recorder, Tektronix devices, QMS printers, the Zeta plotter, and various terminals.

**TETCC.** Categorize bivariate data and compute the tetrachoric correlation coefficient. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b4 **Usage:** CALL TETCC (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**TFPE.** Compute preliminary estimates of parameters for a univariate transfer function model. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L10b2b **Usage:** CALL TFPE (*17 parameters*) **Precision:** Single (Double: DTFPE) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TIME.** Performs a time series analysis on the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L10a3a1 **Usage:** CALL TIME(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TIMEPLOT.** Produces a line printer plot of one or more variables over time intervals. Options: missing values, user control of plot features. *Proprietary stand-alone program using SAS command language.* **Classes:** L3a6, L3e3c **Precision:** Single **Availability:** CAMVAX.

**TIN.** Evaluate the inverse of the Student's t distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a2t **Usage:** R = TIN(P, DF) **Precision:** Single (Double: DTIN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TINTERVAL.** Calculates a t-confidence interval with specified percent confidence. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1a4n, L4c1a **Usage:** TINTerval [with K percent confidence] for data in column C **Precision:** Single **Availability:** 855NOS.

**TINVIT.** Eigenvectors of symmetric tridiagonal matrix corresponding to some specified eigenvalues, using inverse iteration. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c3 **Usage:** CALL TINVIT(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**TNDF.** Evaluate the noncentral Student's t distribution function. *Proprietary Fortran function in IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** L5a1t **Usage:** R = TNDF(T, IDF, DELTA) **Precision:** Single (Double: DTNDF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TOL.** Computes normal and distribution-free tolerance limits for the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a7 **Usage:** CALL TOL(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TPLOT.** Prints pseudo three-dimensional plot of y versus x versus z, with symbols indicating the values of z, and with optional scale specification. *Command in MINITAB proprietary interactive system.* **Classes:** L3c **Usage:** TPLOt y in C [from K to K] vs x in C [from K to K] vs z in C **Precision:** Single **Availability:** 855NOS.

**TPLT.** Generates a Student's t probability plot with degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2t **Usage:** CALL TPLT(X, N, NU) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TPMOV.** To transpose a rectangular matrix, simultaneously moving it from one location to another. The source and target locations must not overlap. *Fortran/meta subroutine in MAGEV.* **Classes:** D1b3 **Usage:** CALL TPMOV(M, N, A, LA, B, LB) **Precision:** Single (Half: HTPMOV) **Availability:** 205 (vectorized)

**TPPF.** Computes the percent point function value for the Student's t distribution with degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L5a2t **Usage:** CALL TPPF(P, NU, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TQL1.** Compute eigenvalues of symmetric tridiagonal matrix by QL method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL TQL1(N, D, E, IERR) **Precision:** Single **Avail-**

**ability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TQL2.** Compute eigenvalues and eigenvectors of symmetric tridiagonal matrix. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL TQL2(NM, N, D, E, Z, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TQLRAT.** Computes eigenvalues of symmetric tridiagonal matrix using a rational variant of the QL method. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL TQLRAT(N, D, E2, IERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TRAN.** Generates a random sample of size N from the Student's t distribution with degrees of freedom parameter NU. *Fortran subroutine in DATAPAC.* **Classes:** L6a20 **Usage:** CALL TRAN(N, NU, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TRBAK1.** Forms the eigenvectors of real symmetric matrix from eigenvectors of symmetric tridiagonal matrix formed by TRED1. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL TRBAK1(NM, N, A, E, M, Z) **Also see:** TRED1 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TRBAK3.** Forms eigenvectors of real symmetric matrix from the eigenvectors of symmetric tridiagonal matrix formed by TRED3. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c4 **Usage:** CALL TRBAK3(NM, N, NV, A, M, Z) **Also see:** TRED3 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TRED1.** Reduce real symmetric matrix to symmetric tridiagonal matrix using orthogonal similarity transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b1 **Usage:** CALL TRED1(NM, N, A, D, E, E2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TRED2.** Reduce real symmetric matrix to symmetric tridiagonal matrix using and accumulating orthogonal transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b1 **Usage:** CALL TRED2(NM, N, A, D, E, Z) **Also see:** TQL2 IMTQL2 **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TRED2.** To reduce a symmetric matrix to a symmetric tridiagonal matrix. This is generally the first step in obtaining the complete eigensystem of the matrix. *Fortran/meta subroutine in MAGEV.* **Classes:** D4c1b1 **Usage:** CALL TRED2(N, A, MA, Z, MZ, D, S, T, ISM) **Precision:** Single (Half: HTRED2) **Availability:** 205 (vectorized)

**TRED3.** Reduce real symmetric matrix stored in packed form to symmetric tridiagonal matrix using orthogonal transformations. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4c1b1 **Usage:** CALL TRED3(N, NV, A, D, E, E2) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**TREE.** Prints a tree diagram from the output generated by SAS procedures CLUSTER or VARCLUS. Can also create an output data set identifying disjoint clusters at a specified level in the tree. Optional user control of plot features. *Proprietary stand-alone program using SAS command language.* **Classes:** L14c **Precision:** Single **Availability:** CAMVAX.

**TREEP.** Print a binary tree. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L14c **Usage:** CALL TREEP (*10 parameters*) **Precision:** Single (Double: DTREEP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TRIANGULAR PROBABILITY PLOT.** Generates a probability plot for the triangular distribution (with density $f(x) = 1 - |x|$ for $x$ in the interval [-1,1]). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2t **Usage:** TRIANGULAR PROBABILITY PLOT [<VARIABLE OF FREQUENCIES >] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**TRID.** To solve the linear system of equations AX = B where A is a tridiagonal square coefficient matrix supplied as real 64-bit (TRID) or real 32-bit (HTRID) numbers in a packed format. Special logic is included to significantly increase the efficiency when A is made up of many smaller, equally sized tridiagonal matrices. *Fortran/meta subroutine in MAGEV.* **Classes:** D2a2a **Usage:** CALL TRID(*9 parameters*) **Precision:** Single (Half: HTRID) **Availability:** 205 (vectorized)

**TRIDIB.** Computes eigenvalues of symmetric tridiagonal matrix in given interval using Sturm sequencing. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL TRIDIB(*14 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**TRIGP.** Evaluates a trigonometric polynomial. *Proprietary Fortran function in PORT.* **Classes:** C3a1 **Usage:** R = TRIGP(N, ALPHA, BETA, THETA) **Precision:** Single (Double: DTRIGP) **Availability:** 855NOS, 205.

**TRIM.** Computes the sample trimmed mean of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL TRIM(X, N, P1, P2, IWRITE, XTRIM) **Precision:** Single **Availability:** 855NOS, 840NOS.

**TRMESH.** Creates a Thiessen triangulation of n arbitrarily spaced points in the plane. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL TRMESH(N, X, Y, IADJ, IEND, IER) **Precision:** Single **Availability:**

855NOS.

**TRNRR.** Transpose a rectangular matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** D1b3 **Usage:** CALL TRNRR (*8 parameters*) **Precision:** Single (Double: DTRNRR) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TSPLOT.** Prints a scatter diagram of a time series, optionally using symbols modulo the period. Handles missing values. *Command in MINITAB proprietary interactive system.* **Classes:** L3a6 **Usage:** TSPLot [with period K [starting at K]] for data in C **Precision:** Single **Availability:** 855NOS.

**TSTURM.** Computes eigenvalues of symmetric tridiagonal matrix in given interval and eigenvectors by Sturm sequencing. *Portable Fortran subroutine in CMLIB (EISPACK sublibrary).* **Classes:** D4a5, D4c2a **Usage:** CALL TSTURM(*19 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**TTEST.** Performs one- or two-sided t-tests. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1a4n, L4c1a **Usage:** TTESt [of mu = K] on data in C, ..., C [; ALTERNATIVE = K.] **Precision:** Single **Availability:** 855NOS.

**TTEST.** Computes a t statistic for testing the hypothesis that the means of two groups of observations are equal. *Proprietary stand-alone program using SAS command language.* **Classes:** L4b1a4 **Precision:** Single **Availability:** CAMVAX.

**TWFRQ.** Tally observations into a two-way frequency table. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L2b **Usage:** CALL TWFRQ (*16 parameters*) **Precision:** Single (Double: DTWFRQ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TWO.** Computes overall mean and covariance matrices of a data set with appropriate labels for the cases and variables. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L1c1, L14d **Usage:** CALL TWO(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**TWODQ.** Automatic (adaptive) integration of a user specified function f(x, y) on one or more triangles to a prescribed relative or absolute accuracy. Two different quadrature formulas are available within TWODQ. This enables a user to integrate functions with boundary singularities. *Portable Fortran subroutine in CMLIB (TWODQ sublibrary).* **Classes:** H2b2a1 **Usage:** CALL TWODQ(*15 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**TWODQ.** Compute a two-dimensional iterated integral using internal calls to a one dimensional automatic integrator. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** H2b1a1 **Usage:** CALL TWODQ (*10 parameters*) **Precision:** Single (Double: DTWODQ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TWOMV.** Compute statistics for mean and variance inferences using samples from two normal populations. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L4b1a4 **Usage:** CALL TWOMV (*9 parameters*) **Precision:** Single (Double: DTWOMV) **Availability:** 855NOS, 855VE, 205, 840NOS.

**TWOSAMPLE.** Performs a one- or two-sided two-sample t-test of equality of two population means with variance estimated either from each sample or the pooled sample. *Command in MINITAB proprietary interactive system.* **Classes:** L4b1a4 **Usage:** TWOSample t [K percent confidence] for data in C and C [; subcommands ALTERNATIVE = K; POOLED.] **Precision:** Single **Availability:** 855NOS.

**TWOT.** Performs a one- or two-sided two-sample t-test of equality of two population means with variance estimated either from each sample or the pooled sample. *Command in MINITAB proprietary interactive system.* **Classes:** L4b1a4 **Usage:** TWOT [K percent confidence] for data in C, groups in C [; subcommands ALTERNATIVE = K; POOLED.] **Precision:** Single **Availability:** 855NOS.

**TWOWAYAOV.** Performs two-way analysis of variance for balanced data (equal number of observations, one or more, in each cell) and prints standard results. Options: fit additive model, save results. *Command in MINITAB proprietary interactive system.* **Classes:** L7b **Usage:** TWOWayaov for data in C, subscripts in C, C [store residuals in C [fits in C]] [; subcommand ADDITIVE.] **Precision:** Single **Availability:** 855NOS.

# U

**UAS.** Compute and print a univariate autoregressive spectrum analysis of a series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a5 **Usage:** CALL UAS (Y, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UASF.** Compute and print a univariate autoregressive spectrum analysis of a series; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a5 **Usage:** CALL UASF (YFFT, N, LYFFT, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UASFS.** Compute and optionally print a univariate autoregressive spectrum analysis of a series using user-supplied control values; use FFT for computations; return autoregressive and Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a5 **Usage:** CALL UASFS (*15 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UASS.** Compute and optionally print a univariate autoregressive spectrum analysis of a series using user-supplied control values; return autoregressive and Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a5 **Usage:** CALL UASS (*14 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UASV.** Compute and print a univariate autoregressive spectrum analysis of a series; input covariances rather than original series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a5 **Usage:** CALL UASV (ACOV, LAGMAX, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UASVS.** Compute and optionally print a univariate autoregressive spectrum analysis of a series using user-supplied control values; input covariances rather than original series; return autoregressive and Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a5 **Usage:** CALL UASVS (*15 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UERSET.** Set message level for IMSL routine UERTST. *Proprietary Fortran subroutine in IMSL.* **Classes:** R3a **Usage:** CALL UERSET (LEVEL, LEVOLD) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UERTST.** Print a message reflecting an error condition. *Proprietary Fortran subroutine in IMSL.* **Classes:** R3c **Usage:** CALL UERTST (IER, NAME) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**UFS.** Compute and print a univariate Fourier spectrum analysis of a series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFS (Y, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSF.** Compute and print a univariate Fourier spectrum analysis of a series; use FFT for computations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSF (YFFT, N, LYFFT, LDSTAK) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSFS.** Compute and optionally print a univariate Fourier spectrum analysis of a series using user-supplied control values; use FFT for computations; return Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSFS (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSM.** Compute and print a univariate Fourier spectrum analysis of a series with missing observations. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSM (Y, YMISS, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSMS.** Compute and optionally print a univariate Fourier spectrum analysis of a series with missing observations using user-supplied control values; return Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSMS (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSMV.** Compute and print a univariate Fourier spectrum analysis of a series with missing observations; input covariances rather than original series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSMV (ACOV, NLPPA, LAGMAX, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSMVS.** Compute and optionally print a univariate Fourier spectrum analysis of a series with missing observations using user-supplied control values; input covariances rather than original series; return Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSMVS (*14 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSS.** Compute and optionally print a univariate Fourier spectrum analysis of a series using user-supplied control values; return Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSS (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSV.** Compute and print a univariate Fourier spectrum analysis of a series; input covariances rather than original series. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSV (ACOV, LAGMAX, N) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UFSVS.** Compute and optionally print a univariate Fourier spectrum analysis of a series using user-supplied control values; input covariances rather than original series; return Fourier spectrum and corresponding frequencies. *Portable Fortran subroutine in STARPAC.* **Classes:** L10a3a4 **Usage:** CALL UFSVS (*13 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**UGETIO.** To retrieve current values and to set new values for input and output unit identifiers. *Proprietary Fortran subroutine in IMSL.* **Classes:** R1, R3b **Usage:** CALL UGETIO (IOPT, NIN, NOUT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UHELP.** Display methods of obtaining information on IMSL conventions regarding various subjects. Provide means for individual sites to supply users with site specific information. *Proprietary Fortran subroutine in IMSL.* **Classes:** R4 **Usage:** CALL UHELP **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UHELP1.** Write information regarding IMSL conventions and notation to an output file. *Proprietary Fortran subroutine in IMSL.* **Classes:** R4 **Usage:** CALL UHELP1 **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UHELP2.** Write information regarding IMSL input and output conventions. *Proprietary Fortran subroutine in IMSL.* **Classes:** R4 **Usage:** CALL UHELP2 **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UHELP3.** Write information regarding IMSL error detecting facilities. *Proprietary Fortran subroutine in IMSL.* **Classes:** R4 **Usage:** CALL UHELP3 **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UHELP4.** Write information regarding matrix/vector storage modes used in IMSL subroutines. *Proprietary Fortran subroutine in IMSL.* **Classes:** R4 **Usage:** CALL UHELP4 **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ULSIA.** Finds the minimal length solution of the underdetermined system of equations AX=B where A is an m-by-n matrix with $m \leq n$. Flexible version of SGLSS. *Portable Fortran subroutine in CMLIB (SGLSS sublibrary).* **Classes:** D9a1 **Usage:** CALL ULSIA(*20 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**UMACH.** Set or retrieve input or output device unit numbers. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY and IMSL SFUN/LIBRARY.* **Classes:** R1 **Usage:** CALL UMACH (N, NUNIT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMB.** Given interval endpoints, this generates a uniform mesh, with needed multiplicities for B-spline use. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL UMB (A, B, NAB, K, X, NX) **Precision:** Single (Double: DUMB) **Availability:** 855NOS, 205.

**UMCGF.** Minimize a function of N variables using a conjugate gradient algorithm and a finite-difference gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1a **Usage:** CALL UMCGF (*10 parameters*) **Precision:** Single (Double: DUMCGF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMCGG.** Minimize a function of N variables using a conjugate gradient algorithm and a user-supplied gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1b **Usage:** CALL UMCGG (*10 parameters*) **Precision:** Single (Double: DUMCGG) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMD.** Given interval endpoints, this generates a uniform mesh of distinct points. *Proprietary Fortran subroutine in PORT.* **Classes:** E3b, K6b **Usage:** CALL UMD (A, B, NAB, X) **Precision:** Single (Double: DUMD) **Availability:** 855NOS, 205.

**UMIAH.** Minimize a function of N variables using a modified Newton method and a user-supplied Hessian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1c **Usage:** CALL UMIAH (*11 parameters*) **Precision:** Single (Double: DUMIAH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMIDH.** Minimize a function of N variables using a modified Newton method and a finite-difference Hessian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1b **Usage:** CALL UMIDH (*10 parameters*) **Precision:** Single (Double: DUMIDH) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMINF.** Minimize a function of N variables using a quasi-Newton method and a finite-difference gradient. *Proprietary*

*Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1a **Usage:** CALL UMINF (*9 parameters*) **Precision:** Single (Double: DUMINF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMINF.** Minimize a function of N variables using a quasi-Newton method and a finite-difference gradient. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** G1a1a **Usage:** CALL UMINF (*9 parameters*) **Precision:** Single (Double: DUMINF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMING.** Minimize a function of N variables using a quasi-Newton method and a user-supplied gradient. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1b **Usage:** CALL UMING (*10 parameters*) **Precision:** Single (Double: DUMING) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UMKFL.** Decompose a non-zero floating-point number into a mantissa and an exponent. *Proprietary Fortran subroutine in PORT.* **Classes:** A6c **Usage:** CALL UMKFL ( F, E, M ) **Precision:** Single (Double: DUMKFL) **Availability:** 855NOS, 205.

**UMPOL.** Minimize a function of N variables using a direct search polytope algorithm. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b2 **Usage:** CALL UMPOL (*8 parameters*) **Precision:** Single (Double: DUMPOL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UNCMIN.** Finds minimum of N-variable function given initial values X0(N). Returns solution in X(N) and minimum function value in F. Needs EXTERNAL statement for user selected SUBROUTINE FCN(N,X,F) returning value of F at X(N). *Portable Fortran software in NMS library.* **Classes:** G1b1a **Usage:** CALL UNCMIN(N, X0, FCN, X, F, INFO, W, LW) **Precision:** Single (Double: UNCMND) **Availability:** PC.

**UNCMND.** *Double precision version of UNCMIN.*

**UNI.** Uniform random-number generator on (0,1). Uses Fibbonacci algorithm. Fast, excellent statistical properties and highly portable. *Portable Fortran function in CMLIB (RV sublibrary).* **Classes:** L6a21 **Usage:** R = UNI(JD) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**UNI.** Uniform random-number generator on (0,1). Uses Fibbonacci algorithm. Fast, excellent statistical properties and highly portable. *Portable Fortran software in NMS library.* **Classes:** L6a21 **Usage:** R = UNI() **Precision:** Single **Availability:** PC.

**UNI.** Returns a single real random variate from the uniform [0,1) distribution. *Proprietary Fortran function in PORT.* **Classes:** L6a21 **Usage:** R = UNI(K) **Also see:** RANBYT RANSET **Precision:** Single **Availability:** 855NOS, 205.

**UNIBAR.** Fits an autoregressive model by a Bayesian procedure. The least squares estimates of the parameters are obtained by the Householder transformation. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2c2 **Precision:** Single **Availability:** 855NOS.

**UNICDF.** Computes the cumulative distribution function value for the uniform (rectangular) distribution on the unit interval (0,1). *Fortran subroutine in DATAPAC.* **Classes:** L5a1u **Usage:** CALL UNICDF(X, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**UNIFORM PROBABILITY PLOT.** Generates a probability plot for uniform distribution (with density $f(x) = x$ for $x$ on the interval [0,1], mean 0.5 and standard deviation $\sqrt{1/12}$). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2u **Usage:** UNIFORM PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**UNIMAR.** A basic program for fitting of autoregressive models of successively higher orders by the method of least squares realized through Householder transformation. The outputs are the estimates of the coefficients, the innovation variances and the corresponding AIC statistics. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2c1 **Precision:** Single **Availability:** 855NOS.

**UNIPDF.** Computes the probability density function value for the uniform (rectangular) distribution on the unit interval (0,1). *Fortran subroutine in DATAPAC.* **Classes:** L5a1u **Usage:** CALL UNIPDF(X, PDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**UNIPLT.** Generates a uniform probability plot on the unit interval (0,1) with mean 0.5 and standard deviation $\sqrt{1/12}$. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2u **Usage:** CALL UNIPLT(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**UNIPPF.** Computes the percent point function value for the uniform (rectangular) distribution on the unit interval (0,1). *Fortran subroutine in DATAPAC.* **Classes:** L5a2u **Usage:** CALL UNIPPF(P, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**UNIRAN.** Generates a random sample of size N from the uniform (rectangular) distribution on the unit interval (0,1). *Fortran subroutine in DATAPAC.* **Classes:** L6a21 **Usage:** CALL UNIRAN(N, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**UNISF.** Computes the sparsity function value for the uniform (rectangular) distribution on the unit interval (0,1). *Fortran subroutine in DATAPAC.* **Classes:** L5a2u **Usage:** CALL UNISF(P, SF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**UNIVARIATE.** Produces simple descriptive statistics for numeric variables including extreme values and quantiles. Options: distribution plots, frequency table, missing values, weights, normality tests. *Proprietary stand-alone pro-*

*gram using SAS command language.* **Classes:** L1a1, L1c1, L3a2 **Precision:** Single **Availability:** CAMVAX.

**UNLSF.** Solve a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a finite-difference Jacobian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1a, K1b1a1 **Usage:** CALL UNLSF (*12 parameters*) **Precision:** Single (Double: DUNLSF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UNLSJ.** Solve a nonlinear least squares problem using a modified Levenberg-Marquardt algorithm and a user-supplied Jacobian. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1b1b, K1b1a2 **Usage:** CALL UNLSJ (*13 parameters*) **Precision:** Single (Double: DUNLSJ) **Availability:** 855NOS, 855VE, 205, 840NOS.

**URANDOM.** Generates K pseudo-random numbers from the uniform (0,1) distribution. *Command in MINITAB proprietary interactive system.* **Classes:** L6a21 **Usage:** URANdom K observations, put into C **Precision:** Single **Availability:** 855NOS.

**USBOX.** Print a boxplot (k samples). *Proprietary Fortran subroutine in IMSL.* **Classes:** L3a3, L3e4 **Usage:** CALL USBOX (X, K, NI, MAXL, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USCWV.** Print a complex vector. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USCWV(ITITLE, NC, A, M, INC, IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USHHST.** Print a horizontal histogram. *Proprietary Fortran subroutine in IMSL.* **Classes:** L3a1 **Usage:** CALL USHHST (T, N, IOPT, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USHST.** Print a vertical histogram. *Proprietary Fortran subroutine in IMSL.* **Classes:** L3a1 **Usage:** CALL USHST (T, N, ISP, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USHST2.** Print a vertical histogram, plotting two frequencies with one bar of the histogram. *Proprietary Fortran subroutine in IMSL.* **Classes:** L3b1 **Usage:** CALL USHST2 (T, U, N, ISP, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USLEAP.** Print results of the best-regressions analysis performed by IMSL routine RLEAP. *Proprietary Fortran subroutine in IMSL.* **Classes:** L8i **Usage:** CALL USLEAP (*9 parameters*) **Also see:** RLEAP **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USMNMX.** Determination of the minimum and maximum values of a vector. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a2, N5a **Usage:** CALL USMNMX (X, N, INC, XMIN, XMAX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USPC.** Print a sample pdf, a theoretical pdf and confidence band information; plot these on option. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a1 **Usage:** CALL USPC (PDF, X, N, N12, N95, IP, IC, W) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USPDF.** Plot of two sample probability distribution functions against their spectra. *Proprietary Fortran subroutine in IMSL.* **Classes:** L3b2 **Usage:** CALL USPDF (X, N, M, W, IW, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USPLO.** Printer plot of up to ten functions. *Proprietary Fortran subroutine in IMSL.* **Classes:** Q **Usage:** CALL USPLO (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USPRP.** Prints a probability plot of data against a normal, lognormal, half-normal, exponential, Weibull, or extreme value distribution. *Proprietary Fortran subroutine in IMSL.* **Classes:** L4a1a2e, L4a1a2h, L4a1a2l, L4a1a2n, L4a1a2w **Usage:** CALL USPRP (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USSLF.** Print a stem-and-leaf display. *Proprietary Fortran subroutine in IMSL.* **Classes:** L3a3 **Usage:** CALL USSLF (X, N, IUNIT, MAXL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USTREE.** Print a binary tree. *Proprietary Fortran subroutine in IMSL.* **Classes:** L14c, N1 **Usage:** CALL USTREE (*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWBM.** Print a matrix stored in band storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USEBM (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWBS.** Print a matrix stored in band symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USWBS (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWCH.** Print a complex matrix stored in Hermitian storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USWCH(ITITLE, NC, A, M, IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWCM.** Print a complex matrix stored in full storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USCWM(ITITLE, NC, A, IA, N, M, IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWFM.** Print a matrix stored in full storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USWFM (ITITLE, NC, A, IA, N, M, IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWFV.** Print a vector. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USWFV (ITITLE, NC, A, M, INC, IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**USWSM.** Print a matrix stored in symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** N1 **Usage:** CALL USWSM (ITITLE, NC, A, M, IOPT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**UVMGS.** Find the minimum point of a nonsmooth function of a single variable. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1a2 **Usage:** CALL UVMGS (F, A, B, TOL, XMIN) **Precision:** Single (Double: DUVMGS) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UVMID.** Find the minimum point of a smooth function of a single variable using both function evaluations and first derivative evaluations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1a1b **Usage:** CALL UVMID (*11 parameters*) **Precision:** Single (Double: DUVMID) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UVMIF.** Find the minimum point of a smooth function of a single variable using only function evaluations. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** G1a1a **Usage:** CALL UVMIF (*7 parameters*) **Precision:** Single (Double: DUVMIF) **Availability:** 855NOS, 855VE, 205, 840NOS.

**UVSTA.** Compute basic univariate statistics. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L1c1, L4c1a **Usage:** CALL UVSTA (*14 parameters*) **Precision:** Single (Double: DUVSTA) **Availability:** 855NOS, 855VE, 205, 840NOS.

# V

**VABMXF.** Maximum absolute value of the elements of a vector or a subset of the elements of a vector. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a2, D1a3c **Usage:** CALL VABMXF (V, L, INC, J, VMAX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VABMXS.** Maximum absolute value of the elements of a row or column of a matrix stored in symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a2, D1a3c **Usage:** CALL VABMXS (V, L, IRO, J, VMAX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VABSMF.** Sum of the absolute values of the elements of a vector or a subset of a vector. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a3a **Usage:** CALL VABSMF (V, L, INC, VSUM) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**VABSMS.** Sum of the absolute values of the elements of a row (or column) of a matrix stored in symmetric storage mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a3a **Usage:** CALL VABSMS (V, L, IRO, VSUM) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VAR.** Computes the sample variance (with denominator N-1) of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1b **Usage:** CALL VAR(X, N, IWRITE, XVAR) **Precision:** Single **Availability:** 855NOS, 840NOS.

**VARCLUS.** Performs either disjoint or hierarchical clustering of variables by maximizing the variation accounted for by either the first principal component or the centroid component of each cluster. Options: weights, missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L14a1a2, L14a1b **Precision:** Single **Availability:** CAMVAX.

**VARCO.** Sets up a tree of clusters of cases and uses the variance components algorithm to predict the value of the mean and variance for a predictor variable for each node in the tree, which can be used as approximations for a new case. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14a1a4 **Usage:** CALL VARCO(*18 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**VARCOMP.** Provides four methods (Type I, MIVQUEO, maximum-likelihood, and restricted maximum-likelihood) for estimating variance components in a general linear model containing random effects and optionally fixed effects. Option: missing values. *Proprietary stand-alone program using SAS command language.* **Classes:** L7d3 **Precision:** Single **Availability:** CAMVAX.

**VBESI0.** Modified Bessel function $I_0(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10b1 **Usage:** CALL VBESI0 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESI1.** Modified Bessel function $I_1(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10b1 **Usage:** CALL VBESI1 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESJ0.** Bessel function $J_0(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10a1 **Usage:** CALL VBESJ0 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESJ1.** Bessel function $J_1(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10a1 **Usage:** CALL VBESJ1 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESK0.** Modified Bessel function $K_0(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10b1 **Usage:** CALL VBESK0 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESK1.** Modified Bessel function $K_1(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10b1 **Usage:** CALL VBESK1 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESY0.** Bessel function $Y_0(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10a1 **Usage:** CALL VBESY0 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBESY1.** Bessel function $Y_1(x)$ for a vector of real arguments. *Fortran 200 subroutine in XMLIBV.* **Classes:** C10a1 **Usage:** CALL VBESY1 (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VBTOD.** Convert a machine-base mantissa and exponent into a base 10 floating-point number. *Proprietary Fortran subroutine in PORT.* **Classes:** A6b **Usage:** CALL VBTOD ( E, M, E10, M10 ) **Precision:** Single (Double: DVBTOD) **Availability:** 855NOS, 205.

**VCONC.** Compute the convolution of two complex vectors. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1a10 **Usage:** CALL VCONC (NX, X, NY, Y, NZ, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VCONR.** Compute the convolution of two real vectors. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** D1a10 **Usage:** CALL VCONR (NX, X, NY, Y, NZ, Z) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VCONVO.** Vector convolution. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a10, J2 **Usage:** CALL VCONVO (A, B, LA, LB, IWK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VCOSH.** cosh(x), for a real vector x. *Fortran 200 subroutine in XMLIBV.* **Classes:** C4c **Usage:** CALL VCOSH (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VCOSQB.** Backward fast Fourier cosine transform of multiple quarter-wave sequences. That is, cosine series representations with only odd wave numbers. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VCOSQB(M, N, X, XT, MDINX, WSAVE) **Also see:** VCOSQI VCOSQF **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. ***On 205:*** Vectorized version available.

**VCOSQF.** Forward fast Fourier cosine transform of multiple quarter-wave sequences. That is, cosine series representations with only odd wave numbers. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VCOSQF(M, N, X, XT, MDIMX, WSAVE) **Also see:** VCOSQI VCOSQB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. ***On 205:*** Vectorized version available.

**VCOST.** Discrete Fourier cosine transform of multiple even sequences. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VCOST(M, N, X, XT, MDIMX, WSAVE) **Also see:** VCOSTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. ***On 205:*** Vectorized version available.

**VCVTBF.** Storage mode conversion of matrices (band to full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTBF (A, N, NUC, NLC, IA, B, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTCH.** Storage mode conversion of matrices (full complex to Hermitian). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTCH (A, N, IA, H) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VCVTFB.** Storage mode conversion of matrices (full to band storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTFB (A, N, NUC, NLC, IA, B, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTFQ.** Storage mode conversion (full to band symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTFQ (A, N, NC, IA, AA, IAA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTFS.** Storage mode conversion of matrices (full to symmetric). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTFS (A, N, IA, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTHC.** Storage mode conversion of matrices (Hermitian to full complex). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTHC (H, N, B, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VCVTQF.** Storage mode conversion (band symmetric to full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTQF (A, N, NC, IA, B, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTQS.** Storage mode conversion (band symmetric to symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTQS (A, N, NC, IA, B) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTSF.** Storage mode conversion of matrices (symmetric to full). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTSF (A, N, B, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VCVTSQ.** Storage mode conversion (symmetric to band symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b9 **Usage:** CALL VCVTSQ (A, N, NC, B, IB) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. ***On 205:*** Vectorized version available.

**VDCPS.** Decompose an integer into its prime factors. *Proprietary Fortran subroutine in IMSL.* **Classes:** B **Usage:** CALL VDCPS (N, NPF, IPF, IEXP, IPWR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VDTOB.** Convert a base-10 mantissa and exponent into an machine-base floating-point number. *Proprietary Fortran subroutine in PORT.* **Classes:** A6b **Usage:** CALL VDTOB ( E10, M10, E, M ) **Precision:** Single (Double: DVDTOB) **Availability:** 855NOS, 205.

**VG.** A collection of Fortran callable subroutines for two dimensional graphics and data plotting. Its main features are ease of use, portability, and interactive capabilities, such as automatic zooming, adding and moving text on the screen, etc. Output to a laser printer or pen plotter. *VOLKSGRAPHER Fortran subprogram library.* **Classes:** Q **Precision:** Single **Availability:** CAMVAX, PC.

**VHS12.** Real Householder transformation computation and applications. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a9, D1b11 **Usage:** CALL VHS12 (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VHS2P.** Print a vertical histogram with every bar subdivided into two parts. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3b1 **Usage:** CALL VHS2P (NBAR, FRQX, FRQY, ISP, TITLE) **Precision:** Single (Double: DVHS2P) **Availability:** 855NOS, 855VE, 205, 840NOS.

**VHSH2C.** Complex Householder transformation to zero a single element of a matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b11 **Usage:** CALL VHSH2C (*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VHSH2R.** Real Householder transformation to zero a single element of a matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b11 **Usage:** CALL VHSH2R (AJ, AJP1, UJ, UJP1, VJ, VJP1) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VHSH3R.** Real Householder transformation to zero two elements of a matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b11 **Usage:** CALL VHSH3R (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VHSTP.** Print a vertical histogram. *Proprietary Fortran subroutine in IMSL STAT/LIBRARY.* **Classes:** L3a1 **Usage:** CALL VHSTP (NBAR, FRQ, ISP, TITLE) **Precision:** Single (Double: DVHSTP) **Availability:** 855NOS, 855VE, 205, 840NOS.

**VIPRFF.** Vector inner product of two vectors or subsets of two vectors. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a4 **Usage:** CALL VIPRFF (X, Y, L, IX, IY, XYIP) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VIPRSS.** Vector inner product of two vectors each of which is part of some matrix stored in symmetric mode. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1a4 **Usage:** CALL VIPRSS (X, Y, L, IX, IY, XYIP) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULBB.** Matrix multiplication (band storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULBB (A, IA, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULBF.** Matrix multiplication (band by full matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULBF (A, IA, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULBS.** Matrix multiplication (band by symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULBS (A, IA, B, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULFB.** Matrix multiplication (full by band matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULFB (A, IA, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULFF.** Matrix multiplication (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULFF (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULFM.** Matrix multiplication of the transpose of matrix A by matrix B (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULFM (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULFP.** Matrix multiplication of matrix A by the transpose of matrix B (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULFP (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULFQ.** Matrix multiplication (full by band symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULFQ (A, M, N, IA, B, NC, IB, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULFS.** Matrix multiplication (full by symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULFS (A, B, L, M, IA, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULQB.** Matrix multiplication (band symmetric by band matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULQB (A, IA, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULQF.** Matrix multiplication (band symmetric by full matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULQF (A, M, NC, IA, B, N, IB, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULQQ.** Matrix multiplication (band symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULQQ (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULQS.** Matrix multiplication (band symmetric by symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULQS (A, N, NC, IA, B, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULSB.** Matrix multiplication (symmetric by band matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULSB (A, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULSF.** Matrix multiplication (symmetric by full matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULSF (A, N, B, M, IB, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULSQ.** Matrix multiplication (symmetric by band symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULSQ (A, N, B, NC, IB, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VMULSS.** Matrix multiplication (symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VMULSS (A, B, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VNRMF1.** 1-norm of matrices (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b2 **Usage:** CALL VNRMF1 (A, N, IA, XNRMA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**VNRMF2.** Euclidean-norm of matrices (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b2 **Usage:** CALL VNRMF2 (A, N, IA, XNRMA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VNRMFI.** Infinity-norm matrices (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b2 **Usage:** CALL VNRMFI (A, N, IA, XNRMA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VNRMS1.** 1-norm of matrices (symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b2 **Usage:** CALL VNRMS1 (A, N, XNRMA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VNRMS2.** Euclidean-norm of matrices (symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b2 **Usage:** CALL VNRMS2 (A, N, XNRMA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VNULL.** A program for detecting rank degeneracy. It can select a set of independent columns of a matrix when numerical rank can be determined. *ROSEPAK is a portable interactive system.* **Classes:** L8c1a1 **Precision:** Single **Availability:** 855NOS.

**VOR.** Constructs a 2-d Voronoi diagram for a set of sites in a simply connected bounded planar polygonal region. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL VOR(*14 parameters*) **Also see:** VPREP **Precision:** Single **Availability:** 855NOS.

**VORU.** Constructs a 2-d Voronoi diagram for a set of sites in the Euclidean plane. *Portable Fortran subroutine in CGLIB.* **Classes:** P **Usage:** CALL VORU(*10 parameters*) **Also see:** VPREP **Precision:** Single **Availability:** 855NOS.

**VP.** Print vertical plot of Y versus input order; linear axes; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL VP (Y, N, NS) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**VPC.** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; user-supplied control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL VPC (*11 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**VPL.** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; default control values and axis limits; no missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL VPL (Y, N, NS, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**VPM.** Print vertical plot of Y versus input order; linear axis; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL VPM (Y, YMISS, N, NS) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**VPMC.** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; user-supplied control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL VPMC (*12 parameters*) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**VPML.** Print vertical plot of Y versus input order; log or linear horizontal (Y) axis; default control values and axis limits; missing values allowed. *Portable Fortran subroutine in STARPAC.* **Classes:** L3a6 **Usage:** CALL VPML (Y, YMISS, N, NS, ILOG) **Availability:** 855NOS (single precision), 855VE (single precision), 205 (single precision), CAMVAX (double precision), 840NOS (single precision).

**VPOLYF.** Matrix polynomial (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b7 **Usage:** CALL VPOLYF (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VRFFTB.** Synthesis (backward transform) of multiple real periodic sequences from their Fourier coefficients. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a1 **Usage:** CALL VRFFTB(M, N, R, RT, MDIMR, WSAVE) **Also see:** VRFFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VRFFTF.** Analysis (forward transform) of multiple real periodic sequences into Fourier coefficients. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a1 **Usage:** CALL VRFFTF(M, N, R, RT, MDIMR, WSAVE) **Also see:** VRFFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VRFFTI.** Initializes WSAVE array for VRFFTF and VRFFTB. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Usage:** CALL VRFFTI(N, WSAVE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSAR.** Sorting of matrices (with options). *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a1b, N6a2b **Usage:** CALL VSAR(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSCOSB.** Synthesis (backward transform) of multiple Fourier cosine coefficients into real staggered grid sequences.

*Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSCOSB(F, L, M, N, FT, C1, C2, WSAVE) **Also see:** VSCOSI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSCOSF.** Analysis (forward transform) of multiple real staggered grid sequences into Fourier cosine coefficients. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSCOSF(F, L, M, N, FT, C1, C2, WSAVE) **Also see:** VSCOSI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSCOSI.** Initialization routine for VSCOSF and VSCOSB. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Usage:** CALL VSCOSI(N, C1, C2, WSAVE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSCOSQ.** Analysis or synthesis (forward or backward transform) of multiple real staggered grid sequences to or from Fourier cosine quarter-wave sequences. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSCOSQ(*10 parameters*) **Also see:** VSCSQI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSCSQI.** Initialization routine for VSCOSQ. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Usage:** CALL VSCSQI(N, C1, C2, C3, C4, WSAVE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSINH.** sinh(x), for a real vector x. *Fortran 200 subroutine in XMLIBV.* **Classes:** C4c **Usage:** CALL VSINH (M, X, F) **Precision:** Single **Availability:** 205 (vectorized)

**VSINQB.** Backward fast Fourier sine transform of multiple quarter-wave sequences. That is, sine series representations with only odd wave numbers. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSINQB(M, N, X, XT, MDIMX, WSAVE) **Also see:** VSINQI VSINQF **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSINQF.** Forward fast Fourier sine transform of multiple quarter-wave sequences. That is, sine series representations with only odd wave numbers. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSINQF(M, N, X, XT, MDIMX, WSAVE) **Also see:** VSINQI VSINQB **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSINT.** Discrete Fourier sine transform of multiple odd sequences. *Portable Fortran subroutine in CMLIB (VFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSINT(M, N, X, XT, MDIMX, WSAVE) **Also see:** VSINTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSODA.** Sorting of columns of a double precision matrix in ascending order of keys in rows. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a2b **Usage:** CALL VSODA(A, IA, NR, NC, NK, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSORA.** Sorting of columns of a real matrix into ascending order of keys in rows. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a2b **Usage:** CALL VSORA(A, IA, NR, NC, NK, WK, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSRFTB.** Synthesis (backward transform) of multiple Fourier coefficients into real staggered grid sequences. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a1 **Usage:** CALL VSRFTB(F, L, M, N, FT, WSAVE) **Also see:** VSRFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSRFTF.** Analysis (forward transform) of multiple real staggered grid sequences into Fourier coefficients. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a1 **Usage:** CALL VSRFTF(F, L, M, N, FT, WSAVE) **Also see:** VSRFTI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSRFTI.** Initialization routine for VSRFTF and VSRFTB. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Usage:** CALL VSRFTI(N, WSAVE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSRTA.** Sorting of arrays by algebraic value. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a2b **Usage:** CALL VSRTA (A, LA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSRTM.** Sorting of arrays by absolute value. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a2b **Usage:** CALL VSRTM (A, LA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSRTP.** Sorting of arrays by absolute value, pointer array returned. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a2b **Usage:** CALL VSRTP (A, LA, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSRTR.** Sorting of arrays by algebraic value, pointer array returned. *Proprietary Fortran subroutine in IMSL.* **Classes:** N6a2b **Usage:** CALL VSRTR (A, LA, IR) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSRTU.** Interchange the rows or columns of a matrix using a permutation vector such as the one obtained from IMSL routines VSRTP or VSRTR. *Proprietary Fortran subroutine in IMSL.* **Classes:** N8 **Usage:** CALL VSRTU (Z, IZ, N, M, IND, IR, WK) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VSSINB.** Synthesis (backward transform) of multiple Fourier sine coefficients into real staggered grid sequences. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSSINB(F, L, M, N, FT, C1, C2, WSAVE) **Also see:** VSSINI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSSINF.** Analysis (forward transform) of multiple real staggered grid sequences into Fourier sine coefficients. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSSINF(F, L, M, N, FT, C1, C2, WSAVE) **Also see:** VSSINI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSSINI.** Initialization routine for VSSINF and VSSINB. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Usage:** CALL VSSINI(N, C1, C2, WSAVE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSSINQ.** Analysis or synthesis (forward or backward transform) of multiple real staggered grid sequences to or from Fourier sine quarter-wave coefficients. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Classes:** J1a3 **Usage:** CALL VSSINQ(*10 parameters*) **Also see:** VSSNQI **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VSSNQI.** Initialization routine for VSSINQ. *Portable Fortran subroutine in CMLIB (VSFFT sublibrary).* **Usage:** CALL VSSNQI(N, C1, C2, C3, C4, WSAVE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS. *On 205:* Vectorized version available.

**VTPROF.** Transpose product of matrix (full storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VTPROF (A, L, M, IA, ATA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VTPROS.** Transpose product of a matrix (symmetric storage mode). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b6 **Usage:** CALL VTPROS (A, N, ATA) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VTRAN.** Transpose a rectangular matrix. *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b3 **Usage:** CALL VTRAN (A, N, M) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VUABQ.** Matrix addition (band + band symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b5 **Usage:** CALL VUABQ (A, IA, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VUAFB.** Matrix addition (full + band matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b5 **Usage:** CALL VUAFB (A, IA, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VUAFQ.** Matrix addition (full + band symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b5 **Usage:** CALL VUAFQ (A, N, IA, B, NC, IB, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VUAFS.** Matrix addition (full + symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b5 **Usage:** CALL VUAFS (A, N, IA, B, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VUASB.** Matrix addition (symmetric + band matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b5 **Usage:** CALL VUASB (A, B, IB, N, C, IC) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**VUASQ.** Matrix addition (symmetric + band symmetric matrices). *Proprietary Fortran subroutine in IMSL.* **Classes:** D1b5 **Usage:** CALL VUASQ (A, N, B, NC, IB, C) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

# W

**WALSH.** Calculates $(X_i + X_j)/2$ and stores these averages and their indices. (Useful for nonparametric tests and confidence intervals.) *Command in MINITAB proprietary interactive system.* **Classes:** L2a **Usage:** WALSh averages of values in C, put into C [indices into C and C] **Precision:** Single **Availability:** 855NOS.

**WCOV.** Computes the within-cluster covariance matrix. *Portable Fortran subroutine in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** CALL WCOV(*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**WDIST.** Computes a weighted Euclidean distance between two cases. *Portable Fortran function in CMLIB (CLUSTER sublibrary).* **Classes:** L14d **Usage:** SDIST = WDIST(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**WEIB.** Performs a Weibull distribution analysis on the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a4w **Usage:** CALL WEIB(X, N) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WEIBULL PLOT.** Generates a Weibull plot to assess the goodness of fit of a data set to a two parameter Weibull distribution. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2w **Usage:** WEIBULL PLOT <VARIABLE>, WEIBULL PLOT <VARIABLE> <TAG VARIABLE> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**WEIBULL PPCC PLOT.** Generates a probability plot correlation coefficient plot for the Weibull distribution (plot of probability plot correlation coefficient vs. the tail parameter $\gamma$ for $\gamma$ ranging from 1 to 100 or in user-set range). *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a3w **Usage:** WEIBULL PPCC PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**WEIBULL PROBABILITY PLOT.** Generates a Weibull probability plot with tail length parameter $\gamma$. *Command(s) in DATAPLOT interactive system.* **Classes:** L4a1a2w **Usage:** WEIBULL PROBABILITY PLOT [<VARIABLE OF FREQUENCIES>] <RESPONSE VARIABLE> **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX.

**WEICDF.** Computes the cumulative distribution function value for the Weibull distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a1w **Usage:** CALL WEICDF(X, GAMMA, CDF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WEIPLT.** Generates a Weibull probability plot with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L4a1a2w **Usage:** CALL WEIPLT(X, N, GAMMA) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WEIPPF.** Computes the percent point function value for the Weibull distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L5a2w **Usage:** CALL WEIPPF(P, GAMMA, PPF) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WEIRAN.** Generates a random sample of size N from the Weibull distribution with tail length parameter GAMMA. *Fortran subroutine in DATAPAC.* **Classes:** L6a23 **Usage:** CALL WEIRAN(N, GAMMA, ISTART, X) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WIND.** Computes the sample Windsorized mean of the data in the input vector X. *Fortran subroutine in DATAPAC.* **Classes:** L1a1a **Usage:** CALL WIND(X, N, P, P1, P2, IWRITE, XWIND) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WINTERVAL.** Calculates a one-sample Wilcoxon rank estimate and confidence interval for the center of a symmetric distribution. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1b1, L4c1b **Usage:** WINTerval [percent confidence K] for data in C, ..., C **Precision:** Single **Availability:** 855NOS.

**WNNLS.** Solves linearly constrained non-negative least squares problem. *Portable Fortran subroutine in CMLIB (FC sublibrary).* **Classes:** D9b1 **Usage:** CALL WNNLS(*12 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**WRCRL.** Print a complex rectangular matrix with a given format and labels. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N1 **Usage:** CALL WRCRL (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**WRCRN.** Print a complex rectangular matrix with integer row and column labels. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** N1 **Usage:** CALL WRCRN (*6 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**WRIRL.** Print an integer rectangular matrix with a given format and labels. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N1 **Usage:** CALL WRIRL (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**WRIRN.** Print an integer rectangular matrix with integer row and column labels. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N1 **Usage:** CALL WRIRN (*6 parameters*)

**Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**WRITE.** Writes out the contents of the vector X. *Fortran subroutine in DATAPAC.* **Classes:** N1 **Usage:** CALL WRITE(X, N, NNLINE, IWIDTH, IDEC) **Precision:** Single **Availability:** 855NOS, 840NOS.

**WROPT.** Set or retrieve an option for printing a matrix. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N1 **Usage:** CALL WROPT (IOPT, ISET, ISCOPE) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**WRRRL.** Print a real rectangular matrix with a given format and labels. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N1 **Usage:** CALL WRRRL (*9 parameters*) **Precision:** Single (Double: DWRRRL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**WRRRN.** Print a real rectangular matrix with integer row and column labels. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY and IMSL STAT/LIBRARY.* **Classes:** N1 **Usage:** CALL WRRRN (*6 parameters*) **Precision:** Single (Double: DWRRRN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**WTEST.** Performs one-sample one- or two-sided Wilcoxon signed-rank tests. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1b1, L4c1b **Usage:** WTESt [of center = K] on data in C, ..., C [; subcommand ALTERNATIVE = K.] **Precision:** Single **Availability:** 855NOS.

# X

**X01AAF.** Returns the value of the constant pi. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X01AAF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X01ABF.** Returns the value of Euler's constant, lambda. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X01ABF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AAF.** Returns the value $\epsilon$, where $\epsilon$ is the smallest positive real machine number such that $1.0+\epsilon > 1.0$. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AAF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02ABF.** Returns the value of RMIN, the smallest positive real floating-point number such that RMIN and -RMIN are exactly representable on the computer. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02ABF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02ACF.** Returns the value of RMAX where RMAX is the largest computable real floating-point number such that RMAX and -RMAX are exactly representable on the computer. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02ACF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02ADF.** Returns the value of TOL, the ratio RMIN/EPS (see X02ABF and X02AAF). *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02ADF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AEF.** Returns the largest negative real value ENEG such that exp(ENEG) can be successfully evaluated without underflow by the compiler-supplied EXP routine. (DEXP for double precision implementations.) *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AEF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AFF.** Returns the value of the largest positive permitted argument for EXP (or DEXP). *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AFF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AGF.** Returns the value of the smallest positive representable floating-point number R such that -R, 1.0/R and -1.0/R are all computable without underflow or overflow. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02AGF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AHF.** Returns the largest positive REAL argument for which the SIN and COS routines return a result with some meaningful accuracy. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AHF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AJF.** Returns the machine precision, i.e., $(1/2)B^{1-P}$ if ROUNDS is true or $B^{1-P}$ otherwise. (See the Chapter Introduction in the NAG Fortran Library Manual.) *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AJF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AKF.** Returns the smallest positive model number, i.e., $B^{EMIN-1}$. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AKF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02ALF.** Returns the largest positive model number, i.e., $(1\text{-}B^{-P})B^{EMAX}$. (See the Chapter Introduction in the NAG Fortran Library Manual.) *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02ALF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02AMF.** Returns the safe range parameter as defined in the Chapter Introduction in the NAG Fortran Library Manual. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** R = X02AMF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BAF.** Returns the value of BASE, the base of the arithmetic used on the computer. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BAF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BBF.** Returns the largest positive integer value. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BBF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BCF.** Returns the value of MAXPW2, the largest integer power to which 2.0 may be raised without overflow. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BCF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BDF.** Returns the value of MINPW2, the largest negative integer power to which 2.0 may be raised without underflow. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BDF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BEF.** Maximum number of decimal digits that can be represented. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BEF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BHF.** Returns the model parameter B (see the Chapter Introduction in the NAG Fortran Library Manual). *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BHF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BJF.** Returns the model parameter P (see the Chapter Introduction in the NAG Fortran Library Manual).

*Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BJF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BKF.** Returns the model parameter EMIN (see the Chapter Introduction in the NAG Fortran Library Manual). *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BKF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02BLF.** Returns the model parameter EMAX (see the Chapter Introduction in the NAG Fortran Library Manual). *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02BLF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02CAF.** Returns an estimate of the active set size in a paged environment; in a non-paged environment, it returns zero. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** I = X02CAF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02DAF.** Switch for taking precautions to avoid underflow. *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** L = X02DAF(X) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X02DJF.** Returns the model parameter ROUNDS (see the Chapter Introduction in the NAG Fortran Library Manual). *Proprietary Fortran function in NAG.* **Classes:** R1 **Usage:** L = X02DJF() **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X03AAF.** Calculates the value of a scalar product using basic or additional precision and adds it to a basic or additional precision initial value. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a4 **Usage:** CALL X03AAF (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X03ABF.** Calculates the value of a complex scalar product using basic or additional precision and adds it to a complex initial value. *Proprietary Fortran subroutine in NAG.* **Classes:** D1a4 **Usage:** CALL X03ABF (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X04AAF.** Returns the value of the current error message unit number, or sets the current error message unit number to a new value. *Proprietary Fortran subroutine in NAG.* **Classes:** R3b **Usage:** CALL X04AAF (IFLAG, NERR) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X04ABF.** Returns the value of the current advisory message unit number, or sets the current advisory message unit number to a new value. *Proprietary Fortran subroutine in NAG.* **Classes:** R3b **Usage:** CALL X04ABF (IFLAG, NADV) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X04BAF.** Writes a single formatted record to an external file. *Proprietary Fortran subroutine in NAG.* **Classes:** N1 **Usage:** CALL X04BAF (NOUT, REC) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**X04BBF.** Reads a single formatted record from an external file. *Proprietary Fortran subroutine in NAG.* **Classes:** N1 **Usage:** CALL X04BBF (NIN, REC, IFAIL) **Precision:** Single **Availability:** 855NOS, 855VE, 205.

**XDLEGF.** *Double precision version of XSLEGF.*

**XDNRMP.** *Double precision version of XSNRMP.*

**XERABT.** Terminate run and print traceback. (Requires system dependent programming to execute properly, else just STOPs.) *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERABT(MESSG, NMESSG) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XERCLR.** Clear current message number. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERCLR **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XERCTL.** Perform special error processing of one message. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERCTL(*5 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XERDMP.** Print error summary and clear tables. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERDMP **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XERMAX.** Set limit of MAX times each message can be printed. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERMAX(MAX) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XERROR.** Process an error message. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERROR(MESSG, NMESSG, NERR, LEVEL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XERRWV.** Process a message with numeric values. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XERRWV(*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XGETF.** Get current value of KONTRL. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XGETF(KONTRL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XGETUA.** Get current output unit numbers. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3c **Usage:** CALL XGETUA(IUNITA, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XGETUN.** Get current output unit number. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:**

R3c **Usage:** CALL XGETUN(IUNIT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XSARMA.** Produces exact maximum likelihood estimates of the parameters of a scalar ARMA model. *Portable stand-alone program using TIMSAC command language.* **Classes:** L10a2d2 **Precision:** Single **Availability:** 855NOS.

**XSETF.** Set KONTRL for XERROR, default is 2. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3a **Usage:** CALL XSETF(KONTRL) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XSETUA.** Set up to 5 output unit numbers. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3b **Usage:** CALL XSETUA(IUNITA, N) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XSETUN.** Set one output unit number. *Portable Fortran subroutine in CMLIB (XERROR sublibrary).* **Classes:** R3b **Usage:** CALL XSETUN(IUNIT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XSLEGF.** Calculates a sequence of values of Legendre functions of the first or second kind of positive or negative order. The argument and either degree or order are fixed. *Portable Fortran subroutine in CMLIB (FCNPAK sublibrary).* **Classes:** C9, C3a2 **Usage:** CALL XSLEGF(*8 parameters*) **Also see:** XSSET **Precision:** Single (Double: XDLEGF) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**XSNRMP.** Calculates a sequence of values of normalized Legendre polynomials for fixed degree and argument and variable order. *Portable Fortran subroutine in CMLIB (FCNPAK sublibrary).* **Classes:** C9, C3a2 **Usage:** CALL XSNRMP(*8 parameters*) **Also see:** XSSET **Precision:** Single (Double: XDNRMP) **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

# Y

**YATES ANALYSIS.** Performs a Yates analysis of an experimental design. It estimates the factor effects for $2^k$ and $2^{k-p}$ factorial designs. The data must be entered in standard form. *Command(s) in DATAPLOT interactive system.* **Classes:** L7d1 **Usage:** YATES ANALYSIS <VARIABLE>. **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

**YOUDEN PLOT.** Generates a Youden plot (to carry out an interlab comparison). The analysis can be either where each lab has made two runs on the same product or one run on two different products. *Command(s) in DATAPLOT interactive system.* **Classes:** L3e3c **Usage:** YOUDEN PLOT <RUN 1 VARIABLE> <RUN 2 VARIABLE> <LAB ID> **Precision:** Single **Availability:** 855VE, 205, CAMVAX.

# Z

**ZANLY.** Finds a zero of a univariate analytic function. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F1b **Usage:** CALL ZANLY (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZANLYT.** Finds zeros of an analytic complex function using the Muller method with deflation. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1b **Usage:** CALL ZANLYT (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZBREN.** Finds a zero of a function which changes sign in a given interval. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F1b **Usage:** CALL ZBREN (*6 parameters*) **Precision:** Single (Double: DZBREN) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZBRENT.** Zero of a function which changes sign in a given interval (Brent algorithm). *Proprietary Fortran subroutine in IMSL.* **Classes:** F1b **Usage:** CALL ZBRENT (F, EPS, NSIG, A, B, MAXFN, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZCPOLY.** Finds zeros of a polynomial with complex coefficients using the Jenkins-Traub method. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1a2 **Usage:** CALL ZCPOLY (A, NDEG, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZERO.** Finds a single real root of a function within an interval specified by the user. *Proprietary Fortran function in PORT.* **Classes:** F1b **Usage:** R = ZERO(F, A, B, T) **Precision:** Single (Double: DZERO) **Availability:** 855NOS, 205.

**ZEROIN.** Finds a zero of a user defined function on a given interval in which the function changes sign. *Portable Fortran subroutine in CMLIB (ZEROIN sublibrary).* **Classes:** F1b **Usage:** CALL ZEROIN(F, B, C, RE, AE, IFLAG) **Precision:** Single **Availability:** 855NOS, 855VE, 205, CAMVAX, 840NOS.

**ZEROIN.** Finds a root of a nonlinear user-supplied function given an initial interval in which the function changes sign. In BASIC. *Portable Basic software in SCRUNCH library.* **Classes:** F1b **Precision:** Single **Availability:** 855NOS.

**ZETA.** A set of Fortran-callable subroutines which provide access to the Zeta plotter, an off-line, 4 color, pen plotter. The library contains the standard Calcomp library routines with modest extensions. *ZETA proprietary Fortran subprogram library.* **Classes:** Q **Precision:** Single **Availability:** 855NOS, 855VE.

**ZFALSE.** Finds a zero of a function given an interval containing the zero. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1b **Usage:** CALL ZFALSE (*8 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZINTERVAL.** Calculates a z-confidence interval with specified percent confidence and standard deviation. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1a4n **Usage:** ZINTerval K percent confidence, assuming sigma = K, on C **Precision:** Single **Availability:** 855NOS.

**ZONE.** Finds a solution of a system of non-linear equations. *Proprietary Fortran subroutine in PORT.* **Classes:** F2 **Usage:** CALL ZONE (*6 parameters*) **Precision:** Single (Double: DZONE) **Availability:** 855NOS, 205.

**ZONEJ.** Finds a solution of a system of non-linear equations. User must provide a SUBROUTINE to compute the Jacobian matrix. *Proprietary Fortran subroutine in PORT.* **Classes:** F2 **Usage:** CALL ZONEJ (*7 parameters*) **Precision:** Single (Double: DZONEJ) **Availability:** 855NOS, 205.

**ZPLRC.** Find the zeros of a polynomial with real coefficients using Laguerre's method. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F1a1 **Usage:** CALL ZPLRC (NDEG, COEFF, ROOT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZPOCC.** Finds the zeros of a polynomial with complex coefficients. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F1a2 **Usage:** CALL ZPOCC (NDEG, COEFF, ROOT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZPOLR.** Finds zeros of a polynomial with real coefficients using Laguerre's method. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1a1 **Usage:** CALL ZPOLR (A, NDEG, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZPORC.** Find the zeros of a polynomial with real coefficients using the Jenkins-Traub three-stage algorithm. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F1a1 **Usage:** CALL ZPORC (NDEG, COEFF, ROOT) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZQADC.** Finds zeros of a quadratic with complex coefficients. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1a2 **Usage:** CALL ZQADC (A, B, C, ZSM, ZLG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZQADR.** Finds zeros of a quadratic with real coefficients. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1a1 **Usage:** CALL ZQADR (A, B, C, ZSM, ZLG, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZREAL.** Finds a real zero of a real function. *Proprietary Fortran subroutine in IMSL MATH/LIBRARY.* **Classes:** F1b **Usage:** CALL ZREAL (*10 parameters*) **Precision:** Single (Double: DZREAL) **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZREAL1.** Finds real zeros of a real function. (To be used when initial guesses are poor.) *Proprietary Fortran subroutine in IMSL.* **Classes:** F1b **Usage:** CALL ZREAL1 (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZREAL2.** Finds real zeros of a real function. (To be used when initial guesses are good.) *Proprietary Fortran subroutine in IMSL.* **Classes:** F1b **Usage:** CALL ZREAL2 (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZRPOLY.** Finds zeros of a polynomial with real coefficients using the Jenkins-Traub method. *Proprietary Fortran subroutine in IMSL.* **Classes:** F1a1 **Usage:** CALL ZRPOLY (A, NDEG, Z, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ZSCNT.** Solve a system of nonlinear equations. *Proprietary Fortran subroutine in IMSL.* **Classes:** F2 **Usage:** CALL ZSCNT (*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ZSPOW.** Solve a system of nonlinear equations. (Uses function values only.) *Proprietary Fortran subroutine in IMSL.* **Classes:** F2 **Usage:** CALL ZSPOW(*9 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS. *On 205:* Vectorized version available.

**ZSRCH.** Generate starting points in an N dimensional space for algorithms which optimize functions of several variables or algorithms which solve simultaneous nonlinear equations. *Proprietary Fortran subroutine in IMSL.* **Classes:** G4d, F3 **Usage:** CALL ZSRCH (A, B, N, K, IP, S, M, IW, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZTEST.** Performs a one- or two-sided z-test for a specified standard deviation. *Command in MINITAB proprietary interactive system.* **Classes:** L4a1a4n **Usage:** ZTESt of mu = K assuming sigma = K, on C [; subcommand ALTERNATIVE = K.] **Precision:** Single **Availability:** 855NOS.

**ZX0LP.** Solve the linear programming problem (phase one or phase two) via the revised simplex algorithm. *Proprietary Fortran subroutine in IMSL.* **Classes:** G2a1 **Usage:** CALL ZX0LP (*16 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZX3LP.** Solve the linear programming problem via the revised simplex algorithm. (Easy to use version.) *Proprietary Fortran subroutine in IMSL.* **Classes:** G2a1 **Usage:** CALL ZX3LP (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZX4LP.** Solve the linear programming problem via the revised simplex algorithm. (Alternate easy to use version.) *Proprietary Fortran subroutine in IMSL.* **Classes:** G2a1 **Usage:** CALL ZX4LP (*13 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXCGR.** A conjugate gradient algorithm for finding the minimum of a function of n variables. *Proprietary Fortran subroutine in IMSL.* **Classes:** G1b1a **Usage:** CALL ZXCGR (*10 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXGSN.** One-dimensional unimodal function minimization using the golden section search method. *Proprietary Fortran subroutine in IMSL.* **Classes:** G1a2 **Usage:** CALL ZXGSN (F, A, B, TOL, XMIN, IER) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXGSP.** One-dimensional unimodal function minimization using the golden section search method. (Data parameters specified.) *Proprietary Fortran subroutine in IMSL.* **Classes:** G1a2 **Usage:** CALL ZXGSP (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXLSF.** One-dimensional minimization of a smooth function using safeguarded quadratic interpolation. *Proprietary Fortran subroutine in IMSL.* **Classes:** G1a1a **Usage:** CALL ZXLSF(*7 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXMIN.** Minimum of a function of n variables using a quasi-Newton method. *Proprietary Fortran subroutine in IMSL.* **Classes:** G1b1a **Usage:** CALL ZXMIN (*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXMWD.** Global minimum (with constraints) of a function of n variables. *Proprietary Fortran subroutine in IMSL.* **Classes:** G2h1a1 **Usage:** CALL ZXMWD(*11 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

**ZXSSQ.** Minimum of the sum of squares of m functions in n variables using a finite difference Levenberg-Marquardt algorithm. *Proprietary Fortran subroutine in IMSL.* **Classes:** K1b1a1, L8e1b1, L8e1b3 **Usage:** CALL ZXSSQ (*17 parameters*) **Precision:** Single **Availability:** 855NOS, 855VE, 205, 840NOS.

# Application Package Reference

*This section provides information about the application packages to which individual software modules belong. For each package, this information includes a brief description, information about portability, a reference, the name of the developer, and the name of current distributor. Also included may be a brief list of the contents of the package. For each computer on which the package is locally supported, further information may be provided about the version, level of support, how to obtain detailed documentation, and how to access the package on that computer.*

*The computers listed in this edition of GAMS are as follows.*

**855NOS** *CYBER 180/855 (NOS operating system) located in Gaithersburg, MD. For further information call 301-975-2968 (FTS-879-2968).*

**855VE** *CYBER 180/855 (NOS/VE operating system) located in Gaithersburg, MD. For further information call 301-975-2968 (FTS-879-2968).*

**205** *CYBER 200/205 (VSOS operating system) located in Gaithersburg, MD. For further information call 301-975-2968 (FTS-879-2968).*

**CAMVAX** *VAX 11/785 (VMS operating system) located in Gaithersburg, MD. For further information call 301-975-3830 (FTS-879-3830). Access restricted.*

**PC** *PCs and clones (DOS operating system). For further information call the Micro-Resource Site, Gaithersburg, MD, 301-975-3837 (FTS-879-3837).*

**840NOS** *CYBER 180/840 (NOS operating system) located in Boulder, CO. For further information call 303-497-3566 (FTS-320-3566).*

# BESPAK

## General Information

**Description** : Fortran subroutines for Bessel functions of complex argument and real (integer or fractional) order similar to earlier subroutines written with the assistance of F. W. J. Olver while the author was at NIST. A call to any one of the subroutines generates function values for fixed argument and a sequence of orders proceeding in unit steps.

**Portability** : Portable

**Reference** : D.J. Sookne. Bessel Functions I and J of Complex Argument and Integer Order. *Journal of Research of the National Bureau of Standards*, Vol 77B, Nos 3&4, Gaithersburg, MD, 1973.

**Developer** : Tel-Aviv University Computation Center, Tel-Aviv University, Tel-Aviv, Israel (David J. Sookne-Sagin)

**Distributor** : D. Lozier, NIST, Bldg 101 Room A302, Gaithersburg, MD 20899 (FTS 879-2706 or 301-975-2706)

## 855NOS Information

**Version** : 1983

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,BESPAK.`

**Module Doc** : `INVOKE,GETDOC,BESPAK,CONTENT.`

**Access** : `INVOKE,GETSRC,BESPAK,`<*name*>, where <*name*> is a subprogram name (see Module Doc).

# BMDP

## General Information

**Description** : Programs for statistical data analysis — data description, line-printer plotting, regression, analysis of variance, frequency tables, time series analysis, multivariate analysis (cluster, correlation, discriminant, factor), and life tables. (Approximately 40 programs.)

**Portability** : Proprietary

**Reference** : W.J. Dixon et al. *BMDP Statistical Software Manual.* University of California Press, Berkeley, CA, 1985.
M.A. Hill. *BMDP User's Digest*, 3rd edition. University of California Press, Berkeley, CA, 1984.

**Developer** : BMDP Statistical Software, Los Angeles, CA.

**Distributor** : BMDP Statistical Software, Inc., 1964 Westwood Blvd., Suite 202, Los Angeles, CA 90025 (213-475-5700)

## 855NOS Information

**Version** : 1987

**Support** : Full

**General Doc** : `INVOKE,GETDOC,BMD.`

**Module Doc** : `INVOKE,GETDOC,BMD,CONTENT.`

**Access** : `INVOKE,BMD?`

## Itemized Contents

**L1 : Data summarization**

P1D P2D P4D P8D PAM

**L2 : Data manipulation**

P1S PDM

**L3 : Elementary statistical graphics**

P1T P5D P6D

**L4 : Elementary data analysis**

P3D P3S P5D

**L7 : Analysis of variance**

P1V P2V P3S P3V P4V P7D P8V P9D

**L8 : Regression**

P1R P3R P5R P6R P9M P9R PAR PLR
P2R P4R

**L9 : Categorical data analysis**

P4F

**L10 : Time series analysis**

P1T P2T

**L11 : Correlation analysis**

P6R

**L12 : Discriminant analysis**

P7M

**L13 : Covariance structure models**

P4M P6M P8M

**L14 : Cluster analysis**

P1M P2M P3M PKM

**L15 : Life testing and survival analysis**

P1L P2L

**N : Data handling**

P4D

# CGLIB

## General Information

**Description** : A Fortran subroutine library for solving various problems in computational geometry.

**Portability** : Portable

**Reference** : None

**Developer** : Various authors

**Distributor** : S. Howe, NIST, Bldg 225 Room B146, Gaithersburg, MD 20899 (FTS 879-3807 or 301-975-3807)

## 855NOS Information

**Version** : 1

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,CGLIB.`

**Module Doc** : `INVOKE,GETDOC,CGLIB,`<*module*>`.`

**Access** : `INVOKE,GETLIB,CGLIB.`
`LIBRARY,CGLIB.`

## Itemized Contents

**ADNODE** Adds a node to a triangulation of a set of points in the plane producing a new triangulation.

**AREA** Given a sequence of points in the plane, this function computes the area bounded by the closed polygonal curve which passes through the points in the specified order.

**BNODES** Given a triangulation of a set of points in the plane, returns the nodes on the boundary of the convex hull of the set of points.

**DECOMP** Decomposes a not-necessarily convex polygon into a sequence of convex polygons.

**DELETE** Deletes a boundary edge from a triangulation of a set of points in the plane.

**EDGE** Given a triangulation of a set of points in the plane, forces an edge between two given points in the triangulation.

**TRMESH** Creates a Thiessen triangulation of n arbitrarily spaced points in the plane.

**VOR** Constructs a 2-d Voronoi diagram for a set of sites in a simply connected bounded planar polygonal region.

**VORU** Constructs a 2-d Voronoi diagram for a set of sites in the Euclidean plane.

# CMLIB

## General Information

| | |
|---|---|
| **Description** | : A collection of high-quality, easily transportable Fortran subroutine sublibraries solving standard problems in many areas of mathematics and statistics. |
| **Portability** | : Portable |
| **Reference** | : See information about individual sublibraries below |
| **Developer** | : See information about individual sublibraries below |
| **Distributor** | : See information about individual sublibraries below |

## 855NOS Information

| | |
|---|---|
| **Version** | : 3 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,CMLIB.` |
| **Module Doc** | : `INVOKE,GETDOC,CMLIB,`<*module*>`.` |
| **Access** | : `INVOKE,GETLIB,CMLIB.`<br>`LIBRARY,CMLIB.` |

## 855VE Information

| | |
|---|---|
| **Version** | : 3 |
| **Support** | : Full |
| **General Doc** | : `FETCH CMLIB USAGE` |
| **Module Doc** | : `FETCH CMLIB` <*module*> `DOC` |
| **Access** | : `CMLIB AL = YES` |

## 205 Information

| | |
|---|---|
| **Version** | : 3 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,CMLIB.` (Under 855NOS) |
| **Module Doc** | : `INVOKE,GETDOC,CMLIB,`<*module*>`.` (Under 855NOS) |
| **Access** | : `PATTACH,MATHPOOL.`<br>`LOAD,BINARY,LIB=CMLIB.` |

## CAMVAX Information

**Version** : 3

**Support** : Limited

**General Doc** : `GETDOC CMLIB`

**Module Doc** : `GETDOC CMLIB` <*module*>

**Access** : `LINK` <*main program*>`,SYS$USER:CMLIB/LIB`

## 840NOS Information

**Version** : 3

**Support** : Full

**General Doc** : `GET,CMDOC/UN=CAMLIB.`
`CMDOC.`

**Module Doc** : `GET,CMDOC/UN=CAMLIB.`
`CMDOC,`<*module*>`.`

**Access** : `ATTACH,CMLIB/UN=CAMLIB,NA.`
`LIBRARY,CMLIB.`

## Itemized Contents

CMLIB is a set of portable Fortran subprogram packages which have been collected together to form a single library. Each of these components is called a *sublibrary.* The following is a description of the sublibraries which make up CMLIB. The entries take the form:

**SUBLIBRARY NAME:** A short description of the sublibrary. | **Version:** a date | **Author:** the author(s) | **Reference:** a book, journal article, or report which describes the software. | **Contact:** an address where queries about the software may be made.

Many of these sublibraries are available through netlib, a software distribution service operated by Argonne National Laboratories and AT&T Bell Telephone Laboratory. These are indicated by the text *Available from netlib* at the end of their entry. To obtain information about how to use netlib, send an electronic mail message with the text **send index** to *netlib@ornl.gov*, *netlib@research.att.com*, or *research!netlib*.

---

**ADAPT:** Multidimensional adaptive quadrature. ADAPT will integrate a user specified function on a hyper-rectangle of dimension 2 through 20 to a specified tolerance. | **Version:** 1984 | **Author:** A.C. Genz | **Reference:** A.C. Genz and A.A. Malik, Algorithm 019: An adaptive algorithm for numerical integration over an n-dimensional rectangular region, *Journal of Computational and Applied Mathematics*, vol. 6, no. 4, 1980, pp. 295–299 | **Contact:** Computer Science Department, Washington State University, Pullman, WA 99164 (A.C. Genz)

**AMOSLIB:** A collection of special function routines with particular emphasis on the special functions of statistics. | **Version:** 1980 | **Author:** D. Amos | **Reference:** D. Amos, Sandia National Laboratories Report SAND 77-1390, 1977 | **Contact:** Sandia National Laboratories, Albuquerque, NM 87185 (D. Amos)

### BLAS associated sublibraries

**BLAS:** Basic linear algebra subroutines. Perform various elementary matrix and vector operations. | **Version:** 1979 | **Author:** C.L. Lawson, R.J. Hanson, D.R. Kincaid, and F.T. Krogh | **Reference:** C.L. Lawson, R.J. Hanson, D.R. Kincaid, and F.T. Krogh, Basic linear algebra subprograms for Fortran use, *ACM Transactions on Mathematical Software*, vol. 5, no. 3, 1979, pp. 308–323 | **Contact:** Jet Propulsion Laboratory, Pasadena, CA 91103 (C.L.Lawson) | Available from netlib.

**XBLAS:** Extended basic linear algebra subroutines. Perform various matrix and vector operations not found in the BLAS. | **Version:** 1980 | **Author:** C.L. Lawson, R.J. Hanson, D.R. Kincaid, F.T. Krogh, J.J. Dongarra and D.K. Kahaner | **Reference:** none | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**BOCLS:** Solves bounded and linearly constrained linear least squares problems. There are two user caliable routines - one which solves the general problem and one which solves only problems with simple bounds. Only a double precision version is available. | **Version:** 1984 | **Author:** R. Hanson | **Reference:** R. Hanson, Linear least squares with bounds and linear constraints, *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 3, 1986, pp. 826–834 | **Contact:** Applied Dynamics International, 3800 Stone School Rd., Ann Arbor, MI 48104 (R.J. Hanson)

### BSPLINE associated sublibraries

**BSPLINE:** Subroutines for computing with piecewise polynomials (B-splines). Includes interpolation, differentiation and integration with B-splines. | **Version:** 1980 | **Author:** D. Amos | **Reference:** C. de Boor, *A Practical Guide to Splines*, Springer-Verlag, 1978 | **Contact:** Sandia National Laboratories, Albuquerque, NM 87185 (D. Amos)

**DBSPLIN:** Double precision version of BSPLINE

**BVSUP:** Solves systems of linear two-point boundary value problems. | **Version:** 1982 | **Author:** M.R. Scott and H.A. Watts | **Reference:** M.R. Scott and H.A. Watts, Computational solution of linear two-point boundary value problems via orthonormalization, *SIAM Journal on Numerical Analysis*, vol. 14, 1977, pp. 40–70 | **Contact:** Sandia National Laboratories, Albuquerque, NM 87185 (H.A. Watts)

**CDRIV:** See SDRIV associated sublibraries.

**CLUSTER:** Subroutines for cluster analysis and related line printer graphics. It includes routines for clustering variables and/or observations using algorithms such as direct joining and splitting, K-means, minimum mutations, and routines for estimating values. | **Version:** 1975 | **Author:** J.A. Hartigan | **Reference:** J.A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, Inc., New York, 1975 | **Contact:** Dept. of Statistics, Yale University, New Haven, CT 06520 (J.A. Hartigan)

**CPQR79:** Finds all zeros of real and complex polynomials via eigenvalue methods. | **Version:** 1982 | **Author:** W. Vandevender | **Reference:** none | **Contact:** Sandia National Laboratories, Albuquerque, NM 87185 (W. Vandevender)

**CPZERO:** Computes all the zeros of polynomials with real or complex coefficients. Error bounds are also computed. | **Version:** 1982 | **Author:** D.K. Kahaner | **Reference:** none | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

### DASSL associated sublibraries

**DDASSL:** Double precision version of SDASSL

**SDASSL:** Solves the system of differential/algebraic equations of the form $g(t, y, y') = 0$, with given initial values. | **Version:** 1983 | **Author:** L.R. Petzold | **Reference:** L.R. Petzold, Differential/algebraic equations are not ODE's, *SIAM Journal on Scientific and Statistical Computing*, vol. 3, no. 3, 1982, pp. 367–384 | **Contact:** Lawrence Livermore National Laboratory, Livermore, CA (L.R. Petzold) | Available from netlib.

**DBSPLIN:** See BSPLINE associated sublibraries.

**DDASSL:** See DASSL associated sublibraries.

**DDRIV:** See SDRIV associated sublibraries.

**DEPAC:** Solves systems of first order ordinary differential equations with arbitrary initial data. | **Version:** 1982 | **Author:** L. Shampine and H.A. Watts | **Reference:** none | **Contact:** Southern Methodist University, Dallas, TX 75275 (L. Shampine); Sandia National Laboratories, Albuquerque, NM 87185 (H.A. Watts)

**DNL2SN:** See NL2SN associated sublibraries.

**DQRLSS:** See LINPACK associated sublibraries.

**DTENSBS:** See TENSBS associated sublibraries.

### EISPACK associated sublibraries

**EISPACK:** Solves various linear algebraic eigenvalue problems. | **Version:** 1983 | **Author:** J.M. Boyle, J.J. Dongarra, B.S. Garbow, Y. Ikebe, V.C. Klema, C.B. Moler, and B.T. Smith | **Reference:** B.T. Smith et al., *Matrix Eigensystem Routines – EISPACK Guide*, Spring-Verlag, New York, 1976; B.S. Garbow et al., *Matrix Eigensystem Routines – EISPACK Guide Extension*, Springer-Verlag, NY, 1977 | **Contact:** Argonne National Laboratory, Argonne, IL 60439 (B.S. Garbow) | Available from netlib.

**LICEPAK:** Solves linear algebraic eigenvalue problems. (Provides an interface to the EISPACK sublibrary). | **Version:** 1982 | **Author:** D.K. Kahaner, C.B. Moler and G.W. Stewart | **Reference:** none | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**FC:** Solves constrained least squares problems of various types. Includes fitting of piecewise polynomial curves to data in the least squares sense, linear least squares problem for band matrices, and general linearly constrained least squares problems. | **Version:** 1982 | **Author:** K.H. Haskell, C.L. Lawson, R.J. Hanson | **Reference:** R.J. Hanson, Sandia Laboratories Report SAND 78-1291, 1978; R.J. Hanson and K.H. Haskell, Algorithm 587 — Two algorithms for the linearly constrained $L_2$ problem, *ACM Transactions on Mathematical Software*, vol. 8, 1982, pp. 323–333; C.L. Lawson and R.J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974 | **Contact:** Applied Dynamics International, 3800 Stone School Rd., Ann Arbor, MI 48104 (R.J. Hanson)

**FCNPAK:** Subprograms to compute various special functions not readily available elsewhere. At present the package contains subroutines for the associated Legendre (Ferrers) functions and the normalized Legendre polynomials. | **Version:** 1987 | **Author:** D.W. Lozier and J. Smith | **Reference:** J.M. Smith, F.J.W. Olver and D.W. Lozier, Extended range arithmetic and Legendre polynomials, *ACM Transactions on Mathematical Software*, vol. 7, no. 1, 1981, pp. 93–105 | **Contact:** NIST, Gaithersburg, MD 20899 (D.W. Lozier)

**FFTPACK:** Subroutines for computing the fast Fourier transform in various forms. | **Version:** 1983 | **Author:** P.N. Swarztrauber | **Reference:** P.N. Swarztrauber, Vectorizing the FFT's, in *Parallel Computations* (G. Rodrigue, ed.), Academic Press, 1982, pp. 51–83 | **Contact:** National Center for Atmospheric Research, Boulder, CO 80307 (P.N. Swarztrauber) | Available from netlib.

**FISHPAK:** Solves separable elliptic boundary value problems in two and three dimensions in a variety of coordinate systems. | **Version:** 1981 | **Author:** P.N. Swarztrauber and R.A. Sweet | **Reference:** none | **Contact:** National Center for Atmospheric Research, Boulder, CO 80307 (P.N. Swarztrauber); University of Colorado at Denver (R.A. Sweet) | Available from netlib.

**FNLIB:** Portable special function routines. | **Version:** 1983 | **Author:** L.W. Fullerton | **Reference:** L.W. Fullerton, in *Portability of Numerical Software*, Springer-Verlag, New York, 1977, pp. 452–483 | **Contact:** none | Available from netlib.

**LICEPAK:** See EISPACK associated sublibraries.

**LINPACK associated sublibraries**

**DQRLSS:** Double precision version of SQRLSS

**LINDRV:** Subprograms to solve various types of linear systems of algebraic equations. Provides a simplified interface to the LINPAKC, LINPAKD, and LINPAKS sublibraries. | **Version:** 1982 | **Author:** E. Voorhees | **Reference:** none | **Contact:** none

**LINPAKC:** Complex precision version of LINPAKS

**LINPAKD:** Double precision version of LINPAKS

**LINPAKS:** Analyze and solve various systems of linear algebraic equations. (Single precision version of LINPACK.) | **Version:** 1982 | **Author:** J.J. Dongarra, C.B. Moler, J.R. Bunch and G.W. Stewart | **Reference:** J.J. Dongarra et al., *LINPACK User's Guide*, SIAM, Philadelphia, 1979 | **Contact:** Argonne National Laboratory, Argonne, IL 60439 (J.J. Dongarra) | Available from netlib.

**SQRLSS:** Solves linear least square problems in the matrix form Ax=b. Easy-to-use driver for LINPACK routines. | **Version:** 1980 | **Author:** D.K. Kahaner | **Reference:** D. Kahaner, C. Moler and S. Nash, *Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ, 1989 | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**LOTPS:** Produces a smooth interpolant to scattered, i.e. non-gridded, data in the plane. | **Version:** 1982 | **Author:** R. Franke | **Reference:** R. Franke, Smooth interpolation of scattered data by local thin-plate splines, *Computers and Mathematics with Applications*, vol. 8, 1982 | **Contact:** Naval Postgraduate School, Monterey, CA 93940 (R. Franke)

**MACHCON:** Functions that return machine-dependent constants. | **Version:** 1983 | **Author:** P.A. Fox, A.D. Hall and N.L. Schryer | **Reference:** P.A. Fox, A.D. Hall and N.L. Schryer, Algorithm 528: Framework for a portable library, *ACM Transactions on Mathematical Software*, vol. 4, no. 2, 1978, pp. 177–188 | **Contact:** Bell Laboratories, Murray Hill, NJ 07974 (N.L. Schryer) | Available from netlib.

**MXENTRP:** Subprograms for computing maximum entropy spectrum estimates for equally spaced time series data. | **Version:** 1983 | **Author:** M.T. Silvia | **Reference:** M.T. Silvia and E.A. Robinsonnone, *Deconvolution of Geophysical Time Series in the Exploration for Oil and Natural Gas*, Elsevier Scientific Publishing Company, New York, 1979 | **Contact:** US Naval Underwater Systems Center, Newport, RI 02841 (Manuel T. Silvia)

**NL2SN:** See NL2SOL associated sublibraries.

**NL2SOL associated sublibraries**

**NL2SN:** Solves nonlinear least squares problems and general optimization problems. | **Version:** 1981 | **Author:** D. Gay | **Reference:** J.E. Dennis, Jr., D.M. Gay, and R.E. Welsch, Algorithm 573: NL2SOL — An adaptive nonlinear least squares algorithm, *ACM Transactions on Mathematical Software*, vol. 7, no. 3, 1981, pp. 369–383 | **Contact:** Bell Laboratories, 600 Mountain Avenue, Murray Hill, NJ 07974 (D.M. Gay)

**DNL2SN:** Double precision version of NL2SN

**ODRPACK:** Subprograms for fitting a linear or nonlinear model to data. It is designed primarily for instances when the independent as well as the dependent variables have significant errors, implementing a highly efficient algorithm for solving the weighted orthogonal distance regression problem. It can also be used to solve the ordinary least squares problem where all of the errors are attributed to the observations of the dependent variable. | **Version:** 1.3, 1987 | **Author:** P.T. Boggs, R.H. Byrd, J.R. Donaldson and R.B. Schnabel | **Reference:** P.T. Boggs, R.H. Byrd, J.R. Donaldson and R.B. Schnabel, Software for weighted orthogonal distance regression, *ACM Transactions on Mathematical Software*, 1989, to appear | **Contact:** NIST, Boulder, CO 80303 (J.R. Donaldson)

### PCHIP associated sublibraries

**PCHIPD:** Double precision version of PCHIPS

**PCHIPS:** Produces aesthetic looking interpolants to univariate data by using piecewise cubic Hermite functions. | **Version:** 1987 | **Author:** F.N. Fritsch | **Reference:** R.E. Carlson and F.N. Fritsch, Monotone piecewise cubic interpolation, *SIAM Journal on Numerical Analysis*, vol. 17, 1980, pp. 238–246 | **Contact:** Mathematics and Statistics Division, Lawrence Livermore National Laboratory, P.O. Box 808 (L-316), Livermore, CA 94550 (F. Fritsch)

**Q1DA:** Automatic evaluation of one-dimensional integrals of a user-defined function of one variable. | **Version:** 1982 | **Author:** D.K. Kahaner | **Reference:** none | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

### QUADPACK associated sublibraries

**QUADPKD:** Double precision version of QUADPAKS

**QUADPKS:** Subprograms for evaluating definite integrals of functions of one variable; including singular integrands and infinite intervals. (Single precision version of QUADPACK.) | **Version:** 1984 | **Author:** R. Piessens, E. de Donker, and D.K. Kahaner | **Reference:** R. Piessens et al., *QUADPACK: A Subroutine Package for Automatic Integration*, Springer-Verlag, New York, 1983 | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**RV:** A collection of portable pseudo-random number generators. | **Version:** 1983 | **Author:** J.L. Blue, D.K. Kahaner and G. Marsaglia | **Reference:** D. Kahaner, C. Moler and S. Nash, *Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ, 1989 | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**SDASSL:** See DASSL associated sublibraries.

### SDRIV associated sublibraries

**SDRIV:** Solves initial value problems for systems of ordinary differential equations, including stiff systems. | **Version:** 1985 | **Author:** D.K. Kahaner and D. Sutherland | **Reference:** D. Kahaner, C. Moler and S. Nash, *Numerical Methods and Software*, Prentice-Hall, Englewood Cliffs, NJ, 1989 | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**CDRIV:** Complex precision version of SDRIV

**DDRIV:** Double precision version of SDRIV

**SGLSS:** Solves AX=B for either overdetermined systems (least squares problems) or underdetermined systems (least length problems) with reliable rank deficiency determination. Uses Householder transformations. | **Version:** 1982 | **Author:** T. Manteuffel | **Reference:** T. Manteuffel, Sandia National Laboratories Report SAND 80-0655, 1980 | **Contact:** Los Alamos National Laboratory, Los Alamos, NM 87545 (T. Manteuffel)

**SLRPACK:** A collection of subprograms for special types of simple regression, including errors in variables. | **Version:** 1985 | **Author:** S.E. Howe, R. Lindstrom, M.G. Sklar and R.D. Armstrong | **Reference:** D.S. Riggs, J.A. Guarnieri and S. Addelman, Fitting straight lines when both variables are subject to error, *Life Sciences*, vol. 22, 1978, pp. 1305-1360; D. York, Least-square fitting of a straight line, *Canadian Journal of Physics*, 44, 1966, pp. 1079-1086; J.H. Williamson, Least-squares fitting of a straight line, *Canadian Journal of Physics*, vol. 46, 1968, pp. 1845-1847; M. Sklar and R. Armstrong, An algorithm for discrete Chebychev curve fitting for the simple model using a dual linear programming approach, *Communications in Statistics: Simulation and Computation*, vol. 13, 1984, pp. 555-569 | **Contact:** NIST, Gaithersburg, MD 20899 (S.E. Howe)

**SLVBLK:** Solves linear systems of algebraic equations where the coefficient matrix is in "almost block diagonal" form. | **Version:** 1980 | **Author:** C. de Boor | **Reference:** C. de Boor and R. Weiss, SOLVEBLOK: A package for solving almost block diagonal linear systems, *ACM Transactions on Mathematical Software*, vol. 6, no. 1, 1980, pp. 80-91 | **Contact:** University of Wisconsin, Madison, WI 53706 (C. de Boor)

**SNLS1E:** A suite of codes for nonlinear least squares problems and systems of nonlinear equations. | **Version:** 1982 | **Author:** K.L. Hiebert | **Reference:** J.J. More, D.C. Sorensen, B.S. Garbow, and K.E. Hillstrom, The MINPACK project, in *Sources and Development of Mathematical Software* (W.R. Cowell, ed.), Prentice-Hall, Englewood Cliffs, NJ, 1984 | **Contact:** none

**SPLP:** Solves linear programming problems involving at most a few thousand constraints and variables. Takes advantage of sparsity in the constraint matrix. | **Version:** 1983 | **Author:** R.J. Hanson and K.L. Hiebert | **Reference:** R.J. Hanson and K.L. Hiebert, A Sparse Linear Programming Subprogram, Sandia National Laboratories Report, SAND81-0297, 1981 | **Contact:** Applied Dynamics International, 3800 Stone School Rd., Ann Arbor, MI 48104 (R.J. Hanson)

**SQRLSS:** See LINPACK associated sublibraries.

**SSORT:** Fast in-core sorting of arrays. | **Version:** 1985 | **Author:** R.E. Jones, J.A. Wisniewski, D.K. Kahaner and F.E. Sullivan | **Reference:** none | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**SUDSODS:** Solves overdetermined and underdetermined systems of linear equations in the least squares sense. | **Version:** 1982 | **Author:** H.A. Watts | **Reference:** H.A. Watts, Sandia National Laboratories Report SAND 77-0683, 1977 | **Contact:** Sandia National Laboratories, Albuquerque, NM 87185 (H.A. Watts)

### TENSBS associated sublibraries

**DTENSBS:** Double precision version of TENSBS

**TENSBS:** Interpolation of gridded data in two or three dimensions using tensor products of one-dimensional B-splines. | **Version:** 1982 | **Author:** R.F. Boisvert | **Reference:** C. de Boor, *A Practical Guide to Splines*, Springer-Verlag, 1978 | **Contact:** NIST, Gaithersburg, MD 20899 (R.F. Boisvert)

**TWODQ:** Adaptively evaluates the integral of a user specified function f(x,y) on one or more triangles in the plane. Computation may be continued to greater accuracy without penalty. Absolute or relative error control. Can handle singular integrands. | **Version:** 1984 | **Author:** D.K. Kahaner and O.W. Rechard | **Reference:** D.K. Kahaner and O.W. Rechard, TWODQD: an adaptive routine for two-dimensional integration, *Journal of Computational and Applied Mathematics*, vol. 17, 1987, pp. 215–234 | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**UNCMIN:** Two subprograms for the unconstrained minimization problem: minimize the real-valued function f of n variables, where f is assumed to be twice continuously differentiable. | **Version:** 1984 | **Author:** R.B. Schnabel, J.E. Koontz and B.E. Weiss | **Reference:** R.B. Schnabel, J.E. Koontz and B.E. Weiss, A modular system of algorithms for unconstrained minimization, *ACM Transactions on Mathematical Software*, vol. 11, no. 4, 1985, pp. 419–440 | **Contact:** University of Colorado, Boulder, CO (R.B. Schnabel)

**VFFT:** Subprograms for the fast Fourier transform of multiple real sequences. | **Version:** 2, 1987 | **Author:** R.A. Sweet, L.L. Lindgren and R.F. Boisvert | **Reference:** P.N. Swarztrauber, Vectorizing the FFT's, in *Parallel Computations* (G. Rodrigue, ed.), Academic Press, 1982, pp. 51–83 | **Contact:** University of Colorado at Denver, Denver, CO (R.A. Sweet); NIST, Gaithersburg, MD 20899 (R.F. Boisvert)

**VHS3:** Subprograms for the solution of a three-dimensional Helmholtz equation on a staggered grid. | **Version:** 1, 1985 | **Author:** R.A. Sweet | **Reference:** none | **Contact:** University of Colorado at Denver, Denver, CO (R.A. Sweet)

**VSFFT:** Subprograms for the fast Fourier transform of multiple real sequences defined on a staggered grid. | **Version:** 1, 1985 | **Author:** L.L. Lindgren and R.A. Sweet | **Reference:** none | **Contact:** NIST, Boulder, CO

80303 (L.L. Sweet); University of Colorado at Denver, Denver, CO (R.A. Sweet)

**XBLAS:** See BLAS associated sublibraries.

**XERROR:** Error handling utilities. | **Version:** 1982 | **Author:** R.E. Jones and D.K. Kahaner | **Reference:** R.E. Jones and D.K. Kahaner, XERROR, the SLATEC error-handling package, *Software–Practice and Experience*, vol. 12, 1983, pp. 251–257 | **Contact:** NIST, Gaithersburg, MD 20899 (D.K. Kahaner)

**ZEROIN:** Finds zeros of a function of one variable. | **Version:** 1979 | **Author:** L. Shampine and H.A. Watts | **Reference:** L. Shampine et al., *Numerical Computing: An Introduction*, Saunders, 1973 | **Contact:** Southern Methodist University, Dallas, TX 75275 (L. Shampine); Sandia National Laboratories, Albuquerque, NM 87185 (H.A. Watts)

---

# Cyber 205 Vectorized Version

Portions of the following CMLIB sublibraries have been vectorized in CMLIBV, the vectorized version of CMLIB available on the NIST Cyber 205.

| | |
|---|---|
| BLAS | basic linear algebra subprograms |
| EISPACK | eigenvalue problems |
| LICEPAK | eigenvalue problems (EISPACK drivers) |
| LINPACK | linear algebraic systems |
| LINDRV | linear algebraic systems (LINPACK drivers) |
| VHS3 | Poisson or Helmholtz equation in three dimensions |
| VFFT | multiple real periodic fast Fourier transforms |
| VSFFT | multiple staggered grid fast Fourier transforms |

Further details concerning the vectorization of each of these sublibraries is given below.

## BLAS (20 subprograms)

A set of Basic Linear Algebra Subprograms. Only single precision and complex precision routines are affected. These routines perform operations such as dot products, scalar × vector, vector + scalar × vector, vector copy, vector swap, etc. The BLAS are described in J.J. Dongarra, et al., *LINPACK Users' Guide*, SIAM, Philadelphia, 1979. The vectorized BLAS in CMLIBV are assembly language versions which were provided by Control Data Corp. The vector versions have the same calling sequences as the scalar version; however, *negative storage increments are not allowed* in the vector version.

## EISPACK (36 subprograms)

Suprograms for computing eigenvalues and eigenvectors of matrices. Vectorized versions of twenty-five the EISPACK subprograms have been obtained from the Purdue University Computer Center (PUCC). The following EISPACK drivers use the vectorized routines (unless otherwise stated the routine computes eigenvalues and eigenvectors of the given matrix):

| | |
|---|---|
| CH | Complex Hermitian matrix. |
| RG | Real general matrix. (Only the path computing eigenvalues only is fully vectorized.) |
| RGG | Real generalized eigenproblem $Ax = \lambda Bx$. |
| RS | Real symmetric matrix. |
| RSB | Real symmetric band matrix. (Partially vectorized.) |
| RSG | Real symmetric generalized eigenproblem $Ax = \lambda Bx$. |
| RSGAB | Real symmetric generalized eigenproblem $ABx = \lambda x$. |
| RSGBA | Real symmetric generalized eigenproblem $BAx = \lambda x$. (Only the path computing eigenvalues only is fully vectorized.) |
| RSP | Real symmetric packed matrix. |

| | |
|---|---|
| RST | Real symmetric tridiagonal matrix. (Only the path computing both eigenvalues and eigenvectors is fully vectorized.) |
| RT | Special real tridiagonal matrices. (Partially vectorized.) |
| SVD | Compute the singular value decomposition of an arbitrary real rectangular matrix. |

The EISPACK subprograms explicitly vectorized at PUCC are the following:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BALANC | BALBAK | ELMHES | ELTRAN | HQR | HTRIB3 | HTRIBK | HTRIDI |
| IMTQL2 | QZHES | QZIT | QZVAL | QZVEC | REBAK | REDUC | REDUC2 |
| SVD | TQL1 | TQL2 | TQLRAT | TRBAK1 | TRBAK3 | TRED1 | TRED2 |
| TRED3 | | | | | | | |

### LICEPAK (4 subprograms)

Subprograms for computing eigenvalues and eigenvectors of matrices. These routines are easy-to-use drivers for EISPACK. Thus, some of these routines show improved performance due to the vectorization of EISPACK (see above). The affected routines are:

| | |
|---|---|
| CHIEV | Complex Hermitian matrix. |
| SGEEV | Real general matrix. (Partially vectorized.) |
| SSIEV | Real symmetric matrix. |
| SSPEV | Real symmetric packed matrix. |

### LINPACK (95 subprograms)

Subprograms for solving systems of linear algebraic equations. Only the single and complex routines are affected. LINPACK is described in J.J. Dongarra, et al., *LINPACK Users' Guide*, SIAM, Philadelphia, 1979. The vectorized LINPACK were provided by Control Data Corp. Vector speedups in these routines are obtained primarily through the use of vectorized BLAS (see above). A speedup of 6 over the scalar version is obtained, for example, when solving a full system of order 300 with the routines SGEFA/SGESL.

### LINDRV (20 subprograms)

Subprograms for solving systems of linear algebraic equations. These routines are easy-to-use drivers for LINPACK (described above). Although the LINDRV routines are identical in both CMLIB and CMLIBV, the CMLIBV versions will show the same increase in performance as LINPACK.

### VHS3 (4 subprograms)

Subprograms for solving the Helmholtz equation in a rectangular three-dimensional region with Dirichlet, Neumann, or periodic boundary conditions. The routines are based on a finite difference approximation on a staggered grid, and the discrete problem is solved using fast Fourier transform techniques.

### VFFT (13 subprograms)

Subprograms for computing the fast Fourier transform of multiple real, periodic sequences. The following transforms are provided:

- real, periodic transform
- sine transform
- cosine transform
- sine quarter-wave transform (odd wave numbers only)

- cosine quarter-wave transform (odd wave numbers only)

These routines have been vectorized by transforming multiple sequences in parallel rather than by vectorizing the FFT algorithm itself. Thus, these routines are appropriate for the Cyber 205 only when the number of sequences to be transformed is large. When less than five sequences are to be transformed, the scalar version (or the alternate CMLIB routines in FFTPACK) will run faster than the vectorized VFFT routines.

### VSFFT (13 subprograms)

Subprograms for computing the fast Fourier transform of multiple real sequences defined on a staggered grid. (A staggered grid is one on which boundary values are imposed midway between grid points rather than at a grid point.) Such transforms may be used to efficiently solve Poisson-type partial differential equations. These routines have been vectorized by transforming multiple sequences in parallel rather than by vectorizing the FFT algorithm itself. Thus, these routines are appropriate for the Cyber 205 only when the number of sequences to be transformed is large. When less than five sequences are to be transformed, the scalar version will run faster than the vectorized VSFFT routines.

# Collected Algorithms of the ACM

## General Information

**Description** : The Collected ALGOrithms of the Association for Computing Machinery (ACM) is a collection of programs and subroutines covering a wide range of topics in mathematics and computer science. The algorithms are published by the journal, *ACM Transactions on Mathematical Software* (TOMS), and are reasonably portable. They are available in source form only.

**Portability** : Portable

**Reference** : Algorithms Policy. *ACM Transactions on Mathematical Software*, 12(2):171-174, 1986.

**Developer** : The Association for Computing Machinery, 11 W 42nd St, New York, NY 10036

**Distributor** : ACM Algorithms Distribution Service, c/o IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Blvd., Houston, TX 77042-3020

## 855NOS Information

**Version** : Algorithms 493–664 (TOMS Volumes 1–14)

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,CALGO.`

**Module Doc** : `INVOKE,GETDOC,CALGO,CONTENT.`

**Access** : `INVOKE,GETSRC,CALGO,A`<*algorithm number*>.

## Itemized Contents

Volume and Issue (V,I) refer to the journal *ACM Transactions on Mathematical Software.*

| Alg. | V,I | Lines | Title |
|---|---|---|---|
| 493 | 1,2 | 721 | Zeroes of a Real Polynomial |
| 494 | 1,3 | 155 | PDEONE, Solution of Systems of Partial Differential Equations |
| 495 | 1,3 | 298 | Solution of an Over Determined System of Linear Equations in the Chebyshev Norm |
| 496 | 1,3 | 430 | The LZ Algorithm to Solve the Generalized Eigenvalue Problem for Complex Matrices |
| 497 | 1,4 | 607 | Automatic Integration of Functional Differential Equations |
| 498 | 1,4 | 348 | Airy Functions Using Chebyshev Series Approximations |
| 499 | 2,1 | 746 | An Efficient Scanning Technique |
| 500 | 2,1 | 344 | Minimization of Unconstrained Multivariate Functions |
| 501 | 2,1 | 611 | Fortran Translation of Algorithm 409, Discrete Chebyshev Curve Fit |
| 502 | 2,1 | 328 | Dependence of Solution of Nonlinear Systems on a Parameter |
| 503 | 2,2 | 1927 | An Automatic Program for Fredholm Integral Equations of the Second Kind |
| 504 | 2,2 | 555 | GERK: Global Error Estimation for Ordinary Differential Equations |

| | | | |
|---|---|---|---|
| 505 | 2,2 | 144 | A List Insertion Sort for Keys with Arbitrary Key Distribution |
| 506 | 2,3 | 563 | HQR3 and EXCHNG: Fortran Subroutines for Calculating and Ordering the Eigenvalues of a Real Upper Hessenberg Matrix |
| 507 | 2,3 | 88 | Procedures for Quintic Natural Spline Interpolation |
| 508 | 2,4 | 565 | Matrix Bandwidth and Profile Reduction |
| 509 | 2,4 | 227 | A Hybrid Profile Reduction Algorithm |
| 510 | 2,4 | 326 | Piecewise Linear Approximations to Tabulated Data |
| 511 | 3,1 | 1512 | CDC 6600 Subroutines IBESS and JBESS for Bessel Functions $I_\nu(x)$ and $J_\nu(x)$, $x \geq 0$, $\nu \geq 0$ |
| 512 | 3,1 | 140 | A Normalized Algorithm for the Solution of Positive Definite Symmetric Quindiagonal Systems of Linear Equations |
| 513 | 3,1 | 108 | Analysis of In-Situ Transposition |
| 514 | 3,2 | 76 | A New Method of Cubic Curve Fitting Using Local Data |
| 515 | 3,2 | 48 | Generation of a Vector from the Lexicographical Index |
| 516 | 3,2 | 415 | An Algorithm for Obtaining Confidence Intervals and Point Estimates Based on Ranks in the Two-Sample Location Problem |
| 517 | 3,2 | 383 | A Program for Computing the Condition Numbers of Matrix Eigenvalues without Computing Eigenvectors |
| 518 | 3,3 | 63 | Incomplete Bessel Function $I_0$: The Von Mises Distribution |
| 519 | 3,3 | 351 | Three Algorithms for Computing Kolmogorov-Smirnov Probabilities with Arbitrary Boundaries and Certification of Algorithm 487 |
| 520 | 3,3 | 498 | An Automatic Revised Simplex Method for Constrained Resource Network Scheduling |
| 521 | 3,3 | 237 | Repeated Integrals of the Coerror Function |
| 522 | 3,4 | 1404 | ESOLVE, Congruence Techniques for the Exact Solution of Integer Systems of Linear Equations |
| 523 | 3,4 | 368 | CONVEX, A New Convex Hull Algorithm for Planar Sets |
| 524 | 4,1 | 8405 | MP, A Fortran Multiple-Precision Arithmetic Package |
| 525 | 4,1 | 2251 | ADAPT, Adaptive Smooth Curve Fitting |
| 526 | 4,2 | 1800 | Bivariate Interpolation and Smooth Surface Fitting for Irregularly Distributed Data Points |
| 527 | 4,2 | 2656 | A Fortran Implementation of the Generalized Marching Algorithm |
| 528 | 4,2 | 2844 | Framework for a Portable Library |
| 529 | 4,2 | 280 | Permutations to Block Triangular Form |
| 530 | 4,3 | 612 | An Algorithm for Computing the Eigensystem of Skew-Symmetric Matrices and a Class of Symmetric Matrices |
| 531 | 4,3 | 684 | Contour Plotting |
| 532 | 4,4 | 5760 | Software for Roundoff Analysis |
| 533 | 4,4 | 1548 | NSPIV, A Fortran Subroutine for Sparse Gaussian Elimination with Partial Pivoting |
| 534 | 4,4 | 1152 | STINT: STiff (differential equations) INTegrator |
| 535 | 4,4 | 1152 | The QZ Algorithm to Solve the Generalized Eigenvalue Problem for Complex Matrices |
| 536 | 5,1 | 720 | An Efficient One-Way Enciphering Algorithm |
| 537 | 5,1 | 360 | Characteristic Values of Mathieu's Differential Equations |
| 538 | 5,1 | 1512 | Eigenvectors and Eigenvalues of Real Generalized Symmetric Matrices by Simultaneous Iteration |
| 539 | 5,3 | 4536 | Basic Linear Algebra Subprogram for Fortran Usage |
| 540 | 5,3 | 3312 | PDECOL, General Collocation Software for Partial Differential Equations |
| 541 | 5,3 | 5472 | Efficient Fortran Subprograms for the Solution of Separable Elliptic Partial Differential Equations |
| 542 | 5,4 | 877 | Incomplete Gamma Functions |
| 543 | 5,4 | 1296 | FFT9, Fast Solution of Helmholtz-Type Partial Differential Equations |
| 544 | 5,4 | 2304 | L2A and L2B, Weighted Least Squares Solutions by Modified Gram-Schmidt with Iterative Refinement |
| 545 | 5,4 | 2592 | An Optimized Mass Storage FFT |
| 546 | 6,1 | 540 | SOLVEBLOK |
| 547 | 6,1 | 612 | Fortran Routines for Discrete Cubic Spline Interpolation and Smoothing |
| 548 | 6,1 | 252 | Solution of the Assignment Problem |
| 549 | 6,1 | 360 | Weierstrass Elliptic Functions |
| 550 | 6,1 | 648 | Solid Polyhedron Measures |
| 551 | 6,2 | 828 | A Fortran Subroutine for the $L_1$ Solution of Overdetermined Systems of Linear Equations |
| 552 | 6,2 | 504 | Solution of the Constrained $L_1$ Linear Approximation Problem |
| 553 | 6,2 | 1404 | M3RK, An Explicit Time Integrator for Semidiscrete Parabolic Equations |
| 554 | 6,2 | 612 | BRENTM, A Fortran Subroutine for the Numerical Solution of Systems of Nonlinear Equations |

| | | | |
|---|---|---|---|
| 555 | 6,2 | 1260 | Chow-Yorke Algorithm for Fixed Points or Zeros of $C^2$ Maps |
| 556 | 6,3 | 847 | Exponential Integrals |
| 557 | 6,3 | 792 | PAGP, A Partitioning Algorithm for (Linear) Goal Programming Problems |
| 558 | 6,3 | 720 | A Program for the Multifacility Location Problem with Rectilinear Distance by the Minimum-Cut Approach |
| 559 | 6,3 | 576 | The Stationary Point of a Quadratic Function Subject to Linear Constraints |
| 560 | 6,3 | 3132 | JNF, An Algorithm for Numerical Computation of the Jordan Normal Form of a Complex Matrix |
| 561 | 6,3 | 396 | Fortran Implementation of Heap Programs for Efficient Table Maintenance |
| 562 | 6,3 | 288 | Shortest Path Lengths |
| 563 | 6,4 | 3610 | A Program for Linearly Constrained Discrete $l_1$ Problems |
| 564 | 6,4 | 1042 | A Test Problem Generator for Discrete Linear $L_1$ Approximation Problems |
| 565 | 7,1 | 3159 | PDETWO/PSETM/GEARB: Solution for Systems of Two-Dimensional Nonlinear Partial Differential Equations |
| 566 | 7,1 | 6760 | Fortran Subroutines for Testing Unconstrained Optimization Software |
| 567 | 7,1 | 1114 | Extended Range Arithmetic and Normalized Legendre Polynomials |
| 568 | 3,1 | 16891 | PDS — A Portable Directory System (in ACM Transactions on Programming Languages and Software) |
| 569 | 7,2 | 7725 | COLSYS: Collocation Software for Boundary Value ODE's |
| 570 | 7,2 | 1538 | LOPSI: A Simultaneous Iteration Algorithm for Real Matrices |
| 571 | 7,2 | 397 | Statistics for von Mises' and Fisher's Distribution of Directions $I_1(x)/I_0(x)$, $I_{1.5}(x)/I_{0.5}(x)$ |
| 572 | 7,2 | 2024 | Solution of the Helmholtz Equation for the Dirichlet Problem on General Bounded Three Dimensional Regions |
| 573 | 7,3 | 13080 | NL2SOL — An Adaptive Nonlinear Least-Squares Algorithm |
| 574 | 7,3 | 891 | Shape Preserving Osculatory Quadratic Splines |
| 575 | 7,3 | 262 | Permutations for a Zero-Free Diagonal |
| 576 | 7,3 | 901 | A Fortran Program for Solving Ax = b |
| 577 | 7,3 | 1048 | Algorithms for Incomplete Elliptic Integrals |
| 578 | 7,4 | 515 | Solution of Real Linear Equations in a Paged Virtual Store |
| 579 | 7,4 | 260 | CPSC, Complex Power Series Coefficients |
| 580 | 7,4 | 1558 | QRUP: A Set of Fortran Routines for Updating QR Factorizations |
| 581 | 8,1 | 2945 | An Improved Algorithm for Computing the Singular Value Decomposition |
| 582 | 8,2 | 4835 | The Gibbs-Poole-Stockmeyer and Gibbs-King Algorithms for Reordering Sparse Matrices |
| 583 | 8,2 | 1161 | LSQR: Sparse Linear Equations and Least-Square Problems |
| 584 | 8,2 | 453 | CUBTRI — Automatic Cubature Over a Triangle |
| 585 | 8,3 | 310 | A Subroutine for the General Interpolation and Extrapolation Problems |
| 586 | 8,3 | 11508 | ITPACK 2C: A Fortran Package for Solving Large Sparse Linear Systems by Adaptive Accelerated Iterative Methods |
| 587 | 8,3 | 6654 | Two Algorithms for the Linearly Constrained Least Squares Problem |
| 588 | 8,4 | 2401 | Fast Hankel Transforms Using Related and Lagged Convolutions |
| 589 | 8,4 | 1905 | SICEDR: A Fortran Subroutine for Improving the Accuracy of Computed Matrix Eigenvalues |
| 590 | 8,4 | 2190 | DSUBSP and EXCHQZ: Fortran Subroutines for Computing Deflating Subspaces with Specified Spectrum |
| 591 | 8,4 | 1648 | A Comprehensive Matrix-Free Algorithm for Analysis of Variance |
| 592 | 9,1 | 1395 | A Fortran Subroutine for Computing the Optimal Estimate of $f(x)$ |
| 593 | 9,1 | 5058 | A Package for the Helmholtz Equation in Nonrectangular Planar Regions |
| 594 | 9,1 | 5438 | Software for Relative Error Analysis |
| 595 | 9,1 | 619 | An Enumerative Algorithm for Finding Hamiltonian Circuits in a Directed Graph |
| 596 | 9,2 | 3492 | A Program for a Locally Parametrized Continuation Process |
| 597 | 9,2 | 1365 | Sequence of Modified Bessel Functions of the First Kind |
| 598 | 9,2 | 1254 | An Algorithm to Compute Solvents of the Matrix Equation $AX^2 + BX + C = 0$ |
| 599 | 9,2 | 908 | Sampling from Gamma and Poisson Distributions |
| 600 | 9,2 | 1138 | Translation of Algorithm 507. Procedures for Quintic Natural Spline Interpolation |
| 601 | 9,2 | 1746 | A Sparse Matrix Package — Part II: Special Cases |
| 602 | 9,3 | 1108 | HURRY: An Acceleration Algorithm for Scalar Sequences and Series |
| 603 | 9,3 | 4859 | COLROW and ARCECO: Fortran Packages for Solving Almost Block Diagonal Linear Systems by Modified Alternate Row and Column Elimination |
| 604 | 9,3 | 1025 | A Fortran Program for the Calculation of an Extremal Polynomial |

| | | | |
|---|---|---|---|
| 605 | 9,4 | 5952 | PBASIC — A Verifier Program for ANSI Minimal Basic |
| 606 | 9,4 | 5416 | NITPACK — An Interactive Tree Package |
| 607 | 9,4 | 55560 | Text Exchange System: A Transportable System for Management and Exchange of Programs and Other Text |
| 608 | 9,4 | 511 | Approximate Solution of the Quadratic Assignment Problem |
| 609 | 9,4 | 4863 | A Portable Fortran Subroutine for the Bickley Functions $Ki_n(x)$ |
| 610 | 9,4 | 2910 | A Portable Fortran Subroutine for the Derivation of the Psi Function |
| 611 | 9,4 | 10859 | Subroutines for Unconstrained Minimization Using a Model/Trust-Region Approach |
| 612 | 10,1 | 2944 | TRIEX: Integration Over a TRIangle Using Nonlinear Extrapolation |
| 613 | 10,1 | 223 | Minimum Spanning Tree for Moderate Integer Weights |
| 614 | 10,2 | 1099 | A Fortran Subroutine for Numerical Integration in $H_p$ |
| 615 | 10,2 | 1818 | The Best Subset of Parameters in Least Absolute Value Regression |
| 616 | 10,3 | 441 | Fast Computation of the Hodges-Lehman Location Estimator |
| 617 | 10,3 | 4065 | DAFNE — A Differential Algorithm for Non-Linear Functions |
| 618 | 10,3 | 1589 | Fortran Subroutine for Estimating Sparse Jacobian Matrices |
| 619 | 10,3 | 742 | Automatic Numerical Inversion of the Laplace Transform |
| 620 | 10,4 | 3211 | References and Keywords for Collected Algorithms from ACM |
| 621 | 10,4 | 2025 | Software with Low Storage Requirements for Two-Dimensional, Nonlinear, Parabolic Differential Equations |
| 622 | 10,4 | 15883 | A Simple Macroprocessor |
| 623 | 10,4 | 6496 | Interpolation on the Surface of a Sphere |
| 624 | 10,4 | 5983 | Triangulation and Interpolation at Arbitrarily Distributed Points in the Plane |
| 625 | 10,4 | 4104 | A Two-Dimensional Domain Processor |
| 626 | 10,4 | 1899 | TRICP: A Contour Plot Program for Triangular Meshes |
| 627 | 11,1 | 2199 | A Fortran Subroutine for Solving Volterra Integral Equations |
| 628 | 11,1 | 3040 | An Algorithm for Constructing Canonical Bases of Polynomial Ideals |
| 629 | 11,2 | 5261 | An Integral Equation Program for Laplace's Equation in Three Dimensions |
| 630 | 11,2 | 14092 | BBVSCG — A Variable-Storage Algorithm for Function Minimization |
| 631 | 11,2 | 735 | Finding a Bracketed Zero by Larkin's Method of Rational Interpolation |
| 632 | 11,2 | 769 | A Program for the 0-1 Multiple Knapsack Problem |
| 633 | 11,2 | 3532 | An Algorithm for Linear Dependency Analysis of Multivariate Data |
| 634 | 11,3 | 1687 | CONST and EVAL; Routines for Fitting Multinomials in a Least Squares Sense |
| 635 | 11,3 | 2330 | An Algorithm for the Solution of Systems of Complex Linear Equations in the $L_\infty$ Norm with Constraints on the Unknowns |
| 636 | 11,4 | 2864 | Fortran Subroutines for Estimating Sparse Hessian Matrices |
| 637 | 11,4 | 10764 | GENCOL: Collocation Software on General Domains with Bicubic Hermite Polynomials |
| 638 | 11,4 | 3825 | INTCOL and HERMCOL: Collocation on Rectangular Domains with Bicubic Hermite Polynomials |
| 639 | 12,1 | 1647 | To Integrate Some Infinite Oscillating Tails |
| 640 | 12,1 | 1596 | Efficient Calculation of Frequency Response Matrices from State Space Models |
| 641 | 12,2 | 1658 | Exact Solution of General Integer Systems of Linear Equations |
| 642 | 12,2 | 1092 | A Fast Procedure for Calculating Minimum Cross-Validation Cubic Smoothing Splines |
| 643 | 12,2 | 1168 | FEXACT: A Fortran Subroutine for Fisher's Exact Test on Unordered $r \times c$ Contingency Tables |
| 644 | 12,3 | 20148 | A Portable Package for Bessel Functions of a Complex Argument and Nonnegative Order |
| 645 | 12,3 | 1158 | Subroutine for Testing Programs that Compute the Generalized Inverse of a Matrix |
| 646 | 12,3 | 1683 | PFIND–A Routine to Find a Positive Definite Linear Combination of Two Real Symmetric Matrices |
| 647 | 12,4 | 1062 | Implementation and Relative Efficiency of Quasirandom Sequence Generators |
| 648 | 13,1 | 22844 | NSDTST and STDTST: Fortran Routines for Assessing the Performance of Initial Value Solvers |
| 649 | 13,1 | 7032 | A Package for Computing the Trigonometric Fourier Coefficients Based on Lyness's Algorithm |
| 650 | 13,2 | 427 | Efficient Square Root Implementation on the Motorola 68000 |
| 651 | 13,3 | 18078 | HFFT—High-Order Fast-Direct Solution of the Helmholtz Equation |
| 652 | 13,3 | 13441 | HOMPACK: A Suite of Codes for Globally Convergent Homotopy Algorithms |
| 653 | 13,3 | 4314 | Translation of Algorithm 539: PC-BLAS, Basic Linear Algebra Subprograms for Fortran usage with the INTEL 8087/80287 Numeric Data Processor |
| 654 | 13,3 | 1517 | Fortran Subroutines for Computing the Incomplete Gamma Function Ratios and their Inverse |
| 655 | 13,4 | 3953 | IQPACK: Fortran Subroutines for the Weights of Interpolatory Quadratures |

| | | | |
|---|---|---|---|
| 656 | 14,1 | 31803 | An Extended Set of Basic Linear Algebra Subprograms: Model Implementations and Test Programs |
| 657 | 14,1 | 598 | Plotting Contour Surfaces of a Function of Three Variables |
| 658 | 14,1 | 10760 | ODESSA—An Ordinary Differential Equation Solver with Explicit Simultaneous Sensitivity Analysis |
| 659 | 14,1 | 1002 | Implementing Sobol's Quasirandom Sequence Generator |
| 660 | 14,2 | 1508 | QSHEP2D: Quadratic Shepard Method for Bivariate Interpolation of Scattered Data |
| 661 | 14,2 | 1638 | QSHEP3D: Quadratic Shepard Method for Trivariate Interpolation of Scattered Data |
| 662 | 14,2 | 4537 | A Fortran Software Package for the Numerical Inversion of the Laplace Transform Based on Weeks' Method |
| 663 | 14,2 | 4086 | Translation of Algorithm 539: Basic Linear Algebra Subprograms for Fortran Usag in Fortran 200 for the Cyber 205 |
| 664 | 14,3 | 1536 | A Gauss Algorithm to Solve Systems with Large, Banded Matrices Using Random-Access Disk Storage |

# COULOMB

## General Information

**Description** : The subroutine COULFG computes the regular and irregular Coulomb wave functions F and G, and their derivatives dF/dx and dG/dx. These functions depend on two parameters, $\eta$ and $L$ in the notation of Abramowitz and Stegun, *Handbook of Mathematical Functions*, NBS Applied Mathematics Series 55. A single call to COULFG computes the functions for fixed positive $x$, fixed real $\eta$, and a range of $L$ values in integer steps. Available in source form only.

**Portability** : Portable

**Reference** : A.R. Barnett. COUFLG: Coulomb and Bessel functions and their derivatives, for real arguments, by Steed's method. *Computer Physics Communications*, 27:147-166, 1982.

**Developer** : Department of Physics, The University, Manchester, UK. (A. R. Barnett)

**Distributor** : D. Lozier, NIST, Bldg 101 Room A302, Gaithersburg, MD 20899 (FTS 879-2706 or 301-975-2706)

## 855NOS Information

**Version** : 1982

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,COULOMB.`

**Module Doc** : `INVOKE,GETDOC,COULOMB,CONTENT.`

**Access** : `INVOKE,GETSRC,COULOMB,COULFG.`

# DATAPAC

## General Information

**Description** : A Fortran subroutine library for probability distribution, density, percent point, and sparsity function evaluation; random number generation; line-printer plotting — histograms, scatter diagrams, probability plots; data manipulation; general statistical analysis; time series analysis; polynomial regression; ANOVA. (Approximately 170 subroutines.)

**Portability** : Portable, some conversion required

**Reference** : J.J. Filliben. *User's Guide to Datapac* (version 77.5). NBS, Gaithersburg, MD, 1977.

**Developer** : NIST, Statistical Engineering Division, Gaithersburg, MD 20899 (J.J. Filliben)

**Distributor** : S. Bremer, NIST, Bldg 101 Room A337, Gaithersburg, MD 20899 (FTS 879-2845 or 301-975-2845)

## 855NOS Information

**Version** : 1986

**Support** : Full

**General Doc** : `INVOKE,GETDOC,DATAPAC.`

**Module Doc** : `INVOKE,GETDOC,DATAPAC,`<*module*>`.`

**Access** : `INVOKE,GETLIB,DATAPAC.`
`LIBRARY,DATAPAC.`

## 840NOS Information

**Version** : 1986

**Support** : Full

**Access** : `ATTACH,DATAPAC/UN=CAMLIB,NA.`
`LIBRARY,DATAPAC.`

## Itemized Contents

**D1 : Elementary vector and matrix operations**

COPY DEFINE MAX MIN MOVE

**L1 : Data summarization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| CORR | LOC | MIDM | RANGE | SCALE | SPCORR | STMOM4 | VAR |
| COUNT | MEAN | MIDR | RELSD | SD | STMOM3 | TRIM | WIND |
| FREQ | MEDIAN | PROPOR | SAMPP | | | | |

**L2 : Data manipulation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| DELETE | DISCR3 | DISCRE | REPLAC | RETAIN | SUBSE1 | SUBSE2 | SUBSET |
| DISCR2 | | | | | | | |

**L3 : Elementary statistical graphics**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| HIST | PLOT6 | PLOT9 | PLOTS | PLOTST | PLOTU | PLOTXT | PLTSCT |
| PLOT | PLOT7 | PLOTC | PLOTSC | PLOTT | PLOTX | PLOTXX | PLTXXT |
| PLOT10 | PLOT8 | PLOTCT | | | | | |

**L4 : Elementary data analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| CAUPLT | EV2PLT | GAMPLT | LAMPLT | NOROUT | PLOTU | TAIL | UNIPLT |
| CHSPLT | EXPPLT | GEOPLT | LGNPLT | NORPLT | POIPLT | TOL | WEIB |
| DEXPLT | EXTREM | HFNPLT | LOGPLT | PARPLT | RUNS | TPLT | WEIPLT |
| EV1PLT | | | | | | | |

**L5 : Statistical function evaluation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BINCDF | CHSPPF | EV2CDF | GAMCDF | LAMPDF | LOGPPF | NORSF | UNICDF |
| BINPPF | DEXCDF | EV2PPF | GAMPPF | LAMPPF | LOGSF | PARCDF | UNIPDF |
| CAUCDF | DEXPDF | EXPCDF | GEOCDF | LAMSF | NBCDF | PARPPF | UNIPPF |
| CAUPDF | DEXPPF | EXPPDF | GEOPPF | LGNCDF | NBPPF | POICDF | UNISF |
| CAUPPF | DEXSF | EXPPPF | HFNCDF | LGNPPF | NORCDF | POIPPF | WEICDF |
| CAUSF | EV1CDF | EXPSF | HFNPPF | LOGCDF | NORPDF | TCDF | WEIPPF |
| CHSCDF | EV1PPF | FCDF | LAMCDF | LOGPDF | NORPPF | TPPF | |

**L6 : Random number generation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BETRAN | CHSRAN | EV2RAN | GAMRAN | LAMRAN | NBRAN | POIRAN | UNIRAN |
| BINRAN | DEXRAN | EXPRAN | GEORAN | LGNRAN | NORRAN | RANPER | WEIRAN |
| CAURAN | EV1RAN | FRAN | HFNRAN | LOGRAN | PARRAN | TRAN | |

**L8 : Regression**

POLY

**L10 : Time series analysis**

| | | | |
|---|---|---|---|
| AUTOCO | DEMOD | FOURIE | TIME |

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| CODE | MIN | READ | SKIPR | SORT | SORTC | SORTP | WRITE |
| MAX | RANK | READG | | | | | |

# DATAPLOT

## General Information

**Description** : A program for graphics (2-d, 3-d, color, plotting data and functions, special fonts and symbols, diagrams, histograms, probability plots, box plots, time series plots), fitting (non-linear, multi-linear, polynomial, spline), and probability and statistics (summary statistics, pdf's, cdf's, random numbers, analysis of variance, smoothing). Mathematics routines (roots, functions, differentiation, integration, convolution) are not catalogued in GAMS. (Approximately 215 commands.)

**Portability** : Portable, some conversion required

**Reference** : J.J. Filliben. *DATAPLOT – Introduction and Overview*, NBS SP 667. Gaithersburg, MD, 1984.

**Developer** : NIST, Statistical Engineering Division, Gaithersburg, MD 20899 (J.J. Filliben)

**Distributor** : NTIS, U. S. Department of Commerce, 5285 Port Royal Road, Springfield, VA 22161 (Walter Finch, 703-487-4805)

## 855NOS Information

**Version** : 1985

**Support** : Full

**General Doc** : `INVOKE,GETDOC,DATAPLT.`

**Module Doc** : `HELP` <*command*> (in DATAPLOT)

**Access** : `INVOKE,DATAPLT.`

## 855VE Information

**Version** : 1989

**Support** : Full

**General Doc** : `FETCH DATAPLOT USAGE`

**Module Doc** : `FETCH DATAPLOT` <*first letter of command*> `DOC`
then locate command

**Access** : `DATAPLOT` (interactively), or
`DATAPLOT` I=<*optional input command file*>
O=<*optional output file*>

## 205 Information

**Version** : 1985

**Support** : Full

**General Doc** : `INVOKE,GETDOC,DPLT205.` (Under 855NOS)

**Module Doc** : `HELP` <*command*> (in DATAPLOT under 855NOS)

**Access** : `INVOKE,DPLT205?` (Under 855NOS)

## CAMVAX Information

**Version** : 1985

**Support** : Full

**Module Doc** : `HELP` <*command*> (in DATAPLOT)

**Access** : `DATAPLOT`

## Itemized Contents

**E1 : Interpolation of univariate data (curve fitting)**

EXACT RATIONAL FIT

**L1 : Data summarization**

SUMMARY

**L2 : Data manipulation**

LET
LET FUNCTION
TABULATE

**L3 : Elementary statistical graphics**

3D-PLOT
4-PLOT
BIHISTOGRAM
BOX PLOT
ERROR BAR PLOT
FREQUENCY PLOT
HISTOGRAM
HOMOSCEDASTICITY PLOT
I PLOT
LAG PLOT
MULTIPLOT
PARETO PLOT
PERCENT POINT PLOT
PIE CHART
PLOT
PROFILE PLOT
RUN SEQUENCE PLOT
STAR PLOT
STATISTICS PLOT
STEM AND LEAF DIAGRAM
SYMMETRY PLOT
YOUDEN PLOT

**L4 : Elementary data analysis**

4-PLOT
BETA PROBABILITY PLOT
BINOMIAL PROBABILITY PLOT
BOOTSTRAP PLOT
BOX-COX LINEARITY PLOT
BOX-COX NORMALITY PLOT
CAUCHY PROBABILITY PLOT
CHI-SQUARED PROBABILITY PLOT
CHI-SQUARED PPCC PLOT
CONFIDENCE LIMITS
CONTROL CHART
DOUBLE EXPONENTIAL PROBABILITY PLOT
EXPONENTIAL PROBABILITY PLOT
EXTREME VALUE TYPE 2 PROBABILITY PLOT
EXTREME VALUE TYPE 1 PROBABILITY PLOT
EXTREME VALUE TYPE 2 PPCC PLOT
F PROBABILITY PLOT
GAMMA PPCC PLOT
GAMMA PROBABILITY PLOT
GEOMETRIC PPCC PLOT
GEOMETRIC PROBABILITY PLOT
HALFNORMAL PROBABILITY PLOT
JACKNIFE PLOT
LAMBDA PPCC PLOT
LAMBDA PROBABILITY PLOT
LOGISTIC PROBABILITY PLOT
LOGNORMAL PROBABILITY PLOT
NEGATIVE BINOMIAL PROBABILITY PLOT
NORMAL PROBABILITY PLOT
PARETO PPCC PLOT
PARETO PROBABILITY PLOT
POISSON PPCC PLOT
POISSON PROBABILITY PLOT
QUANTILE-QUANTILE PLOT
ROOTOGRAM
RUNS
SEMI-CIRCULAR PROBABILITY PLOT
T PPCC PLOT
T PROBABILITY PLOT
T-TEST
TRIANGULAR PROBABILITY PLOT
UNIFORM PROBABILITY PLOT
WEIBULL PLOT
WEIBULL PPCC PLOT
WEIBULL PROBABILITY PLOT

**L7 : Analysis of variance**

ANOVA
MEDIAN POLISH
YATES ANALYSIS

**L8 : Regression**

FIT
LOWESS SMOOTH
PRE-FIT
SMOOTH
SPLINE FIT

**L9 : Categorical data analysis**

MEDIAN POLISH

**L10 : Time series analysis**

ALLAN STANDARD DEVIATION PLOT
ALLAN VARIANCE PLOT
COMPLEX DEMODULATION PLOT
CONTROL CHART
CORRELATION PLOT
PERIODOGRAM
SMOOTH
SPECTRUM

**Q : Graphics**

CONTOUR PLOT
FRACTAL PLOT

# DERIV

## General Information

**Description** : The subroutine DIFF computes the first, second, or third derivative of a real function of a single real variable, where the function is defined by a Fortran function subprogram. The method used is Neville's process of extrapolating from a sequence of interpolating polynomials with interpolating points distributed symmetrically about $x_0$ or, if this is not possible, to one side of $x_0$. The package includes three tests. Available in source form only.

**Portability** : Portable

**Reference** : J. Oliver. An Algorithm for Numerical Differentiation of a Function of One Real Variable. *Journal of Computational and Applied Mathematics*, 6(2):145-160,1980.

**Developer** : University of Essex, Department of Computer Science, Colchester, England (J. Oliver)

**Distributor** : D. Lozier, NIST, Bldg 101 Room A302, Gaithersburg, MD 20899 (FTS 879-2706 or 301-975-2706)

## 855NOS Information

**Version** : 1980

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,DERIV.`

**Module Doc** : `INVOKE,GETDOC,DERIV,CONTENT.`

**Access** : `INVOKE,GETSRC,DERIV,DIFF.`

# DISSPLA

## General Information

**Description** : A Fortran subroutine library for producing publication-quality plots of two- and three-dimensional data. Capabilities include drawing of axes, grids, and labels, curve drawing, area shading, interpolation and smoothing, multiple plots per page, surface view plotting, contouring, scaling, projection and rotation, and elaborate character fonts. Many output devices are supported, as well as device-independent output.

**Portability** : Proprietary

**Reference** : *DISSPLA User's Manual*, Version 10.0. ISSCO Graphics Software, San Diego, CA, 1986.

**Developer** : ISSCO Graphics Software, 10505 Sorronto Valley Road, San Diego, CA 92121

**Distributor** : ISSCO Graphics Software, 10505 Sorronto Valley Road, San Diego, CA 92121

## 855NOS Information

**Version** : 10.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,DISSPLA.`

**Module Doc** : `INVOKE,GETDOC,DISSPLA,CONTENT.`

**Access** : `INVOKE,DISSPLA?`

## 855VE Information

**Version** : 10.5

**Support** : Full

**General Doc** : `FETCH DISSPLA USAGE`

**Module Doc** : `FETCH DISSPLA CONTENTS`

**Access** : `DISSPLA PROGRAM=`<*file*>

# 840NOS Information

**Version** : 10.0

**Support** : Full

**General Doc** : GET,GRAFING/UN=GRAF.
GRAFING.

**Module Doc** : GET,GRAFING/UN=GRAF.
GRAFING.

**Access** : ATTACH,DISP100/UN=GRAF.
LIBRARY,DISP100/A.

# ELLPACK

## General Information

**Description** : A software system for solving linear elliptic partial differential equations in general two-dimensional domains and in three-dimensional boxes with a variety of boundary conditions. Users write programs in the ELLPACK language (Fortran extension) which allows one to declare boundary value problems and to select from a large library of modules to solve them numerically. Results can be printed, plotted, and are also available as Fortran-callable functions for further processing.

**Portability** : Proprietary

**Reference** : J.R. Rice and R.F. Boisvert. *Solving Elliptic Problems Using ELLPACK.* Springer-Verlag, New York, 1985.

**Developer** : Purdue University, Computer Science Department, West Lafayette, IN 47907 (J.R. Rice and others)

**Distributor** : ELLPACK, Computer Science Department, Purdue University, West Lafayette, IN 47907 (317-494-6003)

## 855NOS Information

**Version** : 1984

**Support** : Full

**General Doc** : `INVOKE,GETDOC,ELLPACK.`

**Module Doc** : `INVOKE,GETDOC,ELLPACK,CONTENT.`

**Access** : `INVOKE,ELLPACK?`

## Itemized Contents

ELLPACK has two main components: a high-level input language and a large collection of problem-solving modules. Here we briefly give examples of some statements from the ELLPACK language and list the available problem-solving modules. A complete description of the ELLPACK language and modules is contained in

J. R. Rice and R. F. Boisvert, *Solving Elliptic Problems with ELLPACK*, Springer-Verlag, New York, 1985.

Each ELLPACK program must contain exactly one EQUATION statement, one BOUNDARY statement, and one END statement. Some examples of the first two follow (the first boundary statement form is valid only for rectangular domains).

```
EQUATION.   UXX + UYY + UZZ  =  F(X,Y,Z)

EQUATION.   UXX + 2.0*UYY + D(X,Y)*UY - U  =  SIN(X)*EXP(Y)
```

```
BOUNDARY.  PERIODIC            ON  X =  0.0
                               ON  X =  1.0
           U = 0.0             ON  Y = -1.0
           UY + 2.0*U = G(X)   ON  Y =  1.0

BOUNDARY.  UX = 0.0                   ON  X=PI, Y=T  FOR T=-1.0 TO 5.0
           U  = 0.0                   ON  LINE  PI,5.0   TO  0.0,5.0
           UX = 0.0                   ON  LINE  0.0,5.0  TO  0.0,1.0
           SIN(X)*UX + UY = -SINH(Y)  ON  X=T, Y=COS(T) FOR T=0.0 TO PI
```

The GRID statement is used to impose a rectangular grid on the domain. Several examples follow (the first is only valid for rectangular domains).

```
GRID.   11 X POINTS  $  17 Y POINTS

GRID.   11 X POINTS   0.0 TO 1.0
         9 Y POINTS   0.0, 0.1, 0.15, 0.2, 0.3, 0.5, 0.7, 0.8, 1.0
```

The following is an example of an ELLPACK program.

```
*        ***************************************************
*        *                                                 *
*        *  EXAMPLE ELLPACK PROGRAM                        *
*        *                                                 *
*        *     THIS IS EXAMPLE 1.E1 FROM THE ELLPACK       *
*        *     USER'S GUIDE.                               *
*        *                                                 *
*        ***************************************************

OPTIONS.        TIME  $  MEMORY

EQUATION.       UXX + UYY + 3.0*UX - 4.0*U  =  EXP(X+Y)*SIN(PI*X)

BOUNDARY.       U = 0.0                ON  X = 0.0
                U = SIN(PI*X) - X/2.0  ON  Y =-1.0
                U = Y/2.0              ON  X = 1.0
                U = X                  ON  Y = 2.0

GRID.           6 X POINTS
                6 Y POINTS

DISCRETIZATION. 5 POINT STAR
SOLUTION.       LINPACK BAND

OUTPUT.         TABLE (U)

END.
```

The following table lists the problem-solving modules in ELLPACK.

# ELLPACK Problem-Solving Modules

| | | |
|---|---|---|
| **Discretization Modules** | | |
| *Discretize the boundary-value problem, generating system of linear equations.* | | |
| *Two-dimensions, rectangular domain:* | | |
| 5-POINT STAR | HODIE-ACF | SPLINE GALERKIN |
| HERMITE COLLOCATION | HODIE-HELMHOLTZ | |
| HODIE | INTERIOR COLLOCATION | |
| *Two-dimensions, general domain:* | | |
| 5-POINT STAR | COLLOCATION | |
| *Three-dimensions, rectangular domain:* | | |
| 7-POINT 3D | | |
| **Indexing Modules** | | |
| *Reorder rows and columns of linear system generated in discretization phase.* | | |
| AS IS | MINIMUM DEGREE | REVERSE CUTHILL MCKEE |
| HERMITE COLLORDER | NESTED DISSECTION | |
| INTERIOR COLLORDER | RED-BLACK | |
| **Solution Modules** | | |
| *Solve system of algebraic equations generated in discretization phase.* | | |
| *Direct Methods:* | | |
| BAND GE | ENVELOPE LDU | SPARSE GE NO PIVOTING |
| BAND GE NO PIVOTING | LINPACK BAND | SPARSE LU UNCOMPRESSED |
| ENVELOPE LDLT | LINPACK SPD BAND | |
| *Iterative Methods:* | | |
| JACOBI CG | REDUCED SYSTEM SI | SYMMETRIC SOR SI |
| JACOBI SI | SOR | |
| REDUCED SYSTEM CG | SYMMETRIC SOR CG | |
| **Triple Modules** | | |
| *Discretization and solution in one step.* | | |
| *Two-dimensions, rectangular domain:* | | |
| DYAKANOV-CG | HODIE-FFT | SET U BY BICUBICS |
| DYAKANOV-CG 4 | MARCHING ALGORITHM | SET U BY BLENDING |
| FFT 9-POINT | MULTIGRID MG00 | |
| FISHPAK-HELMHOLTZ | SET | |
| *Two-dimensions, general domain:* | | |
| P2C0-TRIANGLES | | |
| *Three-dimensions, rectangular domain:* | | |
| HODIE 27-POINT 3D | HODIE FFT 3D | |
| **Output Modules** | | |
| *Produce printed or plotted output.* | | |
| *Print/plot information about problem or solution:* | | |
| MAX | SUMMARY | TABLE |
| NORM | PLOT | |
| RMS | PLOT-DOMAIN | |
| *Print internal ELLPACK data:* | | |
| DATA | TABLE-INDEXES | TABLE-DOMAIN |
| TABLE-BOUNDARY | TABLE-PROBLEM | |
| TABLE-EQUATIONS | TABLE-UNKNOWN | |
| **Procedure Modules** | | |
| *Perform various utility tasks.* | | |
| DISPLAY MATRIX PATTERN | NON-UNIQUE | REMOVE BICUBIC BC |
| DOMAIN FILL | PLOT COLLOCATION POINTS | SET UNKNOWNS FOR 5-POINT STAR |
| EIGENVALUES | REMOVE | SET UNKNOWNS FOR HODIE-HELMHOLTZ |
| LIST MODULES | REMOVE BLENDED BC | |

# IMSL

## General Information

**Description** : A Fortran subprogram library for solving standard problems in many areas of mathematics and statistics. (Approximately 500 subprograms.)

**Portability** : Proprietary

**Reference** : *IMSL Reference Manual*, vol. 1-4, (QA297.I19 in NIST Gaithersburg Research Information Center, Admin E-120)

**Developer** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, TX 77042-3020

**Distributor** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, TX 77042-3020 (713-782-6060)

## 855NOS Information

**Version** : 9.2

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,IMSL.`

**Module Doc** : `INVOKE,GETDOC,IMSL,`<*module*>`.`

**Access** : `INVOKE,GETLIB,IMSL.`
`LIBRARY,IMSL.`

## 855VE Information

**Version** : 9.2

**Support** : Limited

**General Doc** : `FETCH (IMSL PREVIOUS) USAGE`

**Module Doc** : `INVOKE,GETDOC,IMSL,`<*module*>`.` (Under 855NOS)

**Access** : `CREACLE A=IMSL V=VER_9_2`
`IMSL AL=YES`

## 205 Information

| | |
|---|---|
| **Version** | : 9.2 |
| **Support** | : Limited |
| **General Doc** | : `INVOKE,GETDOC,IMSL.` (Under 855NOS) |
| **Module Doc** | : `INVOKE,GETDOC,IMSL,<module>.` (Under 855NOS) |
| **Access** | : `PATTACH,MATHPOOL.` `LOAD,BINARY,LIB=IMSL.` |

## 840NOS Information

| | |
|---|---|
| **Version** | : 9.2 |
| **Support** | : Limited |
| **Access** | : `ATTACH,IMSL/UN=LIB5,NA.` `LIBRARY,IMSL.` |

## Itemized Contents

**B : Number theory**

VDCPS

**C : Elementary and special functions**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ALGAMA | MERFCI | MMBSIN | MMBSJR | MMBZIN | MMDELK | MMLINC | MMWPL |
| ERF | MERFI | MMBSIR | MMBSK0 | MMBZJN | MMDEN | MMLIND | MMWPL1 |
| ERFC | MERRCZ | MMBSJ0 | MMBSK1 | MMDAS | MMKEL0 | MMLINF | MMWPQ |
| GAMMA | MMBSI0 | MMBSJ1 | MMBSKR | MMDEI | MMKEL1 | MMLINJ | MMWPQ1 |
| MDNOR | MMBSI1 | MMBSJN | MMBSYN | MMDELE | MMKELD | MMPSI | RLPOL |

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| USMNMX | VCVTCH | VCVTSF | VIPRSS | VMULFP | VMULSB | VNRMS1 | VUABQ |
| VABMXF | VCVTFB | VCVTSQ | VMULBB | VMULFQ | VMULSF | VNRMS2 | VUAFB |
| VABMXS | VCVTFQ | VHS12 | VMULBF | VMULFS | VMULSQ | VPOLYF | VUAFQ |
| VABSMF | VCVTFS | VHSH2C | VMULBS | VMULQB | VMULSS | VTPROF | VUAFS |
| VABSMS | VCVTHC | VHSH2R | VMULFB | VMULQF | VNRMF1 | VTPROS | VUASB |
| VCONVO | VCVTQF | VHSH3R | VMULFF | VMULQQ | VNRMF2 | VTRAN | VUASQ |
| VCVTBF | VCVTQS | VIPRFF | VMULFM | VMULQS | VNRMFI | | |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| LEQ1PB | LEQ2S | LEQT1C | LEQT2F | LINV1F | LINV3F | LUDECP | LUREFF |
| LEQ1S | LEQIF | LEQT1F | LEQT2P | LINV1P | LINV3P | LUELMF | LUREFP |
| LEQ2C | LEQOF | LEQT1P | LIN1PB | LINV2F | LUDAPB | LUELMP | LUREPB |
| LEQ2PB | LEQT1B | LEQT2B | LIN2PB | LINV2P | LUDATF | LUELPB | |

**D3 : Determinants**

LINV3F

**D4 : Eigenvalues, eigenvectors of matrices**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| EBALAC | EHESSF | EHOUSS | EIGCC | EIGRF | EIGZC | EIGZS | EQRT2S |
| EBALAF | EHOUSH | EIGBS | EIGCH | EIGRS | EIGZF | EQRT1S | EQRT3S |
| EHESSC | | | | | | | |

**D6 : Singular value decomposition (SVD)**

LSVDB LSVDF

**D9 : Singular, overdetermined or underdetermined systems of linear equations, generalized inverses**

LGINF LLBQF LLSQF OFIMA3

**E1 : Interpolation of univariate data (curve fitting)**

ICSCCU ICSICU ICSPLN IQHSCU

**E2 : Interpolation of multivariate data (surface fitting)**

IBCCCU IBCIEU IQHSCV

**F : Solution of nonlinear equations**

ZANLYT ZCPOLY ZPOLR ZQADR ZREAL2 ZSCNT ZSPOW ZSRCH
ZBRENT ZFALSE ZQADC ZREAL1 ZRPOLY

**G : Optimization**

ZSRCH ZX3LP ZXCGR ZXGSN ZXGSP ZXLSF ZXMIN ZXMWD
ZX0LP ZX4LP

**H : Differentiation, integration**

DBCQDU DBLIN DCADRE DCSQDU DMLIN DRVTE

**I1 : Ordinary differential equations (ODE's)**

DGEAR DREBS DTPTB DVCPR DVERK

**I2 : Partial differential equations**

DPDES

**J : Integral transforms**

FFT2C FFT3D FFTCC FFTRC FFTSC FLINV VCONVO

**K : Approximation**

DBCEVL DCSQDU ICSEVU ICSMOU ICSSCV IFLSQ RLLAV ZXSSQ
DCSEVU IBCEVL ICSFKU ICSSCU ICSVKU IRATCU RLLMV

**L1 : Data summarization**

BDLTV BECORI BECOVW BECVLI BEIGRP BEMMI BESTAT NMTIE
BECOR BECOVM BECVL BEGRPS BEIUGR BEMMO

**L2 : Data manipulation**

BDCOU1 BDTAB BDTRGO FTRDIF GTPL GTRTN RLGQMI RLPOL
BDCOU2 BDTRGI BDTWT GTDDU GTPR GTTRT RLGQMO

**L3 : Elementary statistical graphics**

USBOX USHHST USHST USHST2 USPDF USSLF

**L4 : Elementary data analysis**

BECTR BEPET GTCN GTRN NBSIGN NMCC OIND SSSAND
BELBIN BESRB GTD2T GTTT NDKER NMKN OTMLNR SSSBLK
BELPOS BESRN GTMNT NAK1 NDMPLE NMKTS SSPAND SSSCAN
BEMNON BESTA2 GTNOR NBCYC NHINC NRBHA SSPBLK SSSEST
BEMSON CBNRHO GTPOK NBQT NKS1 NRWMD SSRAND USPC
BENSON GFIT GTPST NBSDL NKS2 NRWRST SSRBLK USPRP
BEPAT

**L5 : Statistical function evaluation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ERF | MDBETI | MDCHI | MDFI | MDHYP | MDSTI | MDTPS | MSMRAT |
| ERFC | MDBIN | MDCHN | MDGAM | MDNOR | MDTD | MERFCI | NDEST |
| GTPKP | MDBNOR | MDFD | MDGC | MDNRIS | MDTN | MERFI | NMKSF |
| MDBETA | MDCH | MDFDRE | MDGCI | MDSMR | MDTNF | MSENO | |

**L6 : Random number generation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| FTGEN | GGCAY | GGEOT | GGNLG | GGNQF | GGSPH | GGUBFS | GGUW |
| GGAMR | GGCHS | GGEXN | GGNML | GGNSM | GGSRS | GGUBS | GGVCR |
| GGBN | GGCOR | GGEXT | GGNO | GGPER | GGSTA | GGUBT | GGVMS |
| GGBNR | GGDA | GGHPR | GGNPM | GGPON | GGTAB | GGUD | GGWIB |
| GGBTR | GGDT | GGMTN | GGNPP | GGPOS | GGTRA | GGUO | |

**L7 : Analysis of variance**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ABIBN | AFACN | AGLMOD | ALSQAN | ANESTE | ARCBAN | NAFRE | NAWRPE |
| ACRDAN | AFACT | AGVACL | AMEANS | ANESTU | ASNKMC | NAK1 | NAWRPU |
| ACTRST | AGBACP | AGXPM | ANCOV1 | AORDR | BEMDP | NAWNRP | |

**L8 : Regression**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BEMIRI | RLDCQM | RLEAP | RLFOTH | RLLAV | RLOPDC | RLRES | RLSUM |
| BEMIRO | RLDCVA | RLFITI | RLFOTW | RLLMV | RLPOL | RLSEP | RSMITZ |
| OFIMA3 | RLDCW | RLFITO | RLINCF | RLMUL | RLPRDI | RLSTP | USLEAP |
| RLCOMP | RLDOPM | RLFOR | RLINPF | RLONE | RLPRDO | RLSUBM | ZXSSQ |

**L9 : Categorical data analysis**

| | | | | | | |
|---|---|---|---|---|---|---|
| BDTWT | BEMDP | CBNRHO | CTLLF | CTPR | CTRBYC | NHEXT |

**L10 : Time series analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| FTARPS | FTCAST | FTCROS | FTFPS | FTKALM | FTML | FTTR | FTWENM |
| FTAUTO | FTCP | FTCRXY | FTFREQ | FTMA | FTRDIF | FTWEIN | FTWENX |

**L12 : Discriminant analysis**

| | |
|---|---|
| ODFISH | ODNORM |

**L13 : Covariance structure models**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| OFCOEF | OFHARR | OFPRI | OFRESI | OFROTA | OFSCHN | OFSCOR | OPRINC |
| OFCOMM | OFIMAG | OFPROT | | | | | |

**L14 : Cluster analysis**

| | | |
|---|---|---|
| OCDIS | OCLINK | USTREE |

**L15 : Life testing and survival analysis**

CLIFE

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| GTPBC | USMNMX | USWBS | USWFM | VSAR | VSORA | VSRTM | VSRTR |
| NMRANK | USTREE | USWCH | USWFV | VSODA | VSRTA | VSRTP | VSRTU |
| USCWV | USWBM | USWCM | USWSM | | | | |

**Q : Graphics**

USPLO

**R : Service routines**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| UERSET | UERTST | UGETIO | UHELP | UHELP1 | UHELP2 | UHELP3 | UHELP4 |

# IMSL MATH/LIBRARY

## General Information

**Description** : A Fortran subroutine library for solving problems in applied mathematics. (Approximately 400 subroutines and functions.)

**Portability** : Proprietary

**Reference** : *IMSL MATH/LIBRARY User's Manual*, Vol 1-3. (QA76.73 in NIST Gaithersburg Research Information Center, Admin E-120) IMSL Inc., Houston, TX, 1987.

**Developer** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, Texas 77042-3020

**Distributor** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, Texas 77042-3020 (713-782-6060)

## 855NOS Information

**Version** : 1.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,IMSLM.`

**Module Doc** : `INVOKE,GETDOC,IMSLM,`<*module*>`.`

**Access** : `INVOKE,GETLIB,IMSLM.`
`LIBRARY,IMSLM.`

## 855VE Information

**Version** : 1.0

**Support** : Full

**General Doc** : `FETCH IMSL USAGE`

**Module Doc** : `FETCH IMSL` <*module*> `DOC`

**Access** : `IMSL AL=YES`

## 205 Information

**Version** : 1.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,IMSLM.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,IMSLM,`<*module*>`.` (Under 855NOS)

**Access** : `PATTACH,MATHPOOL.`
`LOAD,BINARY,LIB=IMSLM.`

## 840NOS Information

**Version** : 1.0

**Support** : Limited

**Access** : `ATTACH,IMSLS/UN=LIB5.`
`LIBRARY,IMSLS.`

## Itemized Contents

**A : Arithmetic, error analysis**

DQADD DQINI DQMUL DQSTO

**B : Number theory**

PRIME

**C : Elementary and special functions**

CONST CUNIT GAMMA HYPOT

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACBCB | CHFCG | CSROTM | DISL2 | ISMAX | MXYTF | SCOPY | SROTM |
| ARBRB | CRBCB | CSSCAL | DISLI | ISMIN | NR1CB | SDDOTA | SROTMG |
| BLINF | CRBRB | CSUB | DSDOT | ISUB | NR1RB | SDDOTI | SSCAL |
| CADD | CRBRG | CSVCAL | HRRRR | ISUM | NR1RR | SDOT | SSET |
| CAXPY | CRGCG | CSWAP | IADD | ISWAP | NR2RR | SDSDOT | SSUB |
| CCBCB | CRGRB | CVCAL | ICAMAX | MCRCR | NRIRR | SHOUAP | SSUM |
| CCBCG | CRGRG | CZCDOT | ICAMIN | MRRRR | POLRG | SHOUTR | SSWAP |
| CCGCB | CRRCR | CZDOTA | ICOPY | MUCBV | SADD | SHPROD | SVCAL |
| CCGCG | CSBRB | CZDOTC | IIMAX | MUCRV | SASUM | SNRM2 | SXYZ |
| CCOPY | CSCAL | CZDOTI | IIMIN | MURBV | SAXPY | SPRDCT | TRNRR |
| CDOTC | CSET | CZDOTU | ISAMAX | MURRV | SCASUM | SROT | VCONC |
| CDOTU | CSFRG | CZUDOT | ISAMIN | MXTXF | SCNRM2 | SROTG | VCONR |
| CHBCB | CSROT | DISL1 | ISET | MXTYF | | | |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| JCGRC | LFCRB | LFIHF | LFSHF | LFTHF | LINRT | LSARG | LSLQH |
| LCHRG | LFCRG | LFIQH | LFSQH | LFTQH | LSACB | LSASF | LSLQS |
| LFCCB | LFCRT | LFIQS | LFSQS | LFTQS | LSACG | LSLCB | LSLRB |
| LFCCG | LFCSF | LFIRB | LFSRB | LFTRB | LSADH | LSLCC | LSLRG |
| LFCCT | LFDHF | LFIRG | LFSRG | LFTRG | LSADS | LSLCG | LSLRT |
| LFCDH | LFDQS | LFISF | LFSSF | LFTSF | LSAHF | LSLCT | LSLSF |
| LFCDS | LFICB | LFSCB | LFTCB | LINCG | LSAQH | LSLDH | LSLTC |
| LFCHF | LFICG | LFSCG | LFTCG | LINCT | LSAQS | LSLDS | LSLTO |
| LFCQH | LFIDH | LFSDH | LFTDH | LINDS | LSARB | LSLHF | PCGRC |
| LFCQS | LFIDS | LFSDS | LFTDS | LINRG | | | |

**D3 : Determinants**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| LFDCB | LFDCT | LFDDS | LFDQH | LFDRB | LFDRG | LFDRT | LFDSF |
| LFDCG | LFDDH | | | | | | |

**D4 : Eigenvalues, eigenvectors of matrices**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| EPICG | EVAHF | EVBSF | EVCRH | EVESF | EVLCH | EVLSF | GVCRG |
| EPIHF | EVASB | EVCCG | EVCSB | EVFHF | EVLHF | GPICG | GVCSP |
| EPIRG | EVASF | EVCCH | EVCSF | EVFSB | EVLRG | GPIRG | GVLCG |
| EPISB | EVBHF | EVCHF | EVEHF | EVFSF | EVLRH | GPISP | GVLRG |
| EPISF | EVBSB | EVCRG | EVESB | EVLCG | EVLSB | GVCCG | GVLSP |

**D5 : QR decomposition, Gram-Schmidt orthogonalization**

LQERR LQRRR

**D6 : Singular value decomposition (SVD)**

LSVCR LSVRR

**D7 : Update matrix decompositions**

LDNCH LUPCH LUPQR

**D9 : Singular, overdetermined or underdetermined systems of linear equations, generalized inverses**

LQRSL LSBRR LSGRR LSQRR

**E1 : Interpolation of univariate data (curve fitting)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BSINT | CSCON | CSDEC | CSHER | CSINT | CSPER | QDDER | QDVAL |
| CSAKM | | | | | | | |

**E2 : Interpolation of multivariate data (surface fitting)**

BS2IN BS3IN QD2DR QD2VL QD3DR QD3VL SURF

**F : Solution of nonlinear equations**

NEQNF NEQNJ ZANLY ZBREN ZPLRC ZPOCC ZPORC ZREAL

**G : Optimization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BCOAH | CDGRD | DLPRS | GDHES | QPROG | UMIDH | UMPOL | UVMGS |
| BCODH | CHGRD | FDGRD | GGUES | UMCGF | UMINF | UNLSF | UVMID |
| BCONG | CHHES | FDHES | NCONF | UMCGG | UMING | UNLSJ | UVMIF |
| BCPOL | CHJAC | FDJAC | NCONG | UMIAH | | | |

**H : Differentiation, integration**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BS2IG | DERIV | GQRUL | QDAGI | QDAWC | QDAWO | QDNG | RECQR |
| BS3IG | FQRUL | QAND | QDAGP | QDAWF | QDAWS | RECCF | TWODQ |
| BSITG | GQRCF | QDAG | QDAGS | | | | |

**I1 : Ordinary differential equations (ODE's)**

| | | | | |
|---|---|---|---|---|
| BVPFD | BVPMS | IVPAG | IVPBS | IVPRK |

**I2 : Partial differential equations**

| | | |
|---|---|---|
| FPS2H | FPS3H | MOLCH |

**J : Integral transforms**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| FCOST | FFT2D | FFTCF | FFTRF | INLAP | QCOSF | QSINB | QSINF |
| FFT2B | FFTCB | FFTRB | FSINT | QCOSB | | | |

**K : Approximation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BCLSF | BSCPP | BSLSQ | CSDER | CSSED | FNLSQ | PPVAL | UNLSF |
| BCLSJ | BSDER | BSVAL | CSITG | CSSMH | PPDER | RATCH | UNLSJ |
| BCONF | BSLS2 | BSVLS | CSSCV | CSVAL | PPITG | RLINE | |

**L6 : Random number generation**

| | | | | |
|---|---|---|---|---|
| RNGET | RNOPT | RNSET | RNUN | RNUNF |

**L8 : Regression**

RLINE

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACHAR | ICASE | ISAMAX | IWKCIN | SRCH | SVIGP | SVRGP | WRIRN |
| CVTSI | IICSR | ISAMIN | IWKIN | SSRCH | SVRBN | WRCRL | WROPT |
| IACHAR | IIDEX | ISMAX | PAGE | SVIBN | SVRBP | WRCRN | WRRRL |
| ICAMAX | IIMAX | ISMIN | PERMA | SVIBP | SVRGN | WRIRL | WRRRN |
| ICAMIN | IIMIN | ISRCH | PERMU | SVIGN | | | |

**Q : Graphics**

PLOTP

**R : Service routines**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AMACH | ERSET | IERCD | IMACH | NDAYS | NDYIN | TDATE | UMACH |
| DTIME | IDYWK | | | | | | |

**S : Software development tools**

CTIME

# IMSL STAT/LIBRARY

## General Information

**Description** : A Fortran subroutine library for solving problems in statistical analysis. (Approximately 350 subroutines and functions.)

**Portability** : Proprietary

**Reference** : *IMSL STAT/LIBRARY User's Manual*, Vol 1-3. (QA76.73 in NIST Gaithersburg Research Information Center, Admin E-120) IMSL Inc., Houston, TX, 1987.

**Developer** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, Texas 77042-3020

**Distributor** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, Texas 77042-3020 (713-782-6060)

## 855NOS Information

**Version** : 1.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,IMSLS.`

**Module Doc** : `INVOKE,GETDOC,IMSLS,`<*module*>`.`

**Access** : `INVOKE,GETLIB,IMSLS.`
`LIBRARY,IMSLS.`

## 855VE Information

**Version** : 1.0

**Support** : Full

**General Doc** : `FETCH IMSL USAGE`

**Module Doc** : `FETCH IMSL` <*module*> `DOC`

**Access** : `IMSL AL=YES`

## 205 Information

**Version** : 1.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,IMSLS.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,IMSLS,`<*module*>. (Under 855NOS)

**Access** : `PATTACH,MATHPOOL.`
`LOAD,BINARY,LIB=IMSLS.`

## 840NOS Information

**Version** : 1.0

**Support** : Limited

**Access** : `ATTACH,IMSLS/UN=LIB5.`
`LIBRARY,IMSLS.`

## Itemized Contents

**C : Elementary and special functions**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ANORDF | BETA | BETIN | CHIDF | CHIIN | GAMDF | GAMMA | OPOLY |
| ANORIN | BETDF | BINOM | | | | | |

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BLINF | CSROTM | CZDOTI | IIMAX | ISUM | SCASUM | SHOUTR | SSCAL |
| CADD | CSSCAL | CZDOTU | IIMIN | ISWAP | SCNRM2 | SHPROD | SSET |
| CAXPY | CSUB | CZUDOT | ISAMAX | MRRRR | SCOPY | SNRM2 | SSUB |
| CCOPY | CSVCAL | DSDOT | ISAMIN | MXTXF | SDDOTA | SPRDCT | SSUM |
| CDOTC | CSWAP | IADD | ISET | MXTYF | SDDOTI | SROT | SSWAP |
| CDOTU | CVCAL | ICAMAX | ISMAX | SADD | SDOT | SROTG | SVCAL |
| CSCAL | CZCDOT | ICAMIN | ISMIN | SASUM | SDSDOT | SROTM | SXYZ |
| CSET | CZDOTA | ICOPY | ISUB | SAXPY | SHOUAP | SROTMG | TRNRR |
| CSROT | CZDOTC | | | | | | |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

CHFAC LFSRG LFTRG LINRG LSLRG

**D3 : Determinants**

LFDRG

**D4 : Eigenvalues, eigenvectors of matrices**

EPISF EVCSF GVLSP

**D6 : Singular value decomposition (SVD)**

LSVRR

**D9 : Singular, overdetermined or underdetermined systems of linear equations, generalized inverses**

LSBRR LSGRR LSQRR

**G : Optimization**

UMINF

**H : Differentiation, integration**

QDAGS

**J : Integral transforms**

FFTRB FFTRF

**K : Approximation**

RLAV RLEQU RLMV RNLIN RPOLY

**L1 : Data summarization**

CORVC CSTAT GRPES LETTR NTIES UVSTA

**L2 : Data manipulation**

BCTR FREQ GCSCP OPOLY OWFRQ RANKS TWFRQ

**L3 : Elementary statistical graphics**

BOXP CDF2P HHSTP PLOTP SCTP STMLP VHS2P VHSTP

**L4 : Elementary data analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BHAKV | CHIGF | DESPT | KSTWO | PAIRS | RUNS | SMPRR | SMPST |
| BINES | CNCRD | DSQAR | KTRND | POIES | SDPLC | SMPRS | SNRNK |
| BSCAT | CTRHO | INCLD | MVIND | PROBP | SIGNT | SMPSC | TETCC |
| BSPBS | DCUBE | KENDL | NCTRD | QTEST | SMPPR | SMPSR | TWOMV |
| CDF2P | DESKN | KRSKL | NRCES | RNKSM | SMPPS | SMPSS | UVSTA |
| CDFP | DESPL | KSONE | | | | | |

**L5 : Statistical function evaluation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AKS1DF | ANORIN | BINPR | CSNDF | FIN | GCIN | KENDP | TDF |
| AKS2DF | BETDF | BNRDF | ENOS | GAMDF | HYPDF | POIDF | TIN |
| AMILLR | BETIN | CHIDF | FDF | GCDF | HYPPR | POIPR | TNDF |
| ANORDF | BINDF | CHIIN | | | | | |

**L6 : Random number generation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| RNARM | RNEXP | RNGDS | RNLNL | RNNOR | RNPOI | RNSRS | RNUND |
| RNBET | RNEXT | RNGDT | RNMTN | RNNOS | RNSES | RNSTA | RNUNF |
| RNBIN | RNGAM | RNGEO | RNMVN | RNNPP | RNSET | RNTAB | RNUNO |
| RNCHI | RNGCS | RNGES | RNNBN | RNOPG | RNSPH | RNTRI | RNVMS |
| RNCHY | RNGCT | RNGET | RNNOA | RNOPT | RNSRI | RNUN | RNWIB |
| RNCOR | RNGDA | RNHYP | RNNOF | RNPER | | | |

**L7 : Analysis of variance**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ABALD | ALATN | ANWAY | AONEW | CIDMS | FRDMN | ROREX | SNKMC |
| ABIBD | ANEST | AONEC | ATWOB | CTRST | MEDPL | | |

**L8 : Regression**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| OPOLY | RCOMP | RFORP | RINCF | RLEQU | RNLIN | ROTIN | RSTAT |
| RBEST | RCOV | RGIVN | RINPF | RLINE | RONE | RPOLY | RSTEP |
| RCASE | RCOVB | RGLM | RLAV | RLMV | RORDM | RSTAP | TCSCP |
| RCASP | RCURV | | | | | | |

**L9 : Categorical data analysis**

CTCHI CTPRB CTTWO MEDPL PRPFT

**L10 : Time series analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACF | CPFFT | DIFF | IRNSE | MCCF | NSBJF | PACF | SSWD |
| ARMME | CSSWD | DIRIC | KALMN | MLSE | NSLSE | PFFT | SSWP |
| BCTR | CSSWP | FEJER | MAMME | MWFE | NSPE | SPWF | TFPE |
| CCF | | | | | | | |

**L12 : Discriminant analysis**

DMSCR DSCRM NNBRD

**L13 : Covariance structure models**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| FACTR | FDOBL | FIMAG | FPRMX | FROTA | FRVAR | FSCOR | PRINC |
| FCOEF | FHARR | FOPCS | FRESI | | | | |

**L14 : Cluster analysis**

CDIST CLINK CNUMB KMEAN TREEP

**L15 : Life testing and survival analysis**

ACTBL HAZEZ HAZRD HAZST KAPMR

**L16 : Multidimensional scaling**

MSDST MSIDV

**L17 : Statistical data sets**

GDATA

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACHAR | ICAMIN | IIMAX | ISMAX | IWKIN | SCOLR | SVIGP | WRIRN |
| CVTSI | ICASE | IIMIN | ISMIN | PAGE | SRCH | SVRGN | WROPT |
| IACHAR | IICSR | ISAMAX | ISRCH | PERMA | SSRCH | SVRGP | WRRRL |
| ICAMAX | IIDEX | ISAMIN | IWKCIN | PERMU | SVIGN | WRIRL | WRRRN |

**R : Service routines**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AMACH | ERSET | IERCD | IMACH | NDAYS | NDYIN | TDATE | UMACH |
| DTIME | IDYWK | | | | | | |

**S : Software development tools**

CTIME

# IMSL SFUN/LIBRARY

## General Information

**Description** : A Fortran subroutine library for evaluating special functions. (Approximately 175 subroutines and functions.)

**Portability** : Proprietary

**Reference** : *IMSL SFUN/LIBRARY User's Manual*, Vol 1-3. (QA76.73 in NIST Gaithersburg Research Information Center, Admin E-120) IMSL Inc., Houston, TX, 1987.

**Developer** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, Texas 77042-3020

**Distributor** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, Texas 77042-3020 (713-782-6060)

## 855NOS Information

**Version** : 2.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,IMSLSF.`

**Module Doc** : `INVOKE,GETDOC,IMSLSF,`<*module*>`.`

**Access** : `INVOKE,GETLIB,IMSLSF.`
`LIBRARY,IMSLSF.`

## 855VE Information

**Version** : 2.0

**Support** : Full

**General Doc** : `FETCH IMSL USAGE`

**Module Doc** : `FETCH IMSL` <*module*> `DOC`

**Access** : `IMSL AL=YES`

## 205 Information

| | |
|---|---|
| **Version** | : 2.0 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,IMSLSF.` (Under 855NOS) |
| **Module Doc** | : `INVOKE,GETDOC,IMSLSF,`<*module*>`.` (Under 855NOS) |
| **Access** | : `PATTACH,MATHPOOL.` `LOAD,BINARY,LIB=IMSLSF.` |

## 840NOS Information

| | |
|---|---|
| **Version** | : 2.0 |
| **Support** | : Limited |
| **Access** | : `ATTACH,IMSLS/UN=LIB5.` `LIBRARY,IMSLS.` |

## Itemized Contents

**A : Arithmetic, error analysis**

CABS CARG

**C : Elementary and special functions**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACOS | ALOG10 | BI | BSK1 | CCBRT | CLOG | E1 | GAMI |
| ACOSH | ANORDF | BID | BSK1E | CCOS | CLOG10 | EI | GAMIC |
| AI | ANORIN | BIDE | BSKES | CCOSH | COS | ELE | GAMIT |
| AID | ASIN | BIE | BSKS | CCOT | COSDG | ELK | GAMMA |
| AIDE | ASINH | BINOM | BSY0 | CERFE | COSH | ELRC | GAMR |
| AIE | ATAN | BSI0 | BSY1 | CEXP | COT | ELRD | INITS |
| AINT | ATAN2 | BSI0E | BSYS | CEXPRL | CPSI | ELRF | POCH |
| AKEI0 | ATANH | BSI1 | CACOS | CGAMMA | CSEVL | ELRJ | POCH1 |
| AKEI1 | BEI0 | BSI1E | CACOSH | CGAMR | CSIN | ENE | PSI |
| AKEIP0 | BEI1 | BSIES | CASIN | CHI | CSINH | ERF | SHI |
| AKER0 | BEIP0 | BSINS | CASINH | CHIDF | CSQRT | ERFC | SI |
| AKER1 | BER0 | BSIS | CATAN | CHIIN | CTAN | ERFCE | SIN |
| AKERP0 | BER1 | BSJ0 | CATAN2 | CI | CTANH | ERFCI | SINDG |
| ALBETA | BERP0 | BSJ1 | CATANH | CIN | CWPL | ERFI | SINH |
| ALGAMS | BETA | BSJNS | CBETA | CINH | CWPLD | EXP | SPENC |
| ALI | BETAI | BSJS | CBINS | CLBETA | CWPQ | EXPRL | SQRT |
| ALNGAM | BETDF | BSK0 | CBJNS | CLNGAM | CWPQD | FAC | TAN |
| ALNREL | BETIN | BSK0E | CBRT | CLNREL | DAWS | GAMDF | TANH |
| ALOG | | | | | | | |

**L5 : Statistical function evaluation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AKS1DF | BETDF | BNRDF | CSNDF | FDF | GCDF | HYPPR | TDF |
| AKS2DF | BETIN | CHIDF | ERF | FIN | GCIN | POIDF | TIN |
| ANORDF | BINDF | CHIIN | ERFC | GAMDF | HYPDF | POIPR | TNDF |
| ANORIN | BINPR | | | | | | |

**N : Data handling**

IWKCIN IWKIN

**R : Service routines**

AMACH ERSET IERCD IMACH UMACH

# INVAR

## General Information

| | |
|---|---|
| **Description** | : An interactive sytem for solving linear and/or nonlinear least squares problems using a variable separable algorithm. Adapted from the program VARPRO. Features statistical analysis of results and DISSPLA graphics. |
| **Portability** | : Portable, some conversion required |
| **Reference** | : C.M. Wolfe, B.W. Rust, J.H. Dunn, I.E. Brown. An Interactive Nonlinear Least Squares Program, NIST (formerly NBS) Technical Note 1238. NIST, Gaithersburg, MD, 1987. |
| **Developer** | : NIST, Applied and Computational Mathematics Division, Gaithersburg, MD 20899 (B.W. Rust) |
| **Distributor** | : B.W. Rust, NIST, Bldg 225 Room A151, Gaithersburg, MD 20899 (FTS 879-3811 or 301-975-3811) |

## 855NOS Information

| | |
|---|---|
| **Version** | : 1987 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,INVAR.` |
| **Access** | : `INVOKE,INVAR.` |

# JCAM

## General Information

**Description** : A collection of programs and subprograms published in the *Journal of Computational and Applied Mathematics* covering several topics in numerical methods. The programs are reasonably portable. They are available in source form only.

**Portability** : Portable

**Reference** : *Journal of Computational and Applied Mathematics*, c/o Professor R. Piessens, Computer Science Department, Katholieke Universiteit Leuven, Celestijnenlaan 200A, B-3030 Heverlee (BELGIUM)

**Developer** : Computer Science Department, Katholieke Universiteit Leuven, Celestijnenlaan 200A, B-3030 Heverlee (BELGIUM) (R. Piessens)

**Distributor** : *Journal of Computational and Applied Mathematics*, c/o Professor R. Piessens, Computer Science Department, Katholieke Universiteit Leuven, Celestijnenlaan 200A, B-3030 Heverlee (BELGIUM)

## 855NOS Information

**Version** : 1989

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,JCAM.`

**Module Doc** : `INVOKE,GETDOC,JCAM,CONTENT.`

**Access** : `INVOKE,GETSRC,JCAM,`<*module*>.

## Itemized Contents

**DEFINT** Uses double exponential transformation of Mori to compute definite integral automatically to user specified accuracy.

**DEHINT** Uses double exponential transformation of Mori to compute semi-infinite range integral automatically to user specified accuracy.

**DTLS** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution. Algorithm was written by S. Van Huffel (Katholieke Universiteit Leuven, Heverlee, Belgium) under the name 'Extended total least squares.'

**DTRIA** Computation of double integrals over a triangle.

**PSVD** Computes in an efficient and reliable way a basis for the left and/or right singular subspace of a matrix corresponding to its smallest singular values. The dimension of the desired subspace may be given or may depend on a given upper bound for those smallest singular values. Algorithm was written by S. Van Huffel (Catholic Universiteit Leuven, Heverlee, Belgium) under the name 'Partial singular value decomposition algorithm.'

**PTLS** Solves, by using a total least squares approximation, the overdetermined system of equations AX=B where both the data matrix A as well as the observation matrix B are inaccurate. This routine will also solve square and underdetermined systems by computing the minimum norm solution. Algorithm was written by S. Van Huffel (Katholieke Universiteit Leuven, Heverlee, Belgium) under the name 'Partial total least squares,' and is more efficient than the author's routine DTLS.

**SPISC1** Computes a shape preserving polynomial spline of arbitrary degree to a given set of data.

# MAGEV

## General Information

| | |
|---|---|
| **Description** | : A Fortran 200 library of high-performance mathematical and geophysical subroutines designed and developed especially for the Cyber 205. The name, MAGEV, is an acronym for the Math/Geophysical Vector Library. |
| **Portability** | : Machine-specific |
| **Reference** | : None |
| **Developer** | : Petroleum Technology Center, Control Data Corporation, 2000 West Loop South, Houston, Texas 77027 (713)965-5475 (5931) |
| **Distributor** | : Petroleum Technology Center, Control Data Corporation, 2000 West Loop South, Houston, Texas 77027 (713)965-5475 (5931) |

## 205 Information

| | |
|---|---|
| **Version** | : V3.3 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,MAGEV.` (Under 855NOS) |
| **Module Doc** | : `INVOKE,GETDOC,MAGEV,`*<module>*`.` (Under 855NOS) |
| **Access** | : `PATTACH,MATHPOOL.`<br>`LOAD,BINARY,LIB=MAGEV.` |

## Itemized Contents

**A : Arithmetic, error analysis**

| | | | | | | |
|---|---|---|---|---|---|---|
| CV64T60 | CVF32TI2 | CVF32TR4 | CVF64VR4 | CVI2TF32 | CVR4TF32 | CVVR4F64 |

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BBDMPY | FSDIVFV | FSTOFV | FVMPYFV | HCMPRS | MXEQ | MXMPYI | SYMSTO |
| CMXMOV | FSMPYFV | FVADDFV | FVSUBFV | MPYUTU | MXMOV | MXMPYT | TPMOV |
| FSADDFV | FSSUBFV | FVDIVFV | FVTOFV | MXCMP | MXMPY | | |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BBDGEL | CGEL | GEL | INVU | MSWIEN | MXINV | SYMUUT | TRID |
| BDGEL | D4GEL | INVSPD | | | | | |

**D4 : Eigenvalues, eigenvectors of matrices**

| | |
|---|---|
| IMTQL | TRED2 |

**J : Integral transforms**

FFT1D

**N : Data handling**

HMERGE HREV ISORTDA MRGSORT QSORT SLICE4 SORTR

**R : Service routines**

LOCF MAPFL QUADIT

**S : Software development tools**

QUADIT

# MATLAB

## General Information

| | |
|---|---|
| **Description** | : An interactive system for matrix calculations, including solving linear systems, linear least squares problems, eigenvalue and eigenvector calculations, QR decomposition, singular value decomposition, and inverses. Based on LINPACK and EISPACK software. |
| **Portability** | : Portable, some conversion required |
| **Reference** | : C. Moler. *MATLAB User's Guide*, University of New Mexico Technical Report CS81-1 (revised). University of New Mexico, Albuquerque, NM, 1982. |
| **Developer** | : Cleve Moler, The MathWorks, 325 Linfield Place, Menlo Park, CA 94025 |
| **Distributor** | : The Cyber and Vax implementations in use at NIST are early developmental versions of MATLAB which are no longer being distributed or maintained. A commercial version with many additional features is available for a wide range of machines from The MathWorks, 21 Eliot St., South Natick, MA 01760 (508-653-1415, info@mathworks.com). |

## 855NOS Information

| | |
|---|---|
| **Version** | : 1983 |
| **Support** | : Limited |
| **General Doc** | : `INVOKE,GETDOC,MATLAB.` |
| **Module Doc** | : `INVOKE,GETDOC,MATLAB.` |
| **Access** | : `INVOKE,MATLAB.` |

## CAMVAX Information

| | |
|---|---|
| **Version** | : 1983 |
| **Support** | : Limited |
| **General Doc** | : `GETDOC MATLAB` –or– `HELP` (in MATLAB) |
| **Module Doc** | : `GETDOC MATLAB` |
| **Access** | : `MATLAB` |

# MINITAB

## General Information

| | |
|---|---|
| **Description** | : A program for data manipulation, plotting, random number generation, general purpose statistical analysis including regression, time series, EDA (exploratory data analysis), ANOVA, and analysis of tables. (Approximately 150 commands; many elementary commands are not explicitly classified in GAMS.) |
| **Portability** | : Proprietary |
| **Reference** | : T.A. Ryan, Jr., B.L. Joiner, and B.F. Ryan. *Minitab Handbook.* PWS Publishers, Boston, MA, 1985.<br>T.A. Ryan, Jr., B.L. Joiner, and B.F. Ryan. *Minitab Reference Manual*, University Park, PA, 1982. |
| **Developer** | : Minitab, Inc. |
| **Distributor** | : Minitab Project, 3081 Enterprise Drive, State College, PA 16801 (814-238-3230) |

## 855NOS Information

| | |
|---|---|
| **Version** | : 1982.1 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,MINITAB` –or– `HELP` (in MINITAB) |
| **Module Doc** | : `HELP` <*command*> (in MINITAB) |
| **Access** | : `INVOKE,MINITAB.` |

## Itemized Contents

**L1 : Data summarization**

| | | | |
|---|---|---|---|
| CORRELATION | CTABLE | LVALS | TABLE |

**L2 : Data manipulation**

| | | | | |
|---|---|---|---|---|
| CENTER | JOIN | NSCORES | SET | WALSH |
| DIFFERENCES | LAG | RANK | | |

**L3 : Elementary statistical graphics**

| | | | | |
|---|---|---|---|---|
| BOXPLOT | HISTOGRAM | MPLOT | STEM-&-LEAF | TSPLOT |
| CPLOT | LPLOT | PLOT | TPLOT | |

**L4 : Elementary data analysis**

| | | | | |
|---|---|---|---|---|
| KRUSKAL-WALLIS | RUNS | TWOSAMPLE | WINTERVAL | ZINTERVAL |
| MANN-WHITNEY | TINTERVAL | TWOT | WTEST | ZTEST |
| ROOTOGRAM | TTEST | | | |

**L5 : Statistical function evaluation**

| | | | | |
|---|---|---|---|---|
| BINOMIAL | POISSON | | | |

**L6 : Random number generation**

| | | | | |
|---|---|---|---|---|
| BRANDOM | DRANDOM | NRANDOM | SAMPLE | URANDOM |
| BTRIALS | IRANDOM | PRANDOM | | |

**L7 : Analysis of variance**

| | | | | |
|---|---|---|---|---|
| AOVONEWAY | KRUSKAL-WALLIS | MPOLISH | ONEWAY | TWOWAYAOV |

**L8 : Regression**

| | | | | |
|---|---|---|---|---|
| REGRESS | RLINE | RSMOOTH | STEPWISE | |

**L9 : Categorical data analysis**

| | | | | |
|---|---|---|---|---|
| CHISQUARE | MPOLISH | TABLE | | |

**L10 : Time series analysis**

| | | | | |
|---|---|---|---|---|
| ACF | CCF | DIFFERENCES | LAG | PACF |
| ARIMA | | | | |

**N : Data handling**

| | | | | |
|---|---|---|---|---|
| ORDER | SORT | | | |

# NAG

## General Information

**Description** : A Fortran subroutine library for solving standard problems in many areas of mathematics, statistics and optimization. (Approximately 500 subroutines.)

**Portability** : Proprietary

**Reference** : *NAG Fortran Library Manual*, vol. 1-5, (QA297.N3 in NIST Research Information Center, Admin E-120). Numerical Algorithms Group, Downers Grove, IL, 1987.

**Developer** : Numerical Algorithms Group, Inc.

**Distributor** : Numerical Algorithms Group, Inc., 1101 31st Street, Suite 100, Downers Grove, IL 60515 (312-971-2337)

## 855NOS Information

**Version** : Mark12

**Support** : Full

**General Doc** : `INVOKE,GETDOC,NAG.`

**Module Doc** : `INVOKE,GETDOC,NAG,`<*module*>`.`

**Access** : `INVOKE,GETLIB,NAG.`
`LIBRARY,NAG.`

## 855VE Information

**Version** : Mark12

**Support** : Full

**General Doc** : `FETCH NAG USAGE`

**Module Doc** : `FETCH NAG` <*module*> `DOC`

**Access** : `NAG AL=YES`

# 205 Information

**Version** : Mark12

**Support** : Full

**General Doc** : `INVOKE,GETDOC,NAG.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,NAG,`<*module*>`.` (Under 855NOS)

**Access** : `PATTACH,MATHPOOL.`
`LOAD,BINARY,LIB=NAG.`

# Itemized Contents

**A : Arithmetic, error analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| A02ABF | C06BAF | C06GBF | C06GCF | C06GQF | C06GSF | F06BLF | F06CLF |
| A02ACF | | | | | | | |

**C : Elementary and special functions**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| A02AAF | F06CCF | S10ACF | S14AAF | S17ACF | S17AKF | S18CEF | S20ACF |
| C06DBF | F06CDF | S11AAF | S14ABF | S17ADF | S18ACF | S18CFF | S20ADF |
| E02AEF | S07AAF | S11ABF | S15ABF | S17AEF | S18ADF | S19AAF | S21BAF |
| E02AHF | S09AAF | S11ACF | S15ACF | S17AFF | S18AEF | S19ABF | S21BBF |
| E02AJF | S09ABF | S13AAF | S15ADF | S17AGF | S18AFF | S19ACF | S21BCF |
| E02AKF | S10AAF | S13ACF | S15AEF | S17AHF | S18CCF | S19ADF | S21BDF |
| F06BCF | S10ABF | S13ADF | S15AFF | S17AJF | S18CDF | | |

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| C06EKF | F01CQF | F06BNF | F06FAF | F06GAF | F06JDF | F06PDF | F06SDF |
| C06FKF | F01CRF | F06CAF | F06FBF | F06GBF | F06JJF | F06PEF | F06SEF |
| F01CAF | F01CSF | F06CBF | F06FCF | F06GCF | F06JKF | F06PFF | F06SFF |
| F01CBF | F01DAF | F06CHF | F06FDF | F06GDF | F06JLF | F06PGF | F06SGF |
| F01CDF | F01DBF | F06DBF | F06FGF | F06GFF | F06JMF | F06PHF | F06SHF |
| F01CEF | F01DCF | F06DFF | F06FJF | F06GGF | F06KCF | F06PMF | F06SMF |
| F01CFF | F01DDF | F06EAF | F06FKF | F06HBF | F06KDF | F06PPF | F06SNF |
| F01CGF | F01DEF | F06ECF | F06FLF | F06HCF | F06KFF | F06PQF | F06SPF |
| F01CHF | F05ABF | F06EDF | F06FPF | F06HDF | F06KJF | F06PRF | F06SQF |
| F01CKF | F06AAF | F06EFF | F06FQF | F06HGF | F06KLF | F06PSF | F06SRF |
| F01CLF | F06BAF | F06EGF | F06FRF | F06HPF | F06KPF | F06SAF | F06SSF |
| F01CMF | F06BEF | F06EJF | F06FSF | F06HQF | F06PAF | F06SBF | X03AAF |
| F01CNF | F06BHF | F06EKF | F06FTF | F06HRF | F06PBF | F06SCF | X03ABF |
| F01CPF | F06BMF | F06EPF | F06FUF | F06HTF | F06PCF | | |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| F01AAF | F01BRF | F01LZF | F03AHF | F04AGF | F04ASF | F04FAF | F04QAF |
| F01ABF | F01BSF | F01MAF | F04AAF | F04AHF | F04ATF | F04LDF | F06PJF |
| F01ACF | F01BTF | F01MCF | F04ABF | F04AJF | F04AWF | F04LEF | F06PKF |
| F01ADF | F01BUF | F01NAF | F04ACF | F04AKF | F04AXF | F04MAF | F06PLF |
| F01BNF | F01BXF | F03AEF | F04ADF | F04ALF | F04AYF | F04MBF | F06SJF |
| F01BPF | F01LBF | F03AFF | F04AEF | F04AQF | F04AZF | F04MCF | F06SKF |
| F01BQF | F01LEF | F03AGF | F04AFF | F04ARF | F04EAF | F04NAF | F06SLF |

**D3 : Determinants**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| F03AAF | F03ACF | F03ADF | F03AEF | F03AFF | F03AGF | F03AHF | F03AMF |
| F03ABF | | | | | | | |

**D4 : Eigenvalues, eigenvectors of matrices**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| F01AEF | F01AMF | F01AYF | F02AAF | F02AJF | F02ARF | F02BCF | F02BLF |
| F01AFF | F01ANF | F01AZF | F02ABF | F02AKF | F02AVF | F02BDF | F02FHF |
| F01AGF | F01APF | F01BCF | F02ADF | F02AMF | F02AWF | F02BEF | F02FJF |
| F01AHF | F01ATF | F01BDF | F02AEF | F02ANF | F02AXF | F02BFF | F02GJF |
| F01AJF | F01AUF | F01BEF | F02AFF | F02APF | F02AYF | F02BJF | F02SDF |
| F01AKF | F01AVF | F01BVF | F02AGF | F02AQF | F02BBF | F02BKF | F06BPF |
| F01ALF | F01AWF | F01BWF | | | | | |

**D5 : QR decomposition, Gram-Schmidt orthogonalization**

| | | | |
|---|---|---|---|
| F01AXF | F01QAF | F01QBF | F05AAF |

**D6 : Singular value decomposition (SVD)**

| | | | | |
|---|---|---|---|---|
| F02SZF | F02WAF | F02WBF | F02WCF | F02WDF |

**D9 : Singular, overdetermined or underdetermined systems of linear equations, generalized inverses**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| E02GAF | E02GCF | F01BLF | F04ANF | F04JDF | F04JGF | F04QAF | F04YAF |
| E02GBF | E04NCF | F04AMF | F04JAF | | | | |

**E1 : Interpolation of univariate data (curve fitting)**

| | | | | | | |
|---|---|---|---|---|---|---|
| E01AAF | E01ABF | E01AEF | E01BAF | E01RAF | E02AFF | E02BAF |

**E2 : Interpolation of multivariate data (surface fitting)**

E01ACF

**F : Solution of nonlinear equations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| C02ADF | C05ADF | C05AJF | C05AXF | C05NBF | C05PBF | C05ZAF | E04HDF |
| C02AEF | C05AGF | C05AVF | C05AZF | C05NCF | C05PCF | E04HCF | |

**G : Optimization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| E04ABF | E04DEF | E04EBF | E04JBF | E04LAF | E04NDF | E04UEF | E04YBF |
| E04BBF | E04DFF | E04HBF | E04KAF | E04LBF | E04NEF | E04VCF | E04ZCF |
| E04CCF | E04DGF | E04HCF | E04KBF | E04MBF | E04UAF | E04VDF | H02BAF |
| E04CGF | E04DJF | E04HDF | E04KCF | E04NAF | E04UCF | E04XAF | H03ABF |
| E04DBF | E04DKF | E04JAF | E04KDF | E04NCF | E04UDF | E04YAF | |

**H : Differentiation, integration**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| D01AHF | D01AMF | D01ARF | D01BCF | D01EAF | D01FDF | D01GCF | D04AAF |
| D01AJF | D01ANF | D01BAF | D01BDF | D01FBF | D01GAF | D01JAF | E02AJF |
| D01AKF | D01APF | D01BBF | D01DAF | D01FCF | D01GBF | D01PAF | E02BDF |
| D01ALF | D01AQF | | | | | | |

**I1 : Ordinary differential equations (ODE's)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| D02AGF | D02CGF | D02GAF | D02KDF | D02NJF | D02NVF | D02QBF | D02XBF |
| D02BAF | D02CHF | D02GBF | D02KEF | D02NMF | D02NWF | D02QDF | D02XGF |
| D02BBF | D02EAF | D02HAF | D02NBF | D02NNF | D02NXF | D02QQF | D02XHF |
| D02BDF | D02EBF | D02HBF | D02NCF | D02NRF | D02NYF | D02RAF | D02XJF |
| D02BGF | D02EGF | D02JAF | D02NDF | D02NSF | D02NZF | D02SAF | D02XKF |
| D02BHF | D02EHF | D02JBF | D02NGF | D02NTF | D02PAF | D02TGF | D02YAF |
| D02CAF | D02EJF | D02KAF | D02NHF | D02NUF | D02QAF | D02XAF | D02ZAF |
| D02CBF | | | | | | | |

**I2 : Partial differential equations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| D03EAF | D03ECF | D03MAF | D03PAF | D03PBF | D03PGF | D03UAF | D03UBF |
| D03EBF | D03EDF | | | | | | |

**I3 : Integral equations**

| | |
|---|---|
| D05AAF | D05ABF |

**J : Integral transforms**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| C06EAF | C06EKF | C06FBF | C06FFF | C06FKF | C06FQF | C06GQF | C06LAF |
| C06EBF | C06FAF | C06FCF | C06FJF | C06FPF | C06FRF | C06GSF | D01AQF |
| C06ECF | | | | | | | |

**K : Approximation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| E02ACF | E02AGF | E02BAF | E02CAF | E02RAF | E04FDF | E04GEF | E04NDF |
| E02ADF | E02AHF | E02BBF | E02CBF | E02RBF | E04GBF | E04HEF | E04NEF |
| E02AEF | E02AJF | E02BCF | E02DAF | E02ZAF | E04GCF | E04HFF | E04YCF |
| E02AFF | E02AKF | E02BDF | E02DBF | E04FCF | E04GDF | | |

**L1 : Data summarization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| G01AAF | G02BAF | G02BDF | G02BGF | G02BKF | G02BMF | G02BPF | G02BRF |
| G01ABF | G02BBF | G02BEF | G02BHF | G02BLF | G02BNF | G02BQF | G02BSF |
| G01ADF | G02BCF | G02BFF | G02BJF | | | | |

**L2 : Data manipulation**

| | |
|---|---|
| G01AEF | G02CEF |

**L3 : Elementary statistical graphics**

| | |
|---|---|
| G01AGF | G01AJF |

**L4 : Elementary data analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| G01AHF | G08AAF | G08ABF | G08ACF | G08ADF | G08BAF | G08CAF | G08DAF |
| G01DDF | | | | | | | |

**L5 : Statistical function evaluation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| G01BAF | G01BCF | G01CAF | G01CCF | G01CEF | G01DBF | S15ABF | S15ADF |
| G01BBF | G01BDF | G01CBF | G01CDF | G01DAF | G01DCF | S15ACF | S15AEF |

**L6 : Random number generation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| G05CAF | G05CGF | G05DDF | G05DHF | G05DMF | G05EBF | G05EFF | G05EWF |
| G05CBF | G05DAF | G05DEF | G05DJF | G05DPF | G05ECF | G05EGF | G05EXF |
| G05CCF | G05DBF | G05DFF | G05DKF | G05DYF | G05EDF | G05EHF | G05EYF |
| G05CFF | G05DCF | G05DGF | G05DLF | G05DZF | G05EEF | G05EJF | G05EZF |

**L7 : Analysis of variance**

| | | | | | |
|---|---|---|---|---|---|
| G04ADF | G04AEF | G04AFF | G04AGF | G08AEF | G08AFF |

**L8 : Regression**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| G02CAF | G02CCF | G02CEF | G02CGF | G02CHF | G02CJF | G08RAF | G08RBF |
| G02CBF | G02CDF | G02CFF | | | | | |

**L9 : Categorical data analysis**

| |
|---|
| G01AFF |

**L10 : Time series analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| G13AAF | G13AEF | G13AJF | G13BDF | G13BHF | G13CBF | G13CEF | G13DAF |
| G13ABF | G13AFF | G13BAF | G13BEF | G13BJF | G13CCF | G13CFF | G13DBF |
| G13ACF | G13AGF | G13BBF | G13BGF | G13CAF | G13CDF | G13CGF | G13DCF |
| G13ADF | G13AHF | G13BCF | | | | | |

**L13 : Covariance structure models**

G11SAF

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| F06FLF | M01ACF | M01AJF | M01APF | M01BCF | M01DAF | M01DKF | M01ZAF |
| F06JLF | M01ADF | M01AKF | M01AQF | M01BDF | M01DBF | M01DZF | M01ZBF |
| F06JMF | M01AEF | M01ALF | M01ARF | M01CAF | M01DCF | M01EAF | M01ZCF |
| F06KLF | M01AFF | M01AMF | M01BAF | M01CBF | M01DFF | M01EBF | X04BAF |
| M01AAF | M01AGF | M01ANF | M01BBF | M01CCF | M01DJF | M01ECF | X04BBF |
| M01ABF | M01AHF | | | | | | |

**P : Computational geometry**

D03MAF

**R : Service routines**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| P01AAF | X02AAF | X02AEF | X02AJF | X02BAF | X02BEF | X02BLF | X02DJF |
| P01ABF | X02ABF | X02AFF | X02AKF | X02BBF | X02BHF | X02CAF | X04AAF |
| X01AAF | X02ACF | X02AGF | X02ALF | X02BCF | X02BJF | X02DAF | X04ABF |
| X01ABF | X02ADF | X02AHF | X02AMF | X02BDF | X02BKF | | |

# NASHLIB

## General Information

**Description** : A collection of Fortran subprograms from *Compact Numerical Methods for Computers; Linear Algebra and Function Minimisation*, by J.C. Nash. The subprograms are written without many of the extra features usually associated with commercial mathematical software, such as extensive error checking, and are most useful for those applications where small program size is particularly important.

**Portability** : Portable

**Reference** : J.C. Nash. *Compact Numerical Methods for Computers; Linear Algebra and Function Minimisation.* (available in NIST Resource Information Center, ADMIN E-120, or Boulder Radio 1202) John Wiley, NY, 1979.

**Developer** : University of Ottawa, Ottawa, Ontario, Canada (J.C. Nash)

**Distributor** : J.C. Nash, University of Ottawa, Ottawa, Ontario, Canada

## 855NOS Information

**Version** : 1979

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,NASHLIB.`

**Module Doc** : `INVOKE,GETDOC,NASHLIB,CONTENT.`

**Access** : `INVOKE,GETSRC,NASHLIB,`<*module*>. (Source only.)

## 840NOS Information

**Version** : 1979

**Support** : No formal support

**Access** : `ATTACH,NASHLIB/UN=CAMLIB.`
`LIBRARY,NASHLIB.`

# Itemized Contents

**A1A2** Compute singular value decomposition and use it to solve linear least squares problem. Illustrates use of algorithms A1SVD and A2LSVD.

**A3** Givens reduction of a real rectangular matrix. Illustrates use of algorithm A3GR.

**A4A4A** Givens reduction, singular value decomposition and application to solve a linear least squares problem. Illustrates use of algorithms A4A and A4LSGS.

**A5A6** Perform Gaussian elimination with partial pivoting on a general system of linear equations. Illustrates use of algorithms A5GE and A6BS.

**A7A8** Use Choleski algorithm to solve a symmetric system of linear equations. Illustrates use of algorithms A7CH and A8CS.

**A9** Use modification of Gauss Jordan algorithm to perform the Bauer Reinsch inversion of positive definite symmetric matrix. Illustrates use of algorithm A9GJ.

**A10** Use inverse interation via Gaussian elimination.

**A11A12** To standardize a complex vector and compute the residuals of a complex eigenvalue and eigenvector. Illustrates use of algorithms A11VS and A12CVR.

**A13** Use the singular value decomposition to find the eigenvalues and eigenvectors of a real symmetric matrix. Illustrates use of algorithm A13ESV.

**A14A15** Use the Jacobi algorithm to find the eigenvalues and eigenvectors of a real symmetric matrix. Use the Jacobi algorithm to solve the generalized eigenvalue problem for real symmetric matrices. Illustrates use of algorithms A14JE and A15GSE.

**A16** Perform a grid or equal-interval line search. Illustrates use of algorithm A16GS.

**A17** Line search using parabolic inverse interpolation. Illustrates use of algorithm A17LS.

**A18** Finds roots using false position and bisection. Illustrates use of algorithm A18RF.

**A19A20** Finds the minimum of a function of n variables using a Nelder Mead algorithm and axial search. Illustrates use of algorithms A19NM and A20AS.

**A21** Finds the minimum of a function of n variables using a variable metric algorithm. Illustrates use of algorithm A21VM.

**A22** Finds the minimum of a function of n variables using a conjugate gradient algorithm. Illustrates use of algorithm A22CGM.

**A23** Use Marquardt algorithm to find minimum of a sum of squares (nonlinear least squares). Illustrates use of algorithm A23MRT.

**A24** Use conjugate gradient algorithm to find the solution of set of linear equations with nonnegative definite coefficient matrix. Illustrates use of algorithm A24CG.

**A25** Use the conjugate gradient algorithm to find the minimum of a Rayleigh quotient. Illustrates use of algorithm A25RQM.

# NCAR

## General Information

**Description** : A Fortran subprogram library for displaying scientific data. Its capabilities include x-y coordinate plots, contour plots, world maps, solid-colored maps, 2D vector fields, drawing lines in 3D space, halftone (gray scale), background grids, surface views, iso-surfaces of 3D data, text in various fonts, and dashed lines with user-defined patterns. Many output devices are supported, including device-independent output.

**Portability** : Proprietary

**Reference** : *NCAR Graphics User's Guide*, Version 2.0. National Center for Atmospheric Research, Boulder, CO, 1987.

**Developer** : National Center for Atmospheric Research, Scientific Computing Division, P.O. Box 3000, Boulder, CO 80307

**Distributor** : Scientific Computing Division, National Center for Atmospheric Research, P.O. Box 3000, Boulder, CO 80307

## 855NOS Information

**Version** : 2.00

**Support** : Full

**General Doc** : `INVOKE,GETDOC,NCAR.`

**Module Doc** : `INVOKE,GETDOC,NCAR,CONTENT.`

**Access** : `INVOKE,NCAR?`

## 855VE Information

**Version** : 2.00

**Support** : Full

**General Doc** : `FETCH NCAR USAGE`

**Module Doc** : `FETCH NCAR` <*module*> `DOC`

**Access** : `NCAR AL=YES`

## 205 Information

**Version** : 2.00

**Support** : Full

**General Doc** : `INVOKE,GETDOC,NCAR205.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,NCAR,CONTENT.` (Under 855NOS)

**Access** : `INVOKE,NCAR205?` (Under 855NOS)

## 840NOS Information

**Version** : 2.00

**Support** : Full

**General Doc** :
```
GET,GRAFING/UN=GRAF5.
GRAFING.
```

**Module Doc** :
```
GET,GRAFING/UN=GRAF5.
GRAFING.
```

**Access** :
```
GET,GRAFING/UN=GRAF5.
GRAFING.
```

# NMS

## General Information

**Description** : A collection of high-quality, portable Fortran subroutines for solving common computational problems in mathematics, engineering, and statistics. From the book *Numerical Methods and Software* by David Kahaner, Cleve Moler, and Stephen Nash, Prentice Hall, 1989.

**Portability** : Portable

**Reference** : D.K. Kahaner, C. Moler, and S. Nash. *Numerical Methods and Software.* Prentice Hall, Englewood Cliffs, NJ, 1989.

**Developer** : NIST, Applied and Computational Mathematics Division, Gaithersburg, MD 20899 (D. Kahaner) and George Mason University, Operations Research Department, Fairfax, VA 22030 (S. Nash)

**Distributor** : D. Kahaner, NIST, Bldg 225 Room A151, Gaithersburg MD 20899 (FTS 879-3808 or 301-975-3808)

## PC Information

**Version** : 1989

**Support** : Full

**Access** : Contact D.K. Kahaner, NIST, Bldg 225 Room A151, Gaithersburg, MD 20899 (FTS 879-3808 or 301-975-3808)

## Itemized Contents

**BESI0** Modified Bessel function $I_0(x)$. (Double precision version is DBESI0.)

**BESJ** Sequence of Bessel functions $J_{\alpha+k-1}(x), k = 1, \ldots, n$ for non-negative $\alpha$ and $x$. (Double precision version is DBESJ.)

**CFFT2D** Two dimensional fast Fourier transform (FFT), forward or reverse, of a complex n-by-m matrix F. (Double precision version is DCFT2D.)

**CFFTB** Backward complex FFT of complex C(N). Returns result in C. (Double precision version is DCFFTB.)

**CFFTF** Forward complex FFT of complex C(N). Returns result in C. (Double precision version is DCFFTF.)

**ERF** Error function for input argument X. (Double precision version is DERF.)

**ERFC** Complementary error function for real input argument X. (Double precision version is DERFC.)

**EZFFTB** Backward FFT of N data points in AZERO,A(N/2),B(N/2). Returns data in R(N). (Double precision version is DEZFTB.)

**EZFFTF** Forward FFT of N data points R(N). Returns A(N/2) and B(N/2) as cosine and sine coefficients, AZERO as mean. (Double precision version is DEZFTF.)

| | |
|---|---|
| **FMIN** | Finds minimum of REAL FUNCTION F(X) on interval [AX,BX] to accuracy EPS. Needs EXTERNAL statement for user selected function name. Returns FMIN as location of minimum with AX,BX bracketing this value.(Double precision version is DFMIN.) |
| **FZERO** | Finds zero of REAL FUNCTION F(X) on interval [B,C] to relative and absolute accuracy RE, AE. Requires EXTERNAL statement for user selected function name. R is input estimate of location of zero, if this is known. Best output estimate of zero returned in B. (Double precision version is DFZERO.) |
| **GAMMA** | $\Gamma(x)$, where $x$ is not $0, -1, -2, \ldots$ (Double precision version is DGAMMA.) |
| **PCHEV** | Evaluates piecewise cubic and its derivative at NVAL points in array XVAL given N-arrays X,F,D; results are put in arrays FVAL and DVAL. Usually used following a call to PCHEZ, but can be used independently. (Double precision version is DPCHEV.) |
| **PCHEZ** | Finds either spline or visually-pleasing piecewise cubic interpolant to input N-arrays X,F. Evaluate resulting function with PCHEV; integrate it with PCHQA. (Double precision version is DPCHEZ.) |
| **PCHQA** | Integrates piecewise cubic from A to B given N-arrays X,F,D. Usually used in conjunction with PCHEZ to form cubic, but can be used independently, especially if the abscissas are equally spaced. (Double precision version is DPCHQA.) |
| **Q1DA** | Automatic evaluation of a user-defined function of one variable. Special features include randomization and singularity weakening. (Double precision version is DQ1DA.) |
| **QAGI** | Automatic adaptive integrator for semi-infinite or infinite intervals. Uses nonlinear transformation and extrapolation. (Double precision version is DQAGI.) |
| **QK15** | Evaluates integral of given function on an interval with a 15 point Gauss Kronrod formula and returns error estimate. (Double precision version is DQK15.) |
| **RNOR** | Standard normal generator with zero mean and unit standard deviation. Uses ziggaraut algorithm. Fast, excellent statistical properties and portable. |
| **SDRIV2** | Numerical Integration, Initial Value Problems, Ordinary Differential Equations, Gear/Adams Formulas. (Double precision version is DDRIV2.) |
| **SGEFS** | Solves n-by-n system AX=B; returns the B number of correct digits and 1/cond(A). (Double precision version is DGEFS.) |
| **SNSQE** | Solves N nonlinear equations given in SUBROUTINE FCN(N,X,FVEC,IFLAG) to accuracy TOL. If IOPT=2 then Jacobian subroutine JAC need not be supplied. Needs EXTERNAL statement for user selected subroutine name. NMPINT=0 for simple use. Returns solution in X(N), residuals in FVEC(N). (Double precision version is DNSQE.) |
| **SQRLS** | Solves m-by-n system AX=B in least squares sense. (Double precision version is DQRLS.) |
| **SSVDC** | Computes the singular value decomposition of a real n-by-p matrix X, dimensioned X(LDX,P). Has options to allow computation of only the singular values, or singular values and associated decomposition matrices. (Double precision version is DSVDC.) |
| **UNCMIN** | Finds minimum of N-variable function given initial values X0(N). Returns solution in X(N) and minimum function value in F. Needs EXTERNAL statement for user selected SUBROUTINE FCN(N,X,F) returning value of F at X(N). (Double precision version is UNCMND.) |
| **UNI** | Uniform random-number generator on [0,1). Uses Fibbonacci algorithm. Fast, excellent statistical properties and highly portable. |

# PDELIB

## General Information

**Description** : An informal Fortran subroutine library of portable, public-domain routines which solve general systems of nonlinear initial-boundary-value partial differential equations in one or two space dimensions. Each program is based upon the method of lines.

**Portability** : Portable, some conversion required

**Reference** : None

**Developer** : See information about individual programs below

**Distributor** : See information about individual programs below

## 855NOS Information

**Version** : 1983

**Support** : Full

**General Doc** : `INVOKE,GETDOC,PDELIB.`

**Module Doc** : `INVOKE,GETDOC,PDELIB,`<*module*>`.`

**Access** : `INVOKE,GETLIB,PDELIB.`
`LIBRARY,PDELIB.`

## 205 Information

**Version** : 1983

**Support** : Full

**General Doc** : `INVOKE,GETDOC,PDELIB.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,PDELIB,`<*module*>`.` (Under 855NOS)

**Access** : `PATTACH,MATHPOOL.`
`LOAD,BINARY,LIB=PDELIB.`

# Itemized Contents

**MOL1D** Solves systems of linear or nonlinear initial-boundary-value problems in one space dimension. Can solve hyperbolic equations with or without discontinuities, parabolic equations (including reaction-diffusion equations). Uses the method of lines based on equi-spaced finite differences. Graphical output available. (Author: J.M. Hyman, Los Alamos National Lab, Los Alamos, NM 87545)

**PDECOL** Solves general nonlinear systems of initial-boundary-value problems in one space dimension with general boundary conditions. Spatial derivatives may be of at most second order. Uses method of lines based on collocation of B-spline basis functions. (Author: N. Madsen Lawrence Livermore National Lab, Livermore, CA, and R.F. Sincovec, RIACS, NASA Ames, Moffett Field, CA)

**PDETWO** Solves general nonlinear systems of initial-boundary-value problems in two spatial dimensions with quasi-linear boundary conditions. Uses the method of lines based upon finite differences on a user-specified rectangular mesh. (Author: D.K. Melgaard and R.F. Sincovec, RIACS, NASA Ames, Moffett Field, CA)

# PLOD

## General Information

**Description** : An interactive system for solving systems of ordinary differential equations. User may change various conditions, parameters, intervals, etc., interactively and plot results. Almost no programming experience required.

**Portability** : Portable, some conversion required

**Reference** : E. Agron, I. Chang, G. Gunaratna, D. Kahaner, M. Reed. *Mathematical Software: PLOD.* IEEE Micro 8(4):56-61,1988.

**Developer** : NIST, Applied and Computational Mathematics Division, Gaithersburg, MD 20899 (D. Kahaner)

**Distributor** : D. Kahaner, NIST, Bldg 225 Room A151, Gaithersburg MD 20899 (FTS 879-3808 or 301-975-3808)

## PC Information

**Version** : 6.00

**Support** : Full

**Access** : Contact D.K. Kahaner, NIST, Bldg 225 Room A151, Gaithersburg, MD 20899 (FTS 879-3808 or 301-975-3808)

# PLOT10

## General Information

**Description** : A Fortran subprogram library for two-dimensional general purpose computer graphics. Output is supported only on Tektronix terminals. Capabilities include: 1) low-level plotting calls such as move and draw (TCS), 2) higher level calls such as drawing bar plots (AGII), and 3) Calcomp preview routines. The latter are standard Calcomp library calls which allow users to preview pen plotter output on Tektronix terminals.

**Portability** : Proprietary

**Reference** : *PLOT10 TCS Programmers Reference, AGII Programmers Reference, and TCS Model Device Driver.* Tektronix Inc., Gaithersburg, MD, 1988.

**Developer** : Tektronix Inc., P.O. Box 6026, 700 Professional Drive, Gaithersburg, MD 20877

**Distributor** : Tektronix Inc., P.O. Box 6026, 700 Professional Drive, Gaithersburg, MD 20877

## 855NOS Information

**Version** : 5.0

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,PLOT10.`

**Module Doc** : `INVOKE,GETDOC,PLOT10.`

**Access** : `INVOKE,PLOT10?`

## 855VE Information

**Version** : 5.0

**Support** : Limited

**General Doc** : `FETCH PLOT10 USAGE`

**Module Doc** : `FETCH PLOT10 CONTENTS`

**Access** : `PLOT10 AL = YES`

# 840NOS Information

| | |
|---|---|
| **Version** | : 3.0 |
| **Support** | : Limited |
| **General Doc** | : GET,PLOT10/UN=GRFDOC5.<br>PLOT10. |
| **Module Doc** | : GET,PLOT10/UN=GRFDOC5.<br>PLOT10. |
| **Access** | : GET,PLOT10/UN=GRAF5.<br>PLOT10. |

# PORT

## General Information

**Description** : A Fortran subprogram library for solving standard problems in many areas of mathematics. (Approximately 300 subprograms.)

**Portability** : Proprietary

**Reference** : P. Fox et al. *The PORT Mathematical Subroutine Library Manual* (QA297.P2 in NIST Gaithersburg Research Information Center, Admin E-120). Bell Laboratories, Murray Hill, NJ, 1977.

**Developer** : Bell Laboratories, Murray Hill, NJ (P. Fox)

**Distributor** : AT&T Technology Licensing, Guilford Center, P.O. Box 25000, Greensboro, NC 27420, ATTN: D. Rikard (919-279-4061)

## 855NOS Information

**Version** : 2

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,PORT.`

**Module Doc** : `INVOKE,GETDOC,PORT,`<*module*>`.`

**Access** : `INVOKE,GETLIB,PORT.`
`LIBRARY,PORT.`

## 205 Information

**Version** : 2

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,PORT.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,PORT,`<*module*>`.` (Under 855NOS)

**Access** : `PATTACH,MATHPOOL.`
`LOAD,BINARY,LIB=PORT.`

# Itemized Contents

**A : Arithmetic, error analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| CDADD | CNVBDC | CNVBID | CNVBRD | CNVFDI | CNVFID | CNVFRD | UMKFL |
| CDDIV | CNVBDI | CNVBIR | CNVBRI | CNVFDR | CNVFIR | CNVFRI | VBTOD |
| CDMUL | CNVBDR | CNVBRC | CNVFDC | CNVFIC | CNVFRC | MKFL | VDTOB |
| CDSUB | CNVBIC | | | | | | |

**C : Elementary and special functions**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACOSH | ASINH | BESCJ | CDEXP | COSH | IFLR | TAN | TCHBP |
| ARCOS | ATANH | BESRI | CDLOG | FLR | ORTHP | TANH | TRIGP |
| ARSIN | BESCI | BESRJ | CEIL | ICEIL | SINH | | |

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| EXTRMI | MOVEBC | MOVEBL | MOVEFC | MOVEFI | MOVEFR | SETI | SETR |
| EXTRMR | MOVEBI | MOVEBR | MOVEFD | MOVEFL | SETC | SETL | SNRM2 |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

CLINQ LINEQ

**D4 : Eigenvalues, eigenvectors of matrices**

EIGEN

**D9 : Singular, overdetermined or underdetermined systems of linear equations, generalized inverses**

CLST2 LSTSQ

**E1 : Interpolation of univariate data (curve fitting)**

CSPFI CSPIN

**F : Solution of nonlinear equations**

CPOLY RPOLY ZERO ZONE ZONEJ

**G : Optimization**

EXTRMI EXTRMR FMIN

**H : Differentiation, integration**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BQUAD | CSPDI | GAUSQ | GQM11 | ODEQ | QUAD | RQUAD | SPLNI |
| BSPLI | CSPQU | GQ0IN | | | | | |

**I1 : Ordinary differential equations (ODE's)**

ODES ODES1 ODESE ODESH

**J : Integral transforms**

FFT FFTC FFTCI FFTR FFTRI RLTR

**K : Approximation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BSPL1 | BURM1 | EEBSI | ILUMD | IUMB | LUMB | PUMD | SPLNE |
| BSPLD | CSPFE | EESFF | IMNPB | IUMD | LUMD | SPLN1 | SPLNI |
| BSPLI | DL2SF | EESFI | IPUMB | L2SFF | MNPB | SPLN2 | UMB |
| BSPLN | DL2SW | ILUMB | IPUMD | L2SFH | PUMB | SPLND | UMD |
| BURAM | EEBSF | | | | | | |

**L6 : Random number generation**

RANBYT RANSET UNI

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| EXTRMI | ISTKGT | ISTKQU | SRTAH | SRTDH | SRTPAH | SRTPDH | SRTRH |
| EXTRMR | ISTKIN | ISTKRL | SRTAI | SRTDI | SRTPAI | SRTPDI | SRTRI |
| INTRVI | ISTKMD | ISTKST | SRTAR | SRTDR | SRTPAR | SRTPDR | SRTRR |
| INTRVR | | | | | | | |

**R : Service routines**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ENTER | EPRINT | I1MACH | MONOI | NERROR | RETSRC | SMONOI | SMONOR |
| ENTSRC | ERROFF | LEAVE | MONOR | R1MACH | SETERR | | |

# ROSEPAK

## General Information

**Description** : Two interactive programs: IRLS for iteratively reweighted least squares and VNULL for detecting rank degeneracy.

**Portability** : Portable

**Reference** : D. Coleman, P. Holland, N. Kaden, V. Klema, and S.C. Peters. A System of Subroutines for Iteratively Reweighted Least Squares Computations. *ACM Transactions on Mathematical Software*, 6(3):327-336, 1980.

**Developer** : Massachusetts Institute of Technology, Cambridge, MA (D. Coleman, P. Holland, N. Kaden, V. Klema, S.C. Peters)

**Distributor** : IMSL Inc., 2500 ParkWest Tower One, 2500 CityWest Boulevard, Houston, TX 77042-3020 (713-782-6060)

## 855NOS Information

**Version** : 1985

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,ROSEPAK.`

**Module Doc** : `INVOKE,GETDOC,ROSEPAK,`<*module*>`.`

**Access** : `INVOKE,ROSEPAK?`

# SAS

## General Information

**Description** : A program for statistical analyses ranging from simple descriptive statistics to complex multivariate techniques (including regression, analysis of variance, categorical data analysis, multivariate data analysis, discriminant analysis, cluster analysis, and survivial analysis), information storage and retrieval, data modification and programming, and file handling.

**Portability** : Proprietary

**Reference** : *SAS User's Guide: Basic Version 5 Edition.* SAS Institute, Inc., Cary, North Carolina, 1985.

**Developer** : SAS Institute Inc., Box 8000, Cary, North Carolina 27511-8000

**Distributor** : SAS Institute Inc., Box 8000, Cary, North Carolina 27511-8000

## CAMVAX Information

**Version** : 5.18

**Support** : Full

**General Doc** : `HELP SAS` –or– `help;` (in SAS)

**Access** : `SAS`

## Itemized Contents

**L1 : Data summarization**

CORR MEANS SUMMARY TABULATE UNIVARIATE

**L2 : Data manipulation**

RANK STANDARD

**L3 : Elementary statistical graphics**

CHART PLOT TIMEPLOT UNIVARIATE

**L4 : Elementary data analysis**

TTEST

**L6 : Random number generation**

PLAN

**L7 : Analysis of variance**

ANOVA GLM NESTED NPAR1WAY PLAN VARCOMP

**L8 : Regression**

| | | | | | |
|---|---|---|---|---|---|
| NLIN | PROBIT | REG | RSQUARE | RSREG | STEPWISE |

**L9 : Categorical data analysis**

| | |
|---|---|
| CATMOD | FREQ |

**L12 : Discriminant analysis**

| | | | |
|---|---|---|---|
| CANDISC | DISCRIM | NEIGHBOR | STEPDISC |

**L13 : Covariance structure models**

| | | |
|---|---|---|
| CANCORR | FACTOR | PRINCOMP |

**L14 : Cluster analysis**

| | | | | | |
|---|---|---|---|---|---|
| ACECLUS | CLUSTER | FASTCLUS | NEIGHBOR | TREE | VARCLUS |

**L15 : Life testing and survival analysis**

| | |
|---|---|
| LIFEREG | LIFETEST |

**N : Data handling**

SORT

# SCIENTIFIC DESK

## General Information

**Description** : A Fortran subroutine library for solving a variety of mathematical and statistical problems. Also, a collection of self-contained problem solvers which do not require any progamming. Both are for use on PC's. Supports both Lahey Fortran F77L, and RM-FORT.

**Portability** : Proprietary

**Reference** : *The Scientific Desk.* C. Abaci, Inc., Raleigh, NC, 1989.

**Developer** : C. Abaci, Inc., Raleigh, NC (E.L. Battiste)

**Distributor** : M. McClain, NIST, Bldg 225 Room A151, Gaithersburg, MD 20899 (FTS 879-3837 or 301-975-3837) or E. L.Battiste, C. Abaci, 208 St. Mary's St., Raleigh, NC 27605

## PC Information

**Version** : 4

**Support** : Full

**General Doc** : Move to DESK directory and enter DESK.

**Module Doc** : Move to DESK directory and enter DESK.

**Access** : Link to Fortran subroutines (compiler dependent); invoke problem solvers by name.

## Itemized Contents

**A : Arithmetic, error analysis**

| | | |
|---|---|---|
| A4BMD | A4BMOD | AZA4D |

**C : Elementary and special functions**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| C10A3J | C10B3K | C10DAD | C14RF | C4ACT | C4BCXP | C4CTH | C7BPSI |
| C10AHD | C10BI0 | C10DAE | C14RJ | C4ACT2 | C4BEXP | C5 | C7BTA |
| C10AJ0 | C10BI1 | C10DAM | C2CCUB | C4ACTI | C4CACH | C7ACLG | C7EG |
| C10AJ1 | C10BID | C10DB | C2CSQD | C4ACTN | C4CASH | C7AG | C7EGC |
| C10AJD | C10BK0 | C10DBD | C2CUB | C4BALR | C4CATH | C7ALG | C8AE |
| C10AY0 | C10BK1 | C10DBE | C4ACCO | C4BCLD | C4CCH | C7BCLB | C8AEI |
| C10AY1 | C10BKD | C14RC | C4ACCT | C4BCLR | C4CSH | C7BCPS | C8CD |
| C10AYD | C10DA | C14RD | C4ACSN | C4BCXD | C4CSHD | C7BLB | F1A1E |
| C10B3 | | | | | | | |

**D1 : Elementary vector and matrix operations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| D1A3A | D1A41 | D1A6 | D1A8G | D1B22 | D1B5 | D1B63 | D1B6F |
| D1A3B | D1A5 | D1A7 | D1B2 | D1B2S | D1B61 | D1B64 | D1B9 |
| D1A3C | D1A5C | D1A8 | D1B21 | D1B3 | D1B62 | D1B65 | |

**D2 : Solution of systems of linear equations (including inversion, LU and related decompositions)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| D2A11 | D2A13 | D2A21 | D2A2A | D2B12 | D2B14 | D2B1A2 | D2B1A4 |
| D2A12 | D2A14 | D2A22 | D2B11 | D2B13 | D2B1A | D2B1A3 | D2B2S |

**D3 : Determinants**

D2A13 D2B13 D2B1A3

**D4 : Eigenvalues, eigenvectors of matrices**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| D | D4A21 | D4C1A | D4C1C | D4C23 | D4C2A | D4C2C | D4C4B |
| D4A11 | D4A22D | D4C1B | D4C1D | D4C24 | D4C2B | D4C4 | |

**D6 : Singular value decomposition (SVD)**

D6SVD

**D9 : Singular, overdetermined or underdetermined systems of linear equations, generalized inverses**

D9GLS D9GLU K1A2CD K1A2D

**E1 : Interpolation of univariate data (curve fitting)**

E1ACS E1AM

**F : Solution of nonlinear equations**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| F1A1A | F1A1E | F1A21 | F1A22 | F2B | F2BE | F2BP | F2BPE |
| F1A1B | | | | | | | |

**G : Optimization**

G2A1 G4C

**H : Differentiation, integration**

E3HIN E3INT H2A1 H2A1T H2A1U H2B2A

**I1 : Ordinary differential equations (ODE's)**

I1A1A I1A2 I1A2E I1A2F

**J : Integral transforms**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| J1A2B | J1A3B | J1A3D | J1A3F | J1A3H | J1AEB | J1B2T | J3LD |
| J1A2F | J1A3C | J1A3E | J1A3G | J1A3S | J1AET | | |

**K : Approximation**

G1B2E G1B2U G4F L8G1A L8G1E

**L1 : Data summarization**

L1A1C L1E1B

**L2 : Data manipulation**

L8G1A L8G1E

**L4 : Elementary data analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| L4A1BU | L4A1CD | L4A1D | L4A1DP | L4A1DT | L4A1M | L4B1 | L4B1BR |
| L4A1CC | | | | | | | |

**L5 : Statistical function evaluation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| L4ARD | L5A11 | L5A1F | L5A1H | L5A1L | L5A1T | L5A2C | L5A2N |
| L5A1 | L5A1B | | | | | | |

**L6 : Random number generation**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| L6A12 | L6A14T | L6A20 | L6A21T | L6A25 | L6A3C | L6A76 | L6B13 |
| L6A14F | L6A16 | L6A21F | L6A23 | L6A2B | L6A5 | L6A9 | L6B14 |
| L6A14S | L6A2 | L6A21S | L6A24 | L6A3 | L6A7 | | |

**L7 : Analysis of variance**

L7A11

**L8 : Regression**

| | | | | | |
|---|---|---|---|---|---|
| L8A1A | L8A2BD | L8CP | L8CS | L8G1A | L8G1E |

**N : Data handling**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| N1ECC | N1M | N1RS | N1VSD | N3LNB | N6A2 | N6A2AD | N6A2P |
| N1EEC | N1MSD | N1TC | N3 | N3LOW | N6A2A | N6A2D | N6A2PD |
| N1LM | N1RG | N1V | N3FNB | N3UP | | | |

**P : Computational geometry**

P4MED

**Q : Graphics**

L3C3

**R : Service routines**

| | | | | |
|---|---|---|---|---|
| R1MAC | R1MACI | R3A | R3B | R3C |

# SCRUNCH

## General Information

**Description** : A collection of subprograms, written in BASIC, for mathematical and scientific computations. The subprograms provide the capability to solve systems of linear equations, fit data with splines and by least squares, perform eigenanalysis, find minima and zeros of functions, compute the singular value decomposition of a matrix, and solve systems of differential equations. Available in source form only.

**Portability** : Portable

**Reference** : G. Forsythe, M. Malcolm, and C. Moler. *Computer Methods for Mathematical Computations.* (available at Gaithersburg NIST Resource Information Center, ADMIN E-120, and in Boulder, Radio 1202). Prentice Hall, Englewood Cliffs, NJ, 1977.

**Developer** : San Diego State University, San Diego, California (K. Stewart)

**Distributor** : D. Kahaner, NIST, Bldg 225 Room A151, Gaithersburg MD 20899 (FTS 879-3808 or 301-975-3808)

## 855NOS Information

**Version** : 2.0

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,SCRUNCH.`

**Module Doc** : `INVOKE,GETDOC,SCRUNCH,CONTENT.`

**Access** : `INVOKE,GETSRC,SCRUNCH,`<*module*>`.`

## Itemized Contents

**DECOMP** Decomposes a matrix using Gaussian elimination into a lower and upper triangular factorization and estimates the condition of the matrix. This routine is usually used in conjunction with SOLVE which takes the LU factorization produced by DECOMP to actually solve a system of linear equations.

**FNM** Calculates an approximation to the point where a user defined function attains a minimum on a given interval.

**HECOMP** Performs Householder reduction of a rectangular matrix to upper trianglular form. Usually used in conjunction with HOLVE to find the least squares solution of an overdetermined linear system.

**RKF45** Runge-Kutta Felberg method for the integration of a first order system of ordinary differential equations.

**SIMP** Calculates an estimate of the definite integral of a user supplied function by adaptive quadrature.

**SPLINE** Calculates the coefficients of a spline interpolant to given data. Usually used in conjunction with SEVAL to subsequently evaluate the spline at arbitrary points.

**SVD** Calculates the singular value decomposition (SVD) of a given rectangular matrix.

**SYMEIG** Computes the eigenvalues and, optionally, the eigenvectors of a real symmertric matrix.

**ZEROIN** Finds a root of a nonlinear user-supplied function given an initial interval in which the function changes sign.

# SPECFN

## General Information

**Description** : Fortran subroutines for the error and complementary error functions, the exponential and scaled exponential integrals, sine integrals, cosine integrals, hyperbolic sine integrals, and hyperbolic cosine integrals. The methods used are power series, continued fractions, and asymptotic series.

**Portability** : Portable

**Reference** : I. A. Stegun and R. Zucker. *Journal of Research of the National Bureau of Standards.* ERRINT: Automatic Computing Methods for Special Functions. 74B(3), 1970. EXPINT: Automatic Computing Methods for Special Functions. Part II. The Exponential Integral E_n(x). 78B(4),1974. SICIEI: Automatic Computing Methods for Special Functions. Part III. The Sine, Cosine, Exponential Integrals and Related Functions. 80B(2), 1976.

**Developer** : NIST, Gaithersburg, MD 20899 (I. A. Stegun and R. Zucker)

**Distributor** : D. Lozier, NIST, Bldg 101 Room A302, Gaithersburg, MD 20899 (FTS 879-2706 or 301-975-2706)

## 855NOS Information

**Version** : 1976

**Support** : No formal support

**General Doc** : `INVOKE,GETDOC,SPECFN.`

**Module Doc** : `INVOKE,GETDOC,SPECFN,CONTENT.`

**Access** : `INVOKE,GETSRC,SPECFN,`<*name*>, where <*name*> is a subprogram name (see Module Doc).

## Itemized Contents

**ERRINT** Error function and complementary error function.
**EXPINT** Exponential integrals and scaled exponential integrals.
**SICIEI** Sine, cosine, exponential, scaled exponential, hyperbolic sine and hyperbolic cosine integrals.

# SPSS

## General Information

**Description** : A program for data definition, variable manipulation, file management, and statistical analyses (including frequency distributions and descriptive statistics, correlation coefficients and scatterplots, regression, factor analysis, discriminant analysis, survival analysis, nonparametric statistics, analysis of additive scales, comparisons of means, log-linear models, multidimensional scaling, and Box-Jenkins time series analysis). (Approximately 28 commands.)

**Portability** : Proprietary

**Reference** : *SPSSX User's Guide*, 2nd Edition. SPSS Inc., Chicago, IL, 1986.

**Developer** : SPSS Inc., 444 N. Michigan Avenue, Chicago, Illinois 60611

**Distributor** : SPSS Inc., 444 N. Michigan Avenue, Chicago, Illinois 60611

## 855NOS Information

**Version** : 2.1

**Support** : Full

**General Doc** : `INVOKE,GETDOC,SPSS.`

**Module Doc** : `INVOKE,GETDOC,SPSS,CONTENT.`

**Access** : `INVOKE,SPSS?`

## 855VE Information

**Version** : 2.2

**Support** : Full

**General Doc** : `FETCH SPSSX USAGE`

**Module Doc** : `FETCH SPSSX CONTENTS`

**Access** : `SPSSX`

## 840NOS Information

| | |
|---|---|
| **Version** | : 2.1 |
| **Support** | : Full |
| **General Doc** | : `GET,SPSSDOC/UN=SPSSX.`<br>`SPSSDOC.` |
| **Module Doc** | : `GET,SPSSDOC/UN=SPSSX.`<br>`SPSSDOC.` |
| **Access** | : `ATTACH,SPSSX/UN=SPSSX.`<br>`SPSSX.` |

## Itemized Contents

**L1 : Data summarization**

AGGREGATE, CONDESCRIPTIVE, FREQUENCIES, PEARSON CORR, REPORT, BREAKDOWN, CROSSTABS

**L2 : Data manipulation**

MULT RESPONSE

**L3 : Elementary statistical graphics**

FREQUENCIES, PLOT

**L4 : Elementary data analysis**

NONPAR CORR, NPAR TESTS, PEARSON CORR, T-TEST

**L7 : Analysis of variance**

ANOVA, MANOVA, NPAR TESTS, ONEWAY

**L8 : Regression**

PROBIT, REGRESSION

**L9 : Categorical data analysis**

CROSSTABS, HILOGLINEAR, LOGLINEAR

**L10 : Time series analysis**

BOX-JENKINS

**L11 : Correlation analysis**

PARTIAL CORR

**L12 : Discriminant analysis**

DISCRIMINANT

**L13 : Covariance structure models**

FACTOR

**L14 : Cluster analysis**

CLUSTER, QUICK CLUSTER

**L15 : Life testing and survival analysis**

SURVIVAL

**N : Data handling**

SORT CASES

# STARPAC

## General Information

**Description** : A Fortran subprogram library for statistical data analysis. Its primary applications are time series and nonlinear least squares analysis, but it also includes subprograms for normal random number generation, line printer plots, basic statistical analysis, and linear least squares regression. STARPAC emphasizes the statistical interpretation of computed results.

**Portability** : Portable

**Reference** : J.R. Donaldson and P.V. Tryon. *User's Guide to STARPAC, The Standards Time Series and Regression Package.* NIST, Boulder, CO, 1987.

**Developer** : NIST, Applied and Computational Mathematics Division, Boulder, CO (J.R. Donaldson and P.V. Tryon)

**Distributor** : J.R. Donaldson, NIST, Mail Stop 719, 325 Broadway, Boulder, CO 80303-3328 (FTS 320-5114 or 303-497-5114)

## 855NOS Information

**Version** : 2.07

**Support** : Full

**General Doc** : `INVOKE,GETDOC,STARPAC.`

**Module Doc** : `INVOKE,GETDOC,STARPAC,`<*module*>`.`

**Access** : `INVOKE,GETLIB,STARPAC.`
`LIBRARY,STARPAC.`

## 855VE Information

**Version** : 2.07

**Support** : Full

**General Doc** : `FETCH STARPAC USAGE`

**Module Doc** : `FETCH STARPAC` <*module*> `DOC`

**Access** : `STARPAC AL = YES`

## 205 Information

**Version** : 2.07

**Support** : Full

**General Doc** : `INVOKE,GETDOC,STARPAC.` (Under 855NOS)

**Module Doc** : `INVOKE,GETDOC,STARPAC,`<*module*>`.` (Under 855NOS)

**Access** : `PATTACH,MATHPOOL.`
`LOAD,BINARY,LIB=STARPAC.`

## CAMVAX Information

**Version** : 2.07

**Support** : Limited

**General Doc** : `HELP STARPAC`

**Module Doc** : `HELP STARPAC`

**Access** : `LINK` <*main*>`, SYS$USER:STARPAC/LIB`

## 840NOS Information

**Version** : 2.07

**Support** : Full

**General Doc** : `GET,DOC/UN=STARPAC.`
`GTR,DOC,`<*file*>`.INTRO.`
`FSE,`<*file*>`,A.`

**Module Doc** : See library documentation.

**Access** : `ATTACH,STARPAC/UN=STARPAC,NA.`
`LIBRARY,STARPAC/A.`

## Itemized Contents

**J : Integral transforms**

FFTLEN FFTR

**L1 : Data summarization**

STAT STATS STATW STATWS

**L2 : Data manipulation**

CENTER SAMPLE

**L3 : Elementary statistical graphics**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| HIST | MPPM | MVPL | PPC | SPP | SPPML | SVPMC | VPL |
| HISTC | MPPMC | MVPM | PPL | SPPC | SVP | SVPML | VPM |
| MPP | MPPML | MVPMC | PPM | SPPL | SVPC | VP | VPMC |
| MPPC | MVP | MVPML | PPMC | SPPM | SVPL | VPC | VPML |
| MPPL | MVPC | PP | PPML | SPPMC | SVPM | | |

**L4 : Elementary data analysis**

| | | | | | |
|---|---|---|---|---|---|
| CORR | CORRS | STAT | STATS | STATW | STATWS |

**L6 : Random number generation**

| | |
|---|---|
| NRAND | NRANDC |

**L7 : Analysis of variance**

| | |
|---|---|
| AOV1 | AOV1S |

**L8 : Regression**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| DCKLS | LLSP | LLSPWS | LLSWS | NLSD | NLSS | NLSWD | NLSWS |
| DCKLSC | LLSPS | LLSS | NLS | NLSDC | NLSW | NLSWDC | STPLS |
| LLS | LLSPW | LLSW | NLSC | NLSDS | NLSWC | NLSWDS | STPLSC |

**L10 : Time series analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| ACF | AIMES | BFSMV | CCFMS | GFARFS | LOPASS | UAS | UFSFS |
| ACFD | AIMF | BFSMVS | CCFS | GFSLF | LPCOEF | UASF | UFSM |
| ACFF | AIMFS | BFSS | DEMOD | GFSLFS | MAFLT | UASFS | UFSMS |
| ACFFS | ARFLT | BFSV | DEMODS | HIPASS | MDFLT | UASS | UFSMV |
| ACFM | BFS | BFSVS | DIF | HPCOEF | PGM | UASV | UFSMVS |
| ACFMS | BFSF | CCF | DIFC | IPGM | PGMS | UASVS | UFSS |
| ACFS | BFSFS | CCFF | DIFM | IPGMP | SAMPLE | UFS | UFSV |
| AIME | BFSM | CCFFS | DIFMC | IPGMPS | SLFLT | UFSF | UFSVS |
| AIMEC | BFSMS | CCFM | GFARF | IPGMS | TAPER | | |

# TEMPLATE

## General Information

**Description** : A device and computer-independent general-purpose, two and/or three dimensional Fortran-callable graphics system for static, interactive, and dynamic applications. Included are line drawing, text generation, coordinate system generation, windows, blanking regions, charting and graphing (e.g., contour and surface plots, business graphics), interactive input, graphics structures, display list operations, and color redefinition and selection.

**Portability** : Proprietary

**Reference** : *TEMPLATE V5.0, 2D/3D Reference Manual.* Megatek Corporation, San Diego, CA, 1985.

**Developer** : Megatek Corporation, 9645 Scranton Road, San Diego, CA 92121

**Distributor** : Megatek Corporation, 9645 Scranton Road, San Diego, CA 92121

## 855NOS Information

**Version** : 6.0

**Support** : Full

**General Doc** : `INVOKE,GETDOC,TEMPLAT.`

**Module Doc** : `INVOKE,GETDOC,TEMPLAT,CONTENT.`

**Access** : `INVOKE,TEMPLAT?`

## 855VE Information

**Version** : 6.0

**Support** : Full

**General Doc** : `FETCH TEMPLATE USAGE`

**Module Doc** : FETCH TEMPLATE <*module*> DOC

**Access** : TEMPLATE PROGRAM=<*file*> DRIVER=<*name*>

## 205 Information

| | |
|---|---|
| **Version** | : 6.0 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,TMPL205.` (Under 855NOS) |
| **Module Doc** | : `INVOKE,GETDOC,TEMPLAT,CONTENT.` (Under 855NOS) |
| **Access** | : `INVOKE,TMPL205?` (Under 855NOS) |

## CAMVAX Information

| | |
|---|---|
| **Version** | : 6.0 |
| **Support** | : Full |
| **General Doc** | : `HELP TEMPLATE` |
| **Module Doc** | : `HELP TEMPLATE`, then choose `CONTENT` |
| **Access** | : `TEMPLATE` |

# TIMSAC

## General Information

**Description** : Fortran programs for time series analysis and control using Householder transformations for least squares computation, Bayesian type model fitting procedures and exact maximum likehood computation for the autoregressive and autoregressive moving average models.

**Portability** : Portable

**Reference** : H. Akaike, G. Kitagawa, E. Arahata and F. Tada. *TIMSAC-78* Computer Science Monographs. The Institute of Statistical Mathematics, No. 11, February 1979.

**Developer** : The Institute of Statistical Mathematics, 6-7, 4-chome, Minami-Azabu, Minato-ku, Tokyo, 106, Japan (H. Akaike)

**Distributor** : Division of Mathematical and Computer Sciences, The University of Tulsa, Tulsa, OK 74104

## 855NOS Information

**Version** : 78

**Support** : Limited

**General Doc** : `INVOKE,GETDOC,TIMSAC.`

**Module Doc** : `INVOKE,GETDOC,TIMSAC,`<*module*>`.`

**Access** : `INVOKE,TIMSAC?`

## Itemized Contents

**L6 : Random number generation**

NADCON

**L10 : Time series analysis**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| BLOCAR | BSUBST | MLOCAR | MULBAR | PERARS | UNIBAR | UNIMAR | XSARMA |
| BLOMAR | EXSAR | MLOMAR | MULMAR | | | | |

# VolksGrapher

## General Information

**Description** : A Fortran subroutine library for two dimensional graphics. Its main features are ease of use and interactive capabilities, such as automatic zooming, adding and moving text on the screen, etc. Runs on a variety of computers, such as PC, Sun, VAX, etc.

**Portability** : Portable, some conversion required

**Reference** : Anderson, W., Candela, G., Kahaner, D., Kattner, U. *VolksGrapher*, Version 3.00. Gaithersburg, MD, 1989.

**Developer** : NIST, Applied and Computational Mathematics Divison, Gaithersburg, MD 20899 (D. Kahaner)

**Distributor** : D. Kahaner, NIST, Bldg 225 Room A151, Gaithersburg MD 20899 (FTS 879-3808 or 301-975-3808)

## CAMVAX Information

**Version** : 3.00

**Support** : Full

**General Doc** : `HELP VG`

**Access** : `LINK` <*main*>`, ..., VG/LIB`

## PC Information

**Version** : 3.00

**Support** : Full

**Access** : Link to VG (differs for F77L and RMFORT compilers)

# XMLIBV

## General Information

| | |
|---|---|
| **Description** | : A Fortran 200 subprogram library for solving a variety of mathematical problems efficiently on the Cyber 205. The initial version of the library contains subprograms for the evaluation of various special functions for a vector of arguments. |
| **Portability** | : Machine-specific |
| **Reference** | : None |
| **Developer** | : NIST, Applied and Computational Mathematics Division, Gaithersburg, MD 20899 (R. Boisvert) |
| **Distributor** | : J. Springmann, NIST, Bldg 225 Room B146, Gaithersburg, MD 20899 (FTS 879-3820 or 301-975-3820) |

## 205 Information

| | |
|---|---|
| **Version** | : 1 |
| **Support** | : Full |
| **General Doc** | : `INVOKE,GETDOC,XMLIBV.` (Under 855NOS) |
| **Module Doc** | : `INVOKE,GETDOC,XMLIBV,`<*module*>`.` (Under 855NOS) |
| **Access** | : `PATTACH,MATHPOOL.`<br>`LOAD,BINARY,LIB=XMLIBV.` |

## Itemized Contents

| | |
|---|---|
| **VBESY0** | Bessel function of the second kind of order zero ($Y_0$) for a vector of real arguments |
| **VBESY1** | Bessel function of the second kind of order one ($Y_1$) for a vector of real arguments |
| **VBESI0** | Modified (hyperbolic) Bessel function of the first kind of order zero ($I_0$) for a vector of real arguments |
| **VBESI1** | Modified (hyperbolic) Bessel function of the first kind of order one ($I_1$) for a vector of real arguments |
| **VBESJ0** | Bessel function of the first kind of order zero ($J_0$) for a vector of real arguments |
| **VBESJ1** | Bessel function of the first kind of order one ($J_1$) for a vector of real arguments |
| **VBESK0** | Modified (hyperbolic) Bessel function of the third kind of order zero ($K_0$) for a vector of real arguments |
| **VBESK1** | Modified (hyperbolic) Bessel function of the third kind of order one ($K_1$) for a vector of real arguments |
| **VSINH** | Hyperbolic sine (sinh) for a vector of real arguments |
| **VCOSH** | Hyperbolic cosine (cosh) for a vector of real arguments |

# ZETA

## General Information

| | | |
|---|---|---|
| **Description** | : | A Fortran subroutine library which provide access to the Zeta plotter, an off-line, 4 color, pen plotter. The library contains the standard Calcomp library routines with modest extensions. |
| **Portability** | : | Proprietary |
| **Reference** | : | *Fundamental Plotting Routines.* Bruning, Inc., Martinez, CA 1984 |
| **Developer** | : | Bruning Computer Graphics, 777 Arnold Drive, Martinez, CA 94553 |
| **Distributor** | : | Bruning Computer Graphics, 777 Arnold Drive, Martinez, CA 94553 |

## 855NOS Information

| | | |
|---|---|---|
| **Version** | : | 5.3 |
| **Support** | : | Limited |
| **General Doc** | : | `INVOKE,GETDOC,ZETA.` |
| **Module Doc** | : | `INVOKE,GETDOC,ZETA,CONTENT.` |
| **Access** | : | `INVOKE,GETLIB,ZETA.`<br>`LIBRARY,ZETA.` |

## 855VE Information

| | | |
|---|---|---|
| **Version** | : | 5.3 |
| **Support** | : | Limited |
| **General Doc** | : | `FETCH ZETA USAGE` |
| **Module Doc** | : | `FETCH ZETA CONTENTS` |
| **Access** | : | `ZETA AL = YES` |

# Index

*This section is a keyword index to the GAMS problem classification scheme. For each keyword, it provides one or more GAMS classes where software on that topic may be found.*

NBS-114A (REV. 2-8C)

| U.S. DEPT. OF COMM. **BIBLIOGRAPHIC DATA SHEET** *(See instructions)* | 1. PUBLICATION OR REPORT NO. NISTIR 90-4237 | 2. Performing Organ. Report No. | 3. Publication Date MARCH 1990 |
|---|---|---|---|

4. TITLE AND SUBTITLE

Guide to Available Mathematical Software

5. AUTHOR(S)

R. F. Boisvert, S. E. Howe, D. K. Kahaner, and J. L. Springmann

6. PERFORMING ORGANIZATION *(If joint or other than NBS, see instructions)*

**NATIONAL BUREAU OF STANDARDS**
**U.S. DEPARTMENT OF COMMERCE**
**GAITHERSBURG, MD 20899**

7. Contract/Grant No.

8. Type of Report & Period Covered

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS *(Street, City, State, ZIP)*

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)*

The Guide to Available Mathematical Software (GAMS) is a catalog of mathematical and statistical software which has been made available to computer users at the National Institute of Standards and Technology by its Center for Computing and Applied Mathematics. This edition includes descriptions of 5069 software modules comprising 40 software packages implemented on six different computer systems. The catalog is divided into five parts: (1) a classification scheme for mathematical and statistical problems, (2) a list of software modules organized according to the problems they solve, (3) a more detailed list of software modules in alphabetical order, (4) a description of each software package and its implementations, and (5) a keyword index to the problem classification scheme.

12. KEY WORDS *(Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)*

classification scheme; mathematical software; numerical software; software catalog; software documentation; statistical software

13. AVAILABILITY

[X] Unlimited
☐ For Official Distribution. Do Not Release to NTIS
☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
[X] Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES
682

15. Price
A99

USCOMM-DC 6043-P80

AMERICAN BINDERY - EAST
1993