Applied and
Computational
Mathematics
Division

NISTIR 4490

Center for Computing and Applied Mathematics

Review of Mathematical Function Library for Microsoft FORTRAN, John Wiley & Sons, 1989

D.W. Lozier and F.W.J. Olver

December 1990

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology
Gaithersburg, MD 20899

NISTIR 4490

Review of Mathematical Function Library for Microsoft-FORTRAN, John Wiley & Sons, 1989

D. W. Lozier

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology
Center for Computing and Applied Mathematics
Applied & Computational Mathematics Division
Gaithersburg, MD 20899

F. W. J. Olver

Institute for Physical Science and Technology
University of Maryland
College Park, MD 20742

December 1990

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher, Secretary

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
John W. Lyons, Director

ABSTRACT

A review of Mathematical Function Library for Microsoft-FORTRAN, John Wiley & Sons, 1989, and associated computer software is presented. The package consists of xvii + 341 pages, 25 1/2 cm, loose 3-hole punched leaflets in a ring binder and three 5 1/4" diskettes. Price: $295.

KEYWORDS

book review; mathematical functions; software review; special functions of mathematical physics and statistics

DISCLAIMER

This paper presents the results of research which produces a preliminary evaluation of numerical computing software made available in a commercial product. The research is published to coordinate and encourage work in the important area of software quality; the results of the research are not to be interpreted as an official assessment by the NIST.

1. INTRODUCTION AND DESCRIPTION OF THE UL LIBRARY

In 1964 the National Bureau of Standards (now the National Institute of Standards and Technology) issued a massive handbook of formulas, graphs and numerical tables of the elementary mathematical functions and the so-called higher transcendental functions or special functions of mathematical physics [6]. This NBS Handbook immediately filled, and continues to fill, a tremendous need in scientific work; according to the Science Citation Index (published by the Institute for Scientific Information, Inc., Philadelphia, PA), the current rate at which it is cited in the mathematical and scientific literature is of the order of 1,300 entries per year.

The need for the numerical tables of the elementary functions in the NBS Handbook has now largely disappeared: library routines for generating exponentials, logarithms, and trigonometric and hyperbolic functions are available in all major scientific software libraries as well as being required by the FORTRAN Standard [1]. Also, these routines are incorporated in hand-held calculators designed for scientific calculations.

The loose-leaf manual and accompanying diskettes under review, which we shall refer to as the "**UL Library**", may be regarded as an attempt to replace the numerical tables of the higher transcendental functions supplied in the NBS Handbook by a comprehensive software package. The functions treated include Bessel and related functions, hypergeometric and confluent hypergeometric functions, elliptic functions and integrals, exponential integral and related functions, error function and related functions, Gamma and incomplete Gamma functions, orthogonal polynomials, probability functions and random number generators. This list is not identical to the list of functions tabulated in the NBS Handbook; among the omissions are parabolic cylinder functions,

Mathieu functions, spheroidal wave functions and the Riemann Zeta function. The UL Library is designed to be used on personal computers of IBM type, equipped with a Microsoft FORTRAN 77 compiler. For efficiency, a numeric co-processor is recommended.[1] The operating precision is IEEE double precision (53 bits in the floating-point mantissa), but the accuracy of the computed function values is generally less, sometimes well below single precision.

The UL library is an extremely ambitious project, especially as it appears that all of the programs have been constructed from scratch. For such a project to be completed successfully its authors need to have a thorough knowledge of, and experience in, several areas of classical and numerical analysis, including analytic properties of the higher transcendental functions, asymptotic analysis, approximation theory, and error and stability analyses. How well have the present authors (who are nameless) succeeded?

2. NEGATIVE RESULTS OF NUMERICAL TESTS

A comprehensive answer to the question just posed would necessitate a tremedous amount of numerical testing. We concentrated our testing on just a few functions with which we have had previous software experience, namely Airy, Bessel, hypergeometric, confluent hypergeometric and Legendre functions. Usually the UL Library performed in accordance with the specifications for each routine. However, we studied the documentation in detail, looking for major ways in which the algorithms used might go astray. Unfortunately, we were successful.

---

[1] The library diskettes provide Fortran programs, which could be compiled and used on a personal computer without a co-processor, as well as co-processor assembly code for each subroutine. For the purposes of this review, attention is restricted to the co-processor assembly code.

2. THE FIRST TYPE OF FAILURE

The first, and very serious, type of failure arises when the UL Library delivers completely wrong answers without any error message. An example occurs with the routine BESJR in §8.8, which is designed to generate the Bessel function $J_\nu(x)$ for real values of the argument x and order ν. With $x = 9\pi$ and $\nu = 2\frac{1}{2}, 4\frac{1}{2}, 6\frac{1}{2}, 8\frac{1}{2}, 10\frac{1}{2}$ BESJR yields the following values of $J_\nu(x)$, to ten decimal places:

-0.00670 05817, 0.03764 71379, -0.05418 00781, 0.31114 25305, -0.06567 89923.

These should be compared with the correct values[2]

0.01592 10880, -0.05237 32560, 0.10316 94874, -0.14727 18330, 0.14363 19977.

Not only are the numerical values totally incorrect, all the signs are wrong, too. Similar gross errors occur for $\nu = 22\frac{1}{2}(2)50\frac{1}{2}$. On the other hand, BESJR generates correct values (within the prescribed error tolerance) for the intermediate values $\nu = 12\frac{1}{2}(2)20\frac{1}{2}$, and also for $\nu = \frac{1}{2}$ and $1\frac{1}{2}(2)49\frac{1}{2}$, x again being 9π.

The reason for these errors appears to be that J. C. P. Miller's backward recurrence algorithm has been used, with the trial values normalized on the value of $J_{1/2}(9\pi)$. Since $J_{1/2}(9\pi)$ is zero, this procedure is bound to lead to meaningless answers. Yet this cannot be the entire explanation, otherwise all values in the range $\nu = 21\frac{1}{2}(1)50\frac{1}{2}$ would be incorrect and not merely alternate ones. Thus the documentation must be in error, too. And there may be further inaccuracies here. For example, according to the documentation the value for $\nu = 2\frac{1}{2}$ is computed from the power-series expansion

[2] Obtained by use of D. E. Amos' package [2]. See also comments made below on validation.

of $J_\nu(x)$: either the Miller algorithm was used instead, or, perhaps less likely, the power series was summed incorrectly.

Similar errors occur for other values of ν, both integer and noninteger. For example, with $x = 8.65372\ 79129\ \ldots$ (the third positive zero of $J_0(x)$), BESJR generates accurate values for $\nu = 0,1,2,3$ and $7(1)50$, but grossly inaccurate values for $\nu = 4,5,6$. Furthermore, erroneous values of $J_\nu(x)$ are generated when the value of x is merely moderately close to one of the critical values, the magnitudes of the errors being inversely proportional to the distances of x from the critical value.

A companion routine to BESJR is BESYR (§8.10), which is designed to generate the Bessel function $Y_\nu(x)$ for real values of x and ν. Since one of the algorithms used in BESYR draws upon values of $J_\nu(x)$, we expected—and found—some difficulties. For example, BESYR computed $Y_\nu(9\pi)$ with wrong signs and incorrect numerical values for $\nu = -50\frac{1}{2}(2)-22\frac{1}{2}$ and $-10\frac{1}{2}(2)-2\frac{1}{2}$, while the results were correct at intermediate values of ν. (This similarity of behavior with BESJR reflects the identity $Y_\nu(x) = (-)^n J_{-\nu}(x)$ when $\nu = n - \frac{1}{2}$, n being an integer.)

2.2. The Second Type of Failure

A second type of failure is the generation of results which, while not completely inaccurate, contain errors greatly in excess of the accuracy claimed in the documentation. The routine BESYR illustrates this point. The documentation claims at least 12-digit accuracy when the order ν is an integer. This claim is careless because it takes no account of the inevitable loss of relative precision in the neighborhoods of zeros. But the situation is actually much worse. With $\nu = 119$ and 120, we found that in the range

$x = 848(0.1)852$ the accuracy often fell to 8 or 9 digits, even well away from the zeros of $Y_{119}(x)$ and $Y_{120}(x)$.

Another example is provided by the routines AIRYA and AIRYAD (§§8.1 and 8.3) for the Airy function $Ai(x)$ and its first derivative. In both routines at least 13-digit accuracy is claimed when $-10^{10} \le x \le 100$. But we found many values of x in the (zero-free) range 5.2 to 5.8 for which they yielded only 8 or 9 correct digits.

2.2. The Third Type of Failure

The third type of failure arises when the UL Library generates an innaccurate value and the user is warned by an error message such as "unable to compute the . . . function with acceptable accuracy" or "numeric overflow in the . . . function". If these failures occur too frequently, then there will be huge gaps in the effective ranges claimed for the variables. Such is the case with the routines CHGFU (§9.2) for generating the confluent hypergeometric function $U(a,b,x)$, and HPRGMT (§20.1) for generating the hypergeometric function $F(a,b;c;x)$.

CHGFU employs two algorithms. The first, evaluation of the asymptotic expansion of $U(a,b,x)$ for large x, is quite sound. The second is based on a formula that expresses $U(a,b,x)$ as a difference of two M-type confluent hypergeometric functions, is unsound because of the potential for massive numerical cancellation. As a result, although the documentation claims that the effective ranges of the variables are given by

$$-50 \le a \le 50, \qquad -50 \le b \le 50, \qquad -100 \le x \le 100,$$

with the exclusion of integer values of b and nonpositive integer values of a, extensive regions are inadmissible. These include:

$$\begin{cases} a = 0.5,\ b = 0.5,\ 7.4 \le x \le 19.9; \quad a = 2.7,\ b = 5.4,\ 10.1 \le x \le 437.4; \\ a = 20,\ b = 2.5,\ 0.34 \le x \le 20{,}000. \end{cases}$$

For HPRGMT the documentation states that a and b can have any real value between -10^{10} and 10^{10}, c can have any real value between -10^{20} and 10^{20} (other than a negative integer) and x can have any real value between -10^{10} and 1. Again, these claims are misleading. For example, when x is in the interval [0,1) HPRGMT sums the hypergeometric series

$$F(a,b;c;x) = \sum_{n=0}^{\infty} \frac{a(a+1)\cdots(a+n-1)b(b+1)\cdots(b+n-1)}{c(c+1)\cdots(c+n-1)} \frac{x^n}{n!} .$$

Since the radius of convergence is 1 the algorithm must fail for values of x sufficiently close to 1. Sample intervals of failure were found to be:

$a = b = 100,$	$c = 1,$	$0.95 \le x < 1;$
$a = b = 10000$	$c = 1,$	$0.0013 \le x < 1;$
$a = b = 10^8,$	$c = 1,$	$10^{-10} \le x < 1.$

Even when failure does not occur, execution can be extremely slow. For a = b = c = 1 and z = 0.99999, 20 minutes elapsed on a 25 mhz IBM PS2 Model 80 before the answer was produced.

3. REMARKS ON THE ALGORITHMS USED

The weaknesses in the algorithms used for $J_\nu(x)$, $Y_\nu(x)$ and U(a,b,x) are avoidable. Robust software for generating these functions is already available; see, for example, [3], [8]. Instead of normalizing the trial values of $J_\nu(x)$ obtained in the Miller algorithm via a single value of $J_\nu(x)$, the identity

$$(\tfrac{1}{2}x)^{\nu} = \sum_{k=0}^{\infty} \frac{(\nu+2k)\Gamma(\nu+k)}{k!} J_{\nu+2k}(x), \quad \nu \neq 0,-1,-2,\ldots,$$

[6], eq. (9.1.87), could have been used. This is the appropriate generalization of the identity

$$1 = J_0(x) + 2J_2(x) + 2J_4(x) + \cdots$$

that the authors use in a routine BESJ (§8.7) for generating functions of integer order. A stable way of generating $U(a,b,x)$ is backward integration of the confluent hypergeometric equation, with initial values derived from the asymptotic expansions of $U(a,b,x)$ and $\partial U(a,b,x)/\partial x$ for large x.

In addition to occasional poor choices of algorithm, instances of poor choices of the actual functions being generated were observed. Thus, functions that exhibit exponential growth or decay when the argument x is large would have been better replaced by their logarithms. This would greatly increase the threshold at which overflow or underflow occurs. Such functions include the Gamma function, $\Gamma(x)$, the exponential integral, $Ei(x)$, the complementary error function, erfc x, the Airy functions, $Ai(x)$ and $Bi(x)$, the incomplete Gamma function $\Gamma(a,x)$, the modified Bessel functions $I_\nu(x)$ and $K_\nu(x)$, and the confluent hypergeometric function $M(a,b,x)$. In the case of $\Gamma(x)$, a separate routine *is* given for $\ell n|\Gamma(x)|$ in §13.3, but since this is constructed simply by taking logarithms of the values obtained by the library routine for $\Gamma(x)$, there is no increase in the overflow threshold. The logarithm should have been generated first!

Troublesome singularities can sometimes be avoided by introduction of appropriate factors. Each of the functions $M(a,c,x)$ and $F(a,b;c;x)$ has poles at $c = 0,-1,-2,\ldots$, but both $M(a,c,x)/\Gamma(c)$ and $F(a,b;c;x)/\Gamma(c)$ are

entire functions of c; this was pointed out many years ago in [7]. The statement in §9.2 that U(a,b,x) is not well defined when a and b are negative integers and $|b| \geq |a|$ is incorrect. As noted in [7], p. 258, U(a,b,x) is entire as a function of a and b. The poorly chosen algorithm used to compute U(a,b,x) may fail when a or b is an integer or close to an integer.

4. REMARKS ON VALIDATION

We could continue in this vein and we could also provide a substantial list of typographical errors in the manual and operational defects in the software.[3] Instead, let us turn now to the topic of validation. Several articles have been written on the difficulty of checking software written for the generation of mathematical functions; see, for example, [3]. How were the UL Library routines validated by the authors? The only clue supplied in the manual appears to be the statement on p. 36 that the least accurate results in the output of an algorithm always occur at the interface with another algorithm. Presumably this means that there has been substantial cross-checking at these interfaces. This is indeed a powerful type of check, but one that does not guard against all types of algorithmic and programming errors, as we have already observed with the routines BESJR and BESYR.

There are two subsections (§§6.2, 6.3) in which the authors encourage users to check output by using identities satisfied by the higher transcendental functions. For example, the error and Bessel functions are both special cases of confluent hypergeometric functions. However, these kinds of

---

[3] For example, there is a statement on p. 24 that the computer will halt when a library routine is called and a co-processor is not present. We found this is not always true; furthermore, instead of identifying the absence of a co-processor as the problem, the error messages misleadingly report errors in the library routine.

identities are rather specialized, and there is always the danger that they may have already been used in constructing the library routine. For example, there are identities that relate the Airy functions Ai(x), Bi(x) and their derivatives to Bessel functions and modified Bessel functions of orders ±1/3, ±2/3. But if these identities are used as cross-checks then the inaccuracies we noted above for the routines AIRYA and AIRYAD may not show up because the same kind of inaccuracy is present in the (parent) routine BESKR (§8.14) for the modified Bessel function $K_\nu(x)$. In contrast, a powerful form of cross-check that is *not* mentioned in the manual, but which is widely applicable, is to employ identities of Wronskian or Casoratian type. Examples are:

$$Ai(x)Bi'(x) - Ai'(x)Bi(x) = 1/\pi,$$

$$J_{\nu+1}(x)Y_\nu(x) - J_\nu(x)Y_{\nu+1}(x) = 2/(\pi x).$$

Checks of this type were used to detect errors in the UL Library routines. They were also used to validate results from other libraries used to compare against the UL Library.

5. SUMMARY AND RECOMMENDATIONS

In summary, the UL Library provides PC users with a new set of routines for generating an extensive collection of higher transcendental functions, indeed most of the functions tabulated in the NBS Handbook. The library is relatively easy to install and its price is reasonable. It will provide a useful tool for mathematical physicists and other scientists, especially those engaged in calculations of exploratory type.

However, in a comparatively small sample failures of various kinds were encountered, including some extremely serious ones. It may be that the authors lacked some experience and expertise, or perhaps just the necessary

time, for the mammoth task of constructing and testing a robust library of the kind intended. Therefore, users will need to exercise great care with any output from the library, applying independent checks wherever possible. Furthermore, users must also be prepared for disappointments: the viable ranges of a routine may turn out to be a good deal less than is claimed in the documentation, especially in the case of functions that have not been treated by earlier software workers.

For heavy systematic computations many users will find the more robust IMSL and NAG libraries [4], [5] to be preferable. The variety of functions covered in these libraries is not as large, but the viable ranges of the variables are considerably more extensive and the precision is often higher. Moreover, both IMSL and NAG provide many desirable features, such as linear algebra packages, in addition to routines for generating higher transcendental functions.

6. ACKNOWLEDGEMENT

We wish to express our appreciation to Sang Chin, a student at Duke University, who assisted with much of the numerical work required for this review.

7. REFERENCES

[1] AMERICAN NATIONAL STANDARD PROGRAMMING LANGUAGE FORTRAN, ANSI X3., 9-1978, *American National Standards Institute*, 1430 Broadway, New York, NY 10018.

[2] D. E. AMOS, S. L. DANIEL, and M. K. WESTON, "CDC 6600 Subroutines IBESS and JBESS for Bessel Functions I(x) and J(x), x > 0, ν > 0," *ACM Trans. Math. Software*, v. 3, 1977, pp. 76-92.

[3] W. J. CODY, "Software for special functions," in *Rend. Sem. Mat. Univ. Politec. Torino*, Special issue: "Special functions: theory and computa-tion", 1985, pp. 91-116.

[4] IMSL LIBRARY REFERENCE MANUAL, IMSL, 7500 Bellaire Boulevard, Houston, TX 77036-5085. Edition 9, June 1982.

[5] NAG FORTRAN MINI MANUAL - MARK 12, Numerical Algorithms Group, Inc. 1101 31st St., Suite 100, Downers Grove, IL 60515-1263. First Edition, March 1987.

[6] NATIONAL BUREAU OF STANDARDS, *Handbook of Mathematical Functions* (M. Abramowitz and I. A. Stegun eds.), Applied Mathematics Series No. 55, U.S. Government Printing Office, Washington, D.C., 1964.

[7] F. W. J. OLVER, *Asymptotics and Special Functions*, Academic Press, New York, 1974.

[8] N. M. TEMME, "The numerical computation of the confluent hypergeometric function U(a,b,z)," *Numer. Math.*, v. 41, 1983, pp. 63-82.

ST-114A
EV. 3-90)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NISTIR 4490

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
DECEMBER 1990

TITLE AND SUBTITLE

Review of Mathematical Function Library for Microsoft-FORTRAN, John Wiley & Sons, 1989

AUTHOR(S)

D. W. Lozier and F. W. J. Olver

PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

&

Institute for Physical Science & Technology
University of Maryland
College Park, MD 20742

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED
Final

SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

SUPPLEMENTARY NOTES

ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

A review of Mathematical Function Library for Microsoft-FORTRAN, John Wiley & Sons, 1989, and associated computer software is presented.

KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

book review; mathematical functions; software review; special functions of mathematical physics and statistics

AVAILABILITY

[X] UNLIMITED

[ ] FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).

[ ] ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.

[x] ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
15

15. PRICE
A02

ELECTRONIC FORM