NATIONAL BUREAU OF STANDARDS REPORT

10 695

THE NATIONAL BUREAU OF STANDARDS' LINEAR AND QUADRATIC PROGRAMMING SUBROUTINES

Technical Report

to

Computer Services Division

Center for Computer Sciences and Technology

NBS

U.S. DEPARTMENT OF COMMERCE

NATIONAL BUREAU OF STANDARDS

NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards [1] was established by an act of Congress March 3, 1901. Today, in addition to serving as the Nation's central measurement laboratory, the Bureau is a principal focal point in the Federal Government for assuring maximum application of the physical and engineering sciences to the advancement of technology in industry and commerce. To this end the Bureau conducts research and provides central national services in four broad program areas. These are: (1) basic measurements and standards, (2) materials measurements and standards, (3) technological measurements and standards, and (4) transfer of technology.

The Bureau comprises the Institute for Basic Standards, the Institute for Materials Research, the Institute for Applied Technology, the Center for Radiation Research, the Center for Computer Sciences and Technology, and the Office for Information Programs.

THE INSTITUTE FOR BASIC STANDARDS provides the central basis within the United States of a complete and consistent system of physical measurement; coordinates that system with measurement systems of other nations; and furnishes essential services leading to accurate and uniform physical measurements throughout the Nation's scientific community, industry, and commerce. The Institute consists of an Office of Measurement Services and the following technical divisions:

Applied Mathematics—Electricity—Metrology—Mechanics—Heat—Atomic and Molecular Physics—Radio Physics [2]—Radio Engineering [2]—Time and Frequency [2]—Astrophysics [2]—Cryogenics.[2]

THE INSTITUTE FOR MATERIALS RESEARCH conducts materials research leading to improved methods of measurement standards, and data on the properties of well-characterized materials needed by industry, commerce, educational institutions, and Government; develops, produces, and distributes standard reference materials; relates the physical and chemical properties of materials to their behavior and their interaction with their environments; and provides advisory and research services to other Government agencies. The Institute consists of an Office of Standard Reference Materials and the following divisions:

Analytical Chemistry—Polymers—Metallurgy—Inorganic Materials—Physical Chemistry.

THE INSTITUTE FOR APPLIED TECHNOLOGY provides technical services to promote the use of available technology and to facilitate technological innovation in industry and Government; cooperates with public and private organizations in the development of technological standards, and test methodologies; and provides advisory and research services for Federal, state, and local government agencies. The Institute consists of the following technical divisions and offices:

Engineering Standards—Weights and Measures — Invention and Innovation — Vehicle Systems Research—Product Evaluation—Building Research—Instrument Shops—Measurement Engineering—Electronic Technology—Technical Analysis.

THE CENTER FOR RADIATION RESEARCH engages in research, measurement, and application of radiation to the solution of Bureau mission problems and the problems of other agencies and institutions. The Center consists of the following divisions:

Reactor Radiation—Linac Radiation—Nuclear Radiation—Applied Radiation.

THE CENTER FOR COMPUTER SCIENCES AND TECHNOLOGY conducts research and provides technical services designed to aid Government agencies in the selection, acquisition, and effective use of automatic data processing equipment; and serves as the principal focus for the development of Federal standards for automatic data processing equipment, techniques, and computer languages. The Center consists of the following offices and divisions:

Information Processing Standards—Computer Information — Computer Services — Systems Development—Information Processing Technology.

THE OFFICE FOR INFORMATION PROGRAMS promotes optimum dissemination and accessibility of scientific information generated within NBS and other agencies of the Federal government; promotes the development of the National Standard Reference Data System and a system of information analysis centers dealing with the broader aspects of the National Measurement System, and provides appropriate services to ensure that the NBS staff has optimum accessibility to the scientific information of the world. The Office consists of the following organizational units:

Office of Standard Reference Data—Clearinghouse for Federal Scientific and Technical Information [3]—Office of Technical Information and Publications—Library—Office of Public Information—Office of International Relations.

---

[1] Headquarters and Laboratories at Gaithersburg, Maryland, unless otherwise noted; mailing address Washington, D.C. 20234.
[2] Located at Boulder, Colorado 80302.
[3] Located at 5285 Port Royal Road, Springfield, Virginia 22151.

NATIONAL BUREAU OF STANDARDS REPORT

NBS PROJECT
2053588

February, 1972

NBS REPORT
10 695

THE NATIONAL BUREAU OF STANDARDS' LINEAR AND QUADRATIC PROGRAMMING SUBROUTINES

W. G. Hall
R. H. F. Jackson
P. B. Saunders
Applied Mathematics Division

NBS

U.S. DEPARTMENT OF COMMERCE

NATIONAL BUREAU OF STANDARDS

ACKNOWLEDGEMENT

The authors would like to acknowledge the aid provided by Richard Ku of the Technical Analysis Division; not only for proofing the manuscript and providing numerous unusual test problems, but also for playing the role of "typical user" on whom we tried our ideas.

ABSTRACT

This report documents one phase of an effort to provide users, of the facility operated by the National Bureau of Standards' Computer Services Division, with reliable,well-tested,clearly-described solution algorithms for selected frequently-arising classes of special mathematical problems. The report presents algorithms for the simplex and revised simplex methods of linear programming, as well as their adaptations to quadratic programming. Set up as subroutines, the present versions of these codes use internal storage only, with resultant limitations on the size of the problems which can be treated.

Key Words: Algorithms, linear programming, nonlinear programming, quadratic programming.

CONTENTS

1. INTRODUCTION

The "package" documented in this report consists of two subroutines: SIMPLX, which is an implementation of the simplex method for linear programming, and RVSMPX, which is an implementation of the revised simplex method for linear programming. In addition, both subroutines can solve some types of quadratic programming problems.

The primary goal during the development of this package was to provide reliable user-oriented subroutines for solving linear programming problems. Other desirable program attributes, such as efficiency with respect to core usage and time, and the extension of problem size, were considered to be of less importance. An abundance of monitoring prints is available; each is optional and may be printed or suppressed independently of the other outputs. All output appears on unit 6; there are no other references to peripheral devices within the subroutines. Since each program is a subroutine, there is a return to the calling routine regardless of whether the exit is normal (a solution has been found) or abnormal (the problem is infeasible, unbounded, ill-stated, or numerically unstable). Hence the user must, in his calling routine, test a status parameter upon return from the subroutine to determine the cause of the return.

The calling statements for the subroutines are as similar as possible. In all of these, A is the augmented constraint matrix, X is the output vector, L contains the output switches, etc. In fact, in the interests of standardization, the two subroutines were designed to be as similar as possible with respect to variable names, error-handling, and output messages, without sacrificing any speed or efficiency intrinsic to the respective algorithms.

It is due to this similarity between SIMPLX and RVSMPX, and to an attempt at avoiding duplication, that the documentation here presented is in a somewhat "factored" form: sections 4, 5, and 6 contain information that is common to both subroutines and would have been duplicated had it not been "factored out" into its own section.

It is well-known [7] that the revised simplex method is faster and more efficient computationally than the original simplex method. The main reason for the inclusion of the simplex method is that since its computations are actually performed in the memory space occupied by the A matrix (the A matrix is the simplex tableau) and there is no need to store separately the inverse of the basis matrix, SIMPLX can solve somewhat larger problems than RVSMPX. This should be advantageous to the user whose problem is slightly too large for RVSMPX but not large enough to warrant the use of peripheral storage.

In the interests of completeness we mention here the rule used by RVSMPX and SIMPLX to break ties for the variable to leave the basis. The variable that is chosen to leave the basis when there is a tie is the variable whose pivot element is largest in magnitude. Although this rule does not guarantee the prevention of cycling, practical experience thus far has shown it to be effective.

This report is intended as a user's manual for RVSMPX and SIMPLX. The reader should have some familiarity with both linear and computer programming. Although some parts of the text do require more than a basic familiarity, the user will normally have no need to understand these sections.

2. THE SIMPLEX SUBROUTINE (SIMPLX)

SIMPLX finds, using the simplex method for linear programming, the maximum value of a linear objective function subject to a set of linear constraints with non-negative variables <u>and non-negative right-hand-sides</u>. The problem is:

$$\text{maximize:} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{subject to:} \quad \sum_{j=1}^{n} a_{ij} x_i \leq b_i, \quad i = 1, 2, \ldots, \ell$$

$$\sum_{j=1}^{n} a_{ij} x_i \geq b_i, \quad i = \ell + 1, \ell + 2, \ldots, \ell + g$$

$$\sum_{j=1}^{n} a_{ij} x_i = b_i, \quad i = \ell + g + 1, \ell + g + 2, \ldots \ell + g + e$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, n$$

$$b_i \geq 0, \quad i = 1, 2, \ldots, m$$

where n is the number of "real" (i.e.,original) variables; m, which is equal to $\ell + g + e$, is the total number of constraints; ℓ is the number of contraints with $\leq$ signs (LE constraints); g is the number of constraints with $\geq$ signs (GE constraints); e is the number of constraints with $=$ signs (EQ constraints). Note the requirement that all $b_i \geq 0$.

Within the SIMPLX subroutine, the constraint set is transformed into a system of equations with an initial basic feasible solution through the addition of slack variables for the LE constraints, surplus variables for the GE constraints, and artificial variables for the GE and EQ constraints. The subroutine then proceeds with the "Two-Phase Method" for solving the problem.

During Phase I, an artificial objective function is constructed in such a way that when simplex iterations have driven it to zero, the basis contains no non-zero artificial variables,and is a basic feasible solution to the original problem. However, if the maximum of the artificial objective function is not zero, there is no solution to the original problem, and that problem is declared infeasible. On the other hand, if at the end of Phase I one or more artificial variables remain in the solution set (basis) at a zero level, then the constraints associated with those artificial variables are redundant (i.e., they are not needed). These variables will remain in the basis at a zero level throughout Phase II, and will be members of the final solution. Note that this allows an arbitrary relationship between the number of constraints and the number of variables in the original problem. Phase I is not required if all the constraints in the original systems are LE constraints, since the slack variables alone provide an initial basic feasible solution in this case.

A few words need to be said about the redundancy indication mentioned above. At the "textbook" end of Phase I (all the indicators,the "$z_j - c_j$," non-positive), the NBS simplex and revised simplex subroutines go one step further. If at that point there are any artificial variables in the basis at an "effectively" zero level (see section 5.9), an attempt is made to remove them by replacing them with any non-basic, non-artificial variable whose indicator is zero. Any artificial variable removed in this way was in the basis as a result of degeneracy, and any artificial variable that remains in the basis at a zero level after this procedure, is present as a result of redundancy.

Phase II consists of simplex iterations whose task is to maximize the real (original) objective function. If no maximum exists, the problem is declared unbounded.

SIMPLX consists of approximately 400 FORTRAN V statements that compile into slightly less than 2,000 computer words. Data storage requires approximately $mn+6m+5n$ words. With a small main program on the NBS UNIVAC 1108, under the EXEC II operating system, mn may be as large as 44,000.

The calling statement is as follows:

```
CALL SIMPLX (A,MA,MT,NT,L,X,TØLP,KQP).
```

For an explanation of the parameters in the calling statement and a discussion of their meaning on entering and exiting from the subroutine, see Section 5.

For the purpose of illustration, and to give the user an idea of what the output from a run will look like, the output from an example problem is included here. The problem is:

maximize: $x_1 + 1.5x_2 + 5x_3 + 2x_4$

subject to :

$$
\begin{aligned}
3x_1 + 2x_2 + x_3 + 4x_4 &\le 6 \\
2x_1 + x_2 + 5x_3 + x_4 &\le 4 \\
2x_1 + 6x_2 - 4x_3 + 8x_4 &= 0 \\
x_1, \quad x_2, \quad x_3, \quad x_4, &\ge 0
\end{aligned}
$$

The A matrix was set up as follows:

3	2	1	4	6
2	1	5	1	4
2	6	-4	8	0
1	1.5	5	2	0
0	0	0	0	0

The other input parameters were:

MA=100 L(1)=2 L(4) through L(14)=1

MT=5 L(2)=0 TØLP=0

NT=5 L(3)=0 KQP=0

The following page contains the output from this run.

```
EPSILON =     .325000-04  CAPITAL EPSILON =     .325000-03       0 NON-ZERO ENTRIES ARE EFFECTIVELY EQUAL TO ZERO.


PHASE 1 ITERATION    1.  PIVOT=    8.00000      OBJECTIVE FUNCTION=    .00000000       X(  4) ENTERED THE BASIS, A(  1) LEFT.

BASIC VARIABLES

   X(  4)=      .000000

   S(  1)=     6.000000    S(  2)=     4.000000



******************************************************************************************************************
*                                                                                                                *
*              END OF PHASE 1.    OBJECTIVE FUNCTION =       .00000000       THERE WERE    1 ITERATIONS.         *
*              MINIMUM PIVOT WAS    8.0000     AT ITERATION    1.   REAL OBJECTIVE FUNCTION =    .00000000       *
*                                                                                                                *
******************************************************************************************************************

BASIC VARIABLES

   X(  4)=      .000000

   S(  1)=     6.000000    S(  2)=     4.000000


PHASE 2 ITERATION    1.  PIVOT=    5.50000      OBJECTIVE FUNCTION=    4.3636363       X(  3) ENTERED THE BASIS, S(  2) LEFT.

BASIC VARIABLES

   X(  3)=      .727273    X(  4)=      .363636

   S(  1)=     3.818182



******************************************************************************************************************
*                                                                                                                *
*              END OF PHASE 2.    OBJECTIVE FUNCTION =       4.3636363       THERE WERE    1 ITERATIONS.         *
*              MINIMUM PIVOT WAS    5.5000     AT ITERATION    1.                                                *
*                                                                                                                *
******************************************************************************************************************

BASIC VARIABLES

   X(  3)=      .727273    X(  4)=      .363636

   S(  1)=     3.818182
```

3. THE REVISED SIMPLEX SUBROUTINE (RVSMPX)

The formulation of the problem, restrictions pertaining thereto, and definitions of the variables are the same here as that in section 2 for SIMPLX. The method of solution is, of course, different. The method used is the revised simplex method as presented in chapter 3 of [3], with one important modification. The modification is the provision for re-inverting the basis matrix every [m/2]+5 iterations, in order to reduce round-off error. (The value [m/2]+5 is one that appears in other codes, see[1], and has been shown in practice to be a reasonably good choice. However, if the user feels another value is more appropriate, he may use his value by changing the value of INVC on card number 6 of the subroutine.) A discussion of the advantages of re-inverting the basis matrix appears in section 7-8 of [4]. This provision also allows for the "restart" or "advanced start" capability whereby the user may choose to supply the subroutine with an initial basic feasible solution, if known, in order to speed up the algorithm. For instructions on how to use this advanced start capability, see section 5.10.

RVSMPX consists of approximately 450 FORTRAN V statements that compile into slightly more than 2,000 computer words. Data storage is approximately $mn + m^2 + 2n + 3m$ computer words.

The calling statement is:

CALL RVSMPX (A,MA,B,MB,MT,NT,L,X,TØLP,INV,KQP).

For an explanation of the parameters in this calling statement and a description of their values on entering and exiting from the subroutines, see section 5.

The output messages that may appear are described in section 6.

4. SOLVING QUADRATIC PROGRAMS.

As was mentioned in the introduction, both RVSMPX and SIMPLX have the capability to solve quadratic programming problems. This is accomplished via the parameter KQP, discussed below in section 5.

A quadratic program is of the following form:

maximize: $$\sum_{j=1}^{n} c_j x_j + \sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_i x_j$$

subject to the same set of constraints that appears in section 2. There is, however, one important additional restriction--the objective function above must be concave. If it is not concave, although a feasible solution to the quadratic program may be found, there is no guarantee that this solution will be optimal.

The objective function above is concave if

$$\sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_i x_j \leq 0,$$

for all x_i, where i=1,2,...,n. This is equivalent to saying that the matrix of the d_{ij} is negative semi-definite. The quadratic form above may be shown to be concave by showing that it can be written as the negative of a sum of linear forms,

$$\sum_{i=1}^{n} \sum_{j=1}^{n} d_{ij} x_i x_j = -\sum_k [L_k(x)]^2 \quad ;$$

in many applications (e.g., least squares estimation) this is known *a priori*. Equivalently, one could attempt to diagonalize the matrix of the d_{ij} and check the resulting diagonal elements (the eigenvalues of the d_{ij} matrix). If all are non-positive, the matrix is negative semi-definite. See chapter 8 of [2] for more on this.

The method of solution that is used was originally developed by Wolfe [9]. He noted that the Kuhn-Tucker conditions (see Appendix A and [6]) for the quadratic programming problem are "almost linear", and that, under the concavity assumption mentioned above, they are necessary and sufficient--which means that if one can find a set of x_j, j=1,2...n, that satisfy the Kuhn-Tucker conditions, then those x_j are the optimal x_j for the original problem. Thus, the emphasis was shifted to solving a set of "almost linear" equations. This was accomplished in [9] by utilizing Phase I of the simplex method for linear programming with one modification. That modification is that certain pairs of variables cannot simultaneously be in the basis; it is implemented by restricting the choice of the entering basic variable. This approach is used in both RVSMPX and SIMPLX.

All that is necessary to solve a quadratic program is to set KQP to n, where n is as defined in this section, and to set up the constraint matrix as follows:

A_1	0	0	0	0	0	
A_2	0	0	0	0	0	b
A_3	0	0	0	0	0	
$-2D$	A_1^T	$-A_2^T$	A_3^T	$-A_3^T$	$-I_n$	c
0	0	0	0	0	0	*
*	*	*	*	*	*	*

where the original constraint matrix has been partitioned into A_1, A_2 and A_3, consisting of the LE, GE, and EQ constraint coefficients respectively. A^T indicates the transpose of the matrix A, D is the matrix of the d_{ij}, and I_n is an nxn identity matrix. The solution to the quadratic programming problem will be found in X(1) through X(N) where N is the number of real variables in the original problem.

For a more rigorous development of the algorithm and an explanation of some of the information that might be gleaned from the solution, see appendix A.

5. THE PARAMETERS.

Listed below are the parameters that appear in either one or both of the subroutines. After the name of each variable appears an (R) if the variable is a parameter of RVSMPX; an (S) if the variable is a parameter of SIMPLX; and an (RS) if the variable is a parameter of both RVSMPX and SIMPLX. All variables conform to the FORTRAN implicit naming conventions regarding mode.

1. A (RS)

A is the constraint matrix augmented by a right-hand-side column, an objective function row, and in the case of SIMPLX, a row to be used (by the subroutine) for the coefficients of the artificial objective function. The first ℓ rows of A contain the LE constraint coefficients, the next g rows contain the GE constraint coefficients, and the next e rows contain the EQ constraint coefficients. Row m+1 contains the coefficients of the real objective function; row m+2, whose values are not supplied by the user, contains the artificial objective function coefficients for SIMPLX, and column n+1 contains the non-negative right-hand-sides. Hence, A must be dimensioned at least (m+1)x(n+1) for RVSMPX and (m+2)x(n+1) for SIMPLX. Pictorially, where "*" indicates a value that is not supplied by the user, A is:

$\frac{a_{11}}{}$	$\frac{a_{12}}{}$	.	.	.	$\frac{a_{1n}}{}$	$\frac{b_1}{}$
$\frac{a_{\ell+1,1}}{}$	$\frac{a_{\ell+1,2}}{}$	.	.	.	$\frac{a_{\ell+1,n}}{}$	$\frac{b_{\ell+1}}{}$
$\frac{a_{\ell+g+1,1}}{}$	$\frac{a_{\ell+g+1,2}}{}$	.	.	.	$\frac{a_{\ell+g+1,n}}{}$	$\frac{b_{\ell+g+1}}{}$
a_{m1}	a_{m2}	.	.	.	a_{mn}	b_m
c_1	c_2	.	.	.	c_n	*
*	*	.	.	.	*	*

We note, again, that the last row is not needed for RVSMPX.

Since the A matrix is used by SIMPLX as explicit storage for the condensed simplex tableau (see pp. 119 of [8]), the original values in the A matrix will have been completely destroyed by the time SIMPLX returns to the main program. In most cases there is nothing in the A matrix of value to the user at this point. For RVSMPX, however, the A matrix is completely unchanged during the course of the algorithm.

2. <u>MA</u> (RS)

This variable contains the value of the first dimension of A, as A is dimensioned in the calling routine; e.g., if A is dimensioned as A(20,50), then MA should be set to 20 in the main program before the subroutine is called.

This parameter is unchanged by either of the subroutines.

3. B (R)

This parameter is a matrix whose dimensions must be at least (m+2)x(m+2). Upon exiting from RVSMPX, B will contain the inverse of the matrix of the current basic columns in ((B(I,J),J=1,M),I=1,M); the negatives of the current values of the dual variables in (B(M+1,J),J=1,M); the current values of the basic variables in (B(I,M+2),I=1,M), in an order specified by a portion of the L vector to be explained later; the negative of the value of the real objective function in B(M+1,M+2); and the negative of the value of the artificial objective function in B(M+2,M+2).

4. MB (R)

This contains the value of the first dimension of the B matrix as it is dimensioned in the calling routine. It too is unchanged by the subroutine.

5. MT (RS)

MT is the number of rows of information in the A matrix, which is m+2 for SIMPLX and m+1 for RVSMPX. Note that MT is less than or equal to MA. This variable is not changed by the subroutines.

6. NT (RS)

The value of this variable is the number of columns in A, i.e., n+1. It too is unchanged by SIMPLX or RVSMPX.

7. L (RS)

L is a multipurpose vector that provides storage space for a number of distinct working vectors constructed by the subroutines, in addition to providing indicators and constants for both entry and exit. It also contains output switches which are used to determine how much (if any) output from the subroutines is given. For RVSMPX, L must be dimensioned at least 14+2m+n+g; for SIMPLX, 14+2(m+n)+g. The meaning of L(1) through L(14) is the same for both subroutines for both entry and exit; the differences in the use of the L vector by RVSMPLX and SIMPLX occur beyond L(14). Although the typical user need not understand what happens to this latter portion of the L vector, a description is included here for the user who may wish to modify the subroutines.

L(1): For entry, this must contain the number of LE constraints, or ℓ. The exit value of L(1) is the number of non-zero elements in the original A matrix that are between (-EPS) and EPS and are, therefore, considered by the subroutines to be equal to zero [see 9 below]. If the exit value of L(1) is positive, the user is providing the subroutine with information it does not use. If the exit value of L(1) is large, it is suggested that the user either force the subroutine to use the enumerated small matrix entries by making EPS smaller (see 9 below), or else not provide these numbers (set them to zero). If one of these actions is not taken, numerical problems might arise.

L(2): This should contain the number of GE constraints, or g, when entering the subroutine. The exit value of this variable is the total number of iterations the subroutine performed before termination. It includes the iterations for both phases.

L(3): When entering the subroutines, this is a default print switch. If L(3) is non-zero, the subroutine sets L(4), L(5), L(8), L(12), L(13), and L(14) to 1, and L(6), L(7), L(9), L(10), and L(11) to 0. If L(3) is zero, the user sets L(4) through L(14) to obtain the desired outputs. When leaving the subroutine, the value of L(3) is very important -- it indicates what caused the termination. A value of 0 indicates an optimal basic feasible solution has been obtained; 1 indicates that an optimal basic feasible solution has been obtained but that numerical difficulties arose during the course of the algorithm, and that the user should question the results; 2 indicates that the problem is infeasible; 3 indicates the problem is unbounded; and 4 indicates a system error.

<u>L(4)</u>: If this parameter is non-zero when the subroutines are entered, the values of EPS, CEPS, and L(1) are printed. The value of L(4) remains unchanged by the subroutine.

<u>L(5)</u>: For entry, if L(5) is non-zero, a warning message will be printed if necessary. This variable, too, is untouched by the subroutine.

<u>L(6)</u>: If L(6) is greater than zero when entering RVSMPX or SIMPLX, the iteration summary will be printed after every L(6) iterations in Phase I. Neither of the subroutines changes the value of L(6).

<u>L(7)</u>: If L(7) is greater than zero when entering the subroutines, the basic variables and their values will be printed after every L(7) iterations in Phase I. Neither RVSMPX nor SIMPLX changes this variable.

<u>L(8)</u>: If this switch is non-zero when the subroutines are entered, the final summary for Phase I will be printed. No change occurs in the value of this switch.

<u>L(9)</u>: This switch, if non-zero when entry occurs, will cause the basic variables and their values to be printed at the end of Phase I. Again, no change occurs in the value of this variable.

<u>L(10) through L(13)</u>: These variables perform the same task as L(6) through L(9), except that they refer to Phase II.

<u>L(14)</u>: When entering the subroutine, if L(14) is non-zero, error messages will be printed if errors occur. No change in the value of L(14) occurs.

Values for the remaining portions of the L vector are not supplied by the user, and, as was mentioned earier, the exit values of these variables will probably not be of any importance to the casual user of these subroutines. Nevertheless, in the interests of completeness, their descriptions are provided here.

RVSMPX and SIMPLX use this storage space differently. The description of the manner in which RVSMPX uses this space is provided first. To facilitate the discussion, let

$$
\begin{aligned}
M &= m \\
N &= n \\
MG &= g \\
L1ØFF &= 14 \\
L2ØFF &= L1ØFF + n + \ell + g + g + e \\
&= 14 + n + m + g
\end{aligned}
$$

Then, for J = 1 through N + M + MG, if L(L1ØFF + J) equals 0, X(J) is a real variable; if 1, X(J) is a slack variable; if 2, X(J) is a surplus variable; if 3, X(J) is an artifical variable; if 4, X(J) is an artificial variable that has been removed from the basis, and, therefore, from future consideration (see p. 119 of [4]); if 5, X(J) is an artifical variable that is in the basis at a zero level and cannot be removed. For J = 1 through M, if L(L2ØFF + J) = K, X(K) isthe J^{th} basic variable, and its value appears in B(J,M + 2).

To facilitate the discussion of the manner in which SIMPLX uses this space, let

L1ØFF = 14
L2ØFF = L1ØFF + n = 14 + n,
L3ØFF = L2ØFF + m = 14 + n + m
INØFF = L3ØFF + g = 14 + n+ m + g,
NØTØFF = INØFF + m = 14 + n + m + g + m
L10 = the number of columns in the A matrix after the artificial variables have been removed by the subroutine,
L20 = the number of rows in the A matrix after redundant constrain- have been implicitly removed by the subroutine.

Then, for J = 1 through L10, if L(L1ØFF + J) = K, the J^{th} column of the current A matrix is the K^{th} column of the original A matrix. Also, for J = 1 through L20, if L(L2ØFF + J) = K, the J^{th} row of the current A matrix is the K^{th} row of the original A matrix. Since the surplus and artificial variables for each GE constraint are never needed at the same time, L(L3ØFF + J) for J = 1 through MG is set up as follows: if L(L3ØFF + J) equals 1, the artificial variable for the J^{th} GE constraint is still in the basis; if L(L3ØFF + J) equals 0, the artificial variable for the J^{th} GE constraint has been removed from the basis, and the storage space previously occupied by it is occupied by the surplus variable for that constraint. For J = 1 through M, if L(INØFF + J) = K, the J^{th} basic variable is X(K). Finally, for J = 1 through N, if L(NØTØFF + J) = K, the J^{th} non-basic variable is X(K).

8. X (RS)

This vector is the solution vector. Its dimension should be at least $n + \ell + g + g + e + 1$, or equivalently, $n + m + g + 1$. For the SIMPLX subroutines, the values in the X vector when entry occurs have no effect on the subroutine, since SIMPLX initializes X to zero. For RVSMPX, however, the initial values in the X vector are important when the advanced start option (see 10 below) is used, and unimportant otherwise.

When termination occurs, X contains the terminal values of the variables in the original problem, including any slack, surplus, or artificial variables that were added by the subroutine. The first n elements of X contain the values of x_1 through x_n . Next in order are the slack variables for the LE constraints, then the surplus variables for the GE constraints, the artificial variables for the GE constraints, the artificial variables for the EQ constraints, and finally, in X(N + ML + 2*MG + ME + 1) , the value of the objective function when termination occurs.

The scheme by which values are stored in X is as follows: for each x_i that is not in the basis, X(I) = -EPS/100; hence, for each I for which X(I) is non-negative, x_i is in the basis at its given level. This scheme was employed so as to permit both easy identification of the basic variables and immediate use of the solution vector at the same time, with no undue round-off error.

X will contain these values according to the scheme noted above regardless of the reason for termination. Of course, if termination is with L(3) greater than 1, all the values in X will be the current ones and not necessarily optimal or even feasible.

9. TØLP (RS)

This is a parameter which is used in the construction of epsilon (EPS) and capital epsilon (CEPS), variables that are used as tolerance parameters. Within the subroutines, $|a|<\text{EPS}$ is equivalent to a=0. The user has the following options regarding the value of EPS:

If TØLP is less than zero, $\text{EPS} = \dfrac{|\text{TØLP}| \sum_i \sum_j |a_{i,j}|}{m * n}$

If TØLP equals 0, EPS = $\dfrac{10^{-5}\,\Sigma_{i,j}|a_{i,j}|}{m * n}$.

If TØLP is greater than 0, EPS = TØLP.

Tests have shown the default choice (TØLP=0) to be reasonably effective. The word "reasonably" is used here since there are cases when this choice is inappropriate. It is for this reason that the two other choices for TØLP and EPS have been provided. Furthermore, to aid the user in determining when an inappropriate value of EPS has been selected, warning messages have been provided (see section 6) that indicate that numerical problems have arisen. This may possibly be overcome by changing the value of EPS.

In any case, CEPS is set to 10*EPS and is used solely to determine when a number is "close to zero".

The value of TØLP is not changed within either of the subroutines.

10. <u>INV</u> (R)

This is a switch that allows the user to provide RVSMPX with an initial basic feasible solution. If INV is non-zero, an initial basic feasible solution will be expected. If INV equals zero, the subroutine will start from scratch with Phase I.

The initial basis is passed via the X vector as follows: for i=1 through $n + \ell + g + g + e$, if x_i is a member of the basis, then X(I) is greater than or equal to zero; if x_i is not a member of the basis, then X(I) is less than zero. The actual values are irrelevent so long as the signs are correct. RVSMPX will form a basis with these variables and proceed with Phase I or Phase II, depending on whether or not the basis contains any artificial variables.

When RVSMPX terminates, INV may have a value of 0,1, or 2. In any event, its value is irrelevant.

11. <u>KQP</u> (RS)

This is a switch that allows the user to solve quadratic programming problems. If KQP is zero, the subroutines will consider the problem being solved to be a linear programming problem. In order to solve a quadratic programming problem, KQP must be set to n, the number of real variables in the original quadratic programming problem. For a discussion of the quadratic programming algorithm that is used, the method of setting up the problem, and the interpretation of the results, see section 4.

The value of this switch remains unchanged throughout the subroutine.

6. OUTPUT

As indicated in section 5.7, the output messages are controlled (with one exception) by the values of L(3) through L(14), which are parameters in the subroutine call. This section dicusses the output messages that may result from the use of RVSMPX or SIMPLX. Since there are a few messages that are indigenous to one or the other of the two subroutines, the same notational scheme that was used in section 5 will be used here; i.e., if the message can result from RVSMPX, an (R) will appear after the name of the message; if the message can result from SIMPLX, an (S) will appear after the name of the message; and if the message can result from either one, an (RS) will appear.

1. The Negative Right Hand Side Message (RS).

```
* *  AT LEAST ONE ELEMENT OF THE RIGHT HAND SIDE COLUMN IS LESS THAN ZERO. SUBROUTINE TERMINATES.
```

This message is exceptional in that it is not under control of any switch in the L vector. Immediately upon entry, the subroutines examine the problem data. If any entry in column NT of matrix A (i.e., the right hand sides) is negative, this message is printed, the value of L(3) becomes 2, and the subroutine returns to the calling routine. The problem should be reformulated so that all right hand sides are non-negative.

2. The Epsilon Print (RS).

```
EPSILON =    .964732-02  CAPITAL EPSILON =    .964732-01        0 NON-ZERO ENTRIES ARE EFFECTIVELY EQUAL TO ZERO.
```

This message is printed whenever L(4) is non-zero. The values of epsilon and capital epsilon are based on TØLP, a parameter in the calling sequence (see section 5.9). Major ways in which these values are used are discussed in 5, 9, and 11 below, and in the discussion of L(1) in section 5.7. The number of non-zero entries in the A matrix that are "effectively" equal to zero is stored in L(1). For a discussion of this value see section 5.7.

3. <u>The "Infeasible Initial Basis" Message (R).</u>

```
*******************************************************************************************************
*                                                                                                     *
*  ERROR - THE VARIABLES SPECIFIED AS COMPRISING AN INITIAL SOLUTION DO NOT FORM A BASIC FEASIBLE SOLUTION.  *
*                                                                                                     *
*  THE ERROR WAS DETECTED AT ITERATION    0 OF PHASE 1. AT THAT TIME THE OBJECTIVE FUNCTION VALUE WAS  -21.000000  *
*  AND THE FOLLOWING VARIABLES WERE BASIC.                                                            *
*                                                                                                     *
*******************************************************************************************************

BASIC VARIABLES

  X(  6)=    2.000000    X(  8)=    1.000000    X( 10)=    3.000000    X( 12)=    2.000000    X( 14)=    6.000000
  S(  2)=    5.000000
  A(  5)=   11.000000    A(  6)=   10.000000
```

If the advanced start feature is used (INV non-zero), and the variables specified in the X vector as forming a basis do not form a feasible basis, this message is printed, (if L(14) is non-zero), L(3) is set to 2, and RVSMPX returns to the calling routine. The basic variables that are printed are the variables that had been put into the basis before the error was detected.

There are two ways in which this error may occur:

(i) The number of non-negative variables passed in the X vector is not equal to m. However, in some cases when the number is greater than m, no error will occur. That is, if the m variables that are the first to enter the basis *do* form a feasible basis, then they are used as the basis and no error is recorded.

(ii) The number of non-negative variables passed in the X vector is equal to m, but these variables do not form a feasible basis. In this case, at least one of the m basic variables printed will be negative.

(Note: if a case (i) error occurs, there may be negative basic variables. There is no cause for alarm; the basis forming procedure (actually a matrix inversion) allows basic variables to become negative during its course since, if the variables do form a feasible basis all the variables will be non-negative at the end of this procedure.)

4. The "Small Pivot" Message (RS):

```
* * WARNING * *  SMALL PIVOT ELEMENT AT ITERATION  10 OF PHASE 1. PIVOT =   .12345688-06
```

This message is printed if the chosen pivot element is between EPS and CEPS and if L(5) is non-zero. This is considered a warning message; computations continue, but the user is strongly advised to view the results critically.

5. The alternate Pivot Element Message (S):

```
* * WARNING * *  IN PHASE 1, THERE HAVE BEEN 10 ATTEMPTS TO FIND AN ALTERNATE PIVOT ELEMENT.
                 OF THESE, 6 FOUND NO ALTERNATE ELEMENT GREATER THAN CAPITAL EPSILON.
```

This warning message requires a rather detailed explanation. If the pivot element that is chosen using the standard simplex criterion is "close to zero" (between EPS and CEPS), SIMPLX goes into the alternate pivot selection routine. This routine scans the A matrix in ascending order of the column number, checking the non-basic variables (including the one chosen by the standard simplex criterion, if necessary) for one which has a pivot element greater than CEPS. If one is found, it is used (without scanning the remaining columns) in lieu of the one found by the standard simplex criterion. If none greater than CEPS is found, the largest one is kept. (This may be the one originally found by standard simplex.) If that one is between EPS and CEPS, it is used, and if L(5) is non-zero, the "small pivot" message (number 4 above) is printed. If the largest is less than EPS, the subroutine returns to the column that contained the standard simplex pivot, and sets that pivot element to zero. That column is then scanned and quotients are again formed to get a new minimum quotient, thus determining a new pivot candidate. If this new pivot candidate is acceptable (greater than EPS), Simplex continues and performs the operation. If it is not acceptable, it too is set to zero and this latter process is repeated until either an acceptable pivot element is found or until each element in that column is non-positive and the problem is declared unbounded.

As for the warning message itself, it appears at the end of a phase and gives the number of times the subroutines left the standard simplex pivot and went to the alternate pivot selection routine, and the number of times it returned to the standard simplex pivot from the alternate routine without having found an acceptable pivot.

6. The Iteration Summary (RS):

```
E 1 ITERATION  16.  PIVOT=   14.7880      OBJECTIVE FUNCTION=   .27186017-04   S(  4) ENTERED THE BASIS, A( 16) LEFT.
```

This message is printed after every L(6)th iteration in Phase I if L(6) is positive, and after every L(10)th iteration in Phase II if L(10) is positive. It is useful in tracing the progress of the algorithm.

7. <u>The Basis Print (RS)</u>.

```
BASIC VARIABLES

 X(  1)=     .821361   X(  2)=     .367047   X(  3)=     .002276   X(  4)=     .011932   X(  5)=     .696950
 X(  6)=     .731610   X(  7)=     .607440   X(  8)=     .684551   X(  9)=     .605376   X( 10)=     .129697

 S(  1)=    5.951955   S(  2)=     .234925   S(  3)=   13.179356   S(  4)=     .000000

 A(  5)=     .000000   A(  7)=     .000000   A(  9)=     .000000   A( 10)=     .000000   A( 15)=     .000000
 A( 16)=     .000000
```

This print occurs after every L(7)th iteration in Phase I if L(7) is positive, after every L(11)th iteration in Phase II if L(11) is positive, at the end of Phase I if L(9) is non-zero, and at the end of Phase II if L(13) is non-zero. It is also printed if the problem is determined to be infeasible or unbounded and L(14) is non-zero.

This print simply gives the indices and values of the basic variables. The letter <u>X</u> refers to real variables with indices between 1 and n. The letter <u>S</u> refers to slack or surplus variables with indices between 1 and ℓ + g. The letter <u>A</u> refers to artifical variables, with indices between 1 and g + e.

8. <u>The End-of-Phase Summary (RS)</u>.

```
**************************************************************************************************************
*                                                                                                            *
*        END OF PHASE 1.    OBJECTIVE FUNCTION =      .27136017-04    THERE WERE  17 ITERATIONS.             *
*        MINIMUM PIVOT WAS    .20277     AT ITERATION  12.   REAL OBJECTIVE FUNCTION =  -24.318438           *
*                                                                                                            *
**************************************************************************************************************
```

This message is printed at the end of Phase I if L(8) is non-zero, and at the end of Phase II if L(12) is non-zero. The phrase "REAL OBJECTIVE FUNCTION =" is printed only at the end of Phase I, since at that point the objective function is the artificial objective function.

9. <u>The Infeasible Message (RS)</u>.

```
*************************************************************************************************************************
*                                                                                                                       *
*  ERROR - THE PROBLEM IS INFEASIBLE. THE CONSTRAINTS ASSOCIATED WITH THE ARTIFICIAL VARIABLES BELOW ARE INCONSISTENT.  *
*  IF NONE APPEAR, NUMERICAL DIFFICULTIES HAVE BEEN ENCOUNTERED. THE LARGEST ENTRY IN THE OBJECTIVE FUNCTION ROW        *
*  IS   -5.3214                                                                                                         *
*                                                                                                                       *
*  THE ERROR WAS DETECTED AT ITERATION  10 OF PHASE 1. AT THAT TIME THE OBJECTIVE FUNCTION VALUE WAS  -21.000000        *
*  AND THE FOLLOWING VARIABLES WERE BASIC.                                                                              *
*                                                                                                                       *
*************************************************************************************************************************
```

This is printed when the problem is infeasible and L(14) is non-zero. A problem is said to be infeasible when it is impossible to satisfy one or more constraints. It is detected in the subroutine when all the indicator variables (the $z_j - c_j$) are less than EPS and either one or both of the following is true: The value of the artificial objective function is greater than (g + e)*CEPS, or one or more artificial variables which cannot be made non-basic have value greater than CEPS.

If the maximum value in the objective function row (the indicators) is "close to " EPS and/or the objective function is "close to" CEPS and/or the artificial variables in the basis are "close to" CEPS, then the infeasibility may result from numerical instability. In this case, it may be possible to "achieve" feasibility by changing the value of EPS (see section 5.9). It should be noted, however, that a different value of EPS could cause the algorithm to follow a different path toward the solution. Thus it is not obvious what the direction and magnitude of the change should be.

If this message is printed, it is always followed by a print of the variables which are basic at the time the error was detected.

Whether or not this message is printed, L(3) is set to 2 and the subroutine returns to the calling routine.

10. <u>The "Re-inversion Infeasibility" Message (R)</u>.

```
*************************************************************************************************************
*                                                                                                           *
*  ERROR - THE PROBLEM IS INFEASIBLE.  INFEASIBILITY INDICATED DURING RE-INVERSION OF THE BASIS MATRIX.     *
*                                                                                                           *
*  THE ERROR WAS DETECTED AT ITERATION  10 OF PHASE 1.  AT THAT TIME THE OBJECTIVE FUNCTION VALUE WAS  -21.000000  *
*  AND THE FOLLOWING VARIABLES WERE BASIC.                                                                  *
*                                                                                                           *
*************************************************************************************************************
```

This message is printed, when L(14) is non-zero, if one or more basic variables became negative as a result of re-inverting the basis matrix. This error, in every case, is a result of numerical difficulties and is an indication that the problem, as it is posed, is just not numerically tractable. To overcome this problem, on might try changing the value of EPS (see section 5.9) or even INVC (see section 3); however, it is suggested that the user modify his problem to make it more numerically stable. Techniques such as normalization, scaling values, removing redundancies, etc. can be used to accomplish this.

11. <u>The "Unbounded" Message (RS)</u>.

```
*************************************************************************************************************
*                                                                                                           *
*  ERROR - THE PROBLEM IS UNBOUNDED. THE VARIABLE X( 13) CAN ASSUME AN ARBITRARILY LARGE VALUE, THEREBY YIELDING  *
*  AN ARBITRARILY LARGE VALUE OF THE OBJECTIVE FUNCTION.                                                    *
*                                                                                                           *
*  THE ERROR WAS DETECTED AT ITERATION    4 OF PHASE 2. AT THAT TIME THE OBJECTIVE FUNCTION VALUE WAS    43138.559  *
*  AND THE FOLLOWING VARIABLES WERE BASIC.                                                                  *
*                                                                                                           *
*************************************************************************************************************
```

This error message is printed, when L(14) is non-zero, if the problem is found to be unbounded. A problem is unbounded when at least one variable, together with the objective function, can assume an arbitrarily large value without violating any of the constraints. Unboundedness is detected in the subroutines when all the entries in the column associated with the variable that is about to enter the basis are less than EPS. It is this variable that is identified as "causing" the unboundedness.

In some problems (those that are numerically unstable), this situation might be alleviated by changing the value of EPS (see section 5.9). However, as was mentioned in 9 above, the direction and magnitude of the required change are not obvious.

If this message is printed, it is always followed by a print of the variables that were in the basis at the time the error was detected.

Whether or not this message is printed, L(3) is set to 3 and the subroutine returns to the calling routine.

12. <u>The "Computational Inconsistency" Message (RS)</u>.

```
*****************************************************************************************************
*                                                                                                   *
*  WARNING - COMPUTATIONAL INCONSISTENCY INDICATED AT THE END OF PHASE 1. THE ALGORITHM WILL CONTINUE WITH PHASE 2,  *
*  BUT THE USER IS ADVISED TO CRITICIZE THE RESULTS.                                                *
*                                                                                                   *
*****************************************************************************************************
```

If L(14) is non-zero, this message is printed at the end of Phase I if one or more artificial variables which cannot be removed from the basis is between EPS and CEPS, or if the artificial objective function is between EPS and CEPS. This situation is an indication of possible numerical difficulties. The user is advised at least to check the solution for feasibility.

7. ADDITIONAL WORK AND FUTURE PLANS

The subroutines described in this report are part of a longer-term effort, in which the goal of the Applied Mathematics Division's Operations Research Section is to provide reliable user-oriented solution algorithms for those mathematical optimization problems which are of particular importance for operations-research applications (they often arise in other contexts as well). In addition to SIMPLX and RVSMPX, we also have operational on the NBS computer:

(a) a 0-1 integer programming routine acquired from the RAND Corporation;

(b) a subroutine to solve transportation problems, obtained from the Carnegie-Mellon University;

(c) a prototype code implementing the Dantzig-Wolf decomposition principle for linear programming;

(d) a quadratic programming algorithm, limited to "separable" objective functions (no cross-products of variables), that solves larger problems than SIMPLX or RVSMPX,

(e) a univariate dynamic programming algorithm, developed at Johns Hopkins University and supported by the National Bureau of Standards under Contract CST-1279,

(f) a multivariate dynamic programming algorithm, obtained from Johns Hopkins University , and

(g) several algorithms for finding shortest paths between pairs of nodes in networks.

Except perhaps for (e) and (g), these items have not been tested or documented to the standards set for SIMPLX and RVSMPX. Future work, in addition to such "completion" efforts for selected items from the above list, may include:

(A) testing of the new UNIVAC mathematical programming package (Functional Mathematical Programming System),

(B) as appropriate, extension of preceding items (quite likely beginning with RVSMPX) to handle much larger problems through use of peripheral storage,

(C) provision of capability for parametric and sensitivity analyses,

(D) extension of (d), above, to more general separable nonlinear problems, and

(E) provision of a mixed-integer programming capability.

8. REFERENCES

1. Clasen, R. "Using Linear Programming as a Simplex Subroutine," Rand Report P -3267, Nov. 1965.

2. Finkbeiner, D.T., Introduction to Matrices and Linear Transformations, San Francisco, W.H. Freeman, 1960.

3. Garvin, W.W., Introduction to Linear Programming, New York, McGraw-Hill, 1960.

4. Hadley, G., Linear Programming, Reading, Mass., Addison-Wesley, 1962.

5. Hadley, G., Nonlinear and Dynamic Programming, Reading, Mass., Addison-Wesley, 1964.

6. Kuhn, H.W. and A.W. Tucker, "Nonlinear Programming", in Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, J. Neyman, ed., Berkeley, Cal., University of Cal. Press, 1951, pp. 481-492.

7. Wagner, H.M., "A Comparison of the Original and Revised Simplex Methods," Operations Research, 5(3), 1957.

8. Wagner, H.M., Principles of Operations Research, Englewood Cliffs, N.J., Prentice Hall, 1969.

9. Wolfe,P., "The Simplex Method for Quadratic Programming," Econometrica, 27, 1959, pp. 382-298.

APPENDIX A: THE "ALMOST LINEAR" KUHN-TUCKER CONDITIONS FOR QUADRATIC PROGRAMMING

One form (see section 6-3 of [4]) of the Kuhn-Tucker necessary conditions for the quadratic programming problem given in section 4 to have a solution at x_j ($j = 1$ through n) is:

1. $c_j + 2\sum_{k=1}^{n} d_{jk}x_k - \sum_{i=1}^{m} \lambda_i a_{ij} = 0$, if $x_j > 0$, for $j = 1$ through n

   $c_j + 2\sum_{k=1}^{n} d_{jk}x_k - \sum_{i=1}^{m} \lambda_i a_{ij} \leq 0$, if $x_j = 0$, for $j = 1$ through n

2. $\sum_{j=1}^{n} a_{ij}x_j = b_i$, if $\lambda_i > 0$, $i = 1$ through ℓ

   $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$, if $\lambda_i = 0$, $i = 1$ through ℓ

   $\sum_{j=1}^{n} a_{ij}x_j = b_i$, if $\lambda_i < 0$, $i = \ell + 1$ through $\ell + g$

   $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$, if $\lambda_i = 0$, $i = \ell + 1$ through $\ell + g$

   $\sum_{j=1}^{n} a_{ij}x_j = b_i$, $i = \ell + g + 1$ through m

3. $\lambda_i \geq 0$, $i = 1$ through ℓ

   $\lambda_i \leq 0$, $i = \ell + 1$ through $\ell + g$

   λ_i unrestricted, $i = \ell + g + 1$ through m

   $x_j \geq 0$, $j = 1$ through n

As was noted earlier, these conditions are also sufficient when the objective function is concave. Therefore, the problem reduces to one of finding x_j ($j = 1$ through n) that satisfy conditions 1-3 above for some set of λ_i ($i = 1$ through m).

These condtions may be written as:

1'. $c_j + 2\sum_{k=1}^{n} d_{ik}x_k - \sum_{i=1}^{m} \lambda_i a_{ij} + y_i = 0$, $j = 1$ through n

$x_j y_j = 0$, $j = 1$ through n

2'. $\sum_{j=1}^{n} a_{ij}x_j + s_i = b_i$, $i = 1$ through ℓ

$\sum_{j=1}^{n} a_{ij}x_j - s_i = b_i$, $i = \ell + 1$ through $\ell + g$

$\sum_{j=1}^{n} a_{ij}x_j = b_i$, $i = \ell + g + 1$ through m

$\lambda_i s_i = 0$, $i = 1$ through $\ell + g$

3'. $\lambda_i \geq 0$, $i = 1$ through ℓ

$\lambda_i \leq 0$, $i = \ell + 1$ through $\ell + g$

λ_i unrestricted, $i = \ell + g + 1$ through m

$x_j \geq 0$, $j = 1$ through n

$y_j \geq 0$, $j = 1$ through n

$s_i \geq 0$, $i = 1$ through $\ell + g$

With the exception of the requirements $x_j y_j = 0$ and $\lambda_i s_i = 0$, these conditions form a set of simultaneous linear equations. Futhermore, the variables either are, or can be made to be, non-negative. In order to make all the variables non-negative, substitute $\lambda_i = y_i$, $i = 1$ through ℓ; $\lambda_i = -u_i$, $i = \ell + 1$ through $\ell + g$; $\lambda_i = u_i - v_i$, $i = \ell + g + 1$ through m, into (1')-(3') to get:

1". $$c_j + 2\sum_{k=1}^{n} d_{jk} x_k - \sum_{i=1}^{\ell} u_i a_{ij} + \sum_{i=\ell+1}^{\ell+g} u_i a_{ij}$$

$$- \sum_{i=\ell+g+1}^{m} u_i a_{ij} + \sum_{i=\ell+g+1}^{m} v_i a_{ij} + y_j = 0, \quad j = 1 \text{ through } n$$

2". $$\sum_{j=1}^{n} a_{ij} x_j + s_i = b_i, \quad i = 1 \text{ through } \ell$$

$$\sum_{j=1}^{n} a_{ij} x_j - s_i = b_i, \quad i = \ell + 1 \text{ through } \ell + g$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = \ell + g + 1 \text{ through } m$$

3". $u_i \geq 0$, $i = 1$ through m $\qquad$ $v_i \geq 0$, $i = \ell + g + 1$ through m

$x_j \geq 0$, $y_j \geq 0$, $j = 1$ through n $\qquad$ $x_j y_j = 0$, $j = 1$ through n

$s_i \geq 0$, $i = 1$ through $\ell + g$ $\qquad$ $u_i s_i = 0$, $i = 1$ through $\ell + g$

These conditions are a set of simultaneous "almost linear" equations in non-negative variables. The solution to these equations can be found with Phase I of the simplex method with a modification to account for the nonlinear equations above. That modification is that x_j and y_j are not allowed to be simultaneously in the basis; nor are s_i and u_i.

This modification has been made to both SIMPLX and RVSMPX and is available with the use of the parameter KQP (see section 5.11). Note that in the discussion of this option in section 4, the s_i do not appear. This is so because RVSMPX and SIMPLX store slack and surplus variables implicitly. Therefore it is necessary only to put the original constraint coefficients in this part of the matrix.

APPENDIX B:

LISTING OF RVSMPX

```
JIT FOR REV,REV                                                                    17 AUG 71          19:51:16.445
UNIVAC 1108 FORTRAN V LEVEL    2206 0018 F5018P
THIS COMPILATION WAS DONE ON 17 AUG 71 AT 19:51:16

   SUBROUTINE RVSMPX     ENTRY POINT 004300

   STORAGE USED (BLOCK, NAME, LENGTH)

        0001    *CODE    004372
        0000    *DATA    000770
        0002    *BLANK   000000


   EXTERNAL REFERENCES (BLOCK, NAME)

        0003    NWDU$
        0004    NIO2$
        0005    NEXP1$
        0006    NIO1$
        0007    NERR2$
        0010    NERR3$


   STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001    000245 10L        0001    002560 1002G      0001    002635 1023G      0001    002725 1050G      0001    000701 1050L
    0001    002744 1062G      0001    002766 1073G      0001    000070 110G       0001    000720 1100L      0001    003003 1103G
    0001    000721 1105L      0001    000746 1110L      0001    001021 1120L      0001    003061 1124G      0001    001110 1130L
    0001    001166 1140L      0001    001223 1142L      0001    003126 1143G      0001    001244 1143L      0001    001313 1150L
    0001    001336 1153L      0001    001351 1160L      0001    003235 1174G      0001    001366 1200L      0001    001416 1203L
    0001    001521 1205L      0001    001546 1215L      0001    003300 1217G      0001    001575 1220L      0001    001625 1225L
    0001    003330 1230G      0001    001646 1230L      0001    001766 1240L      0001    003344 1242G      0001    002000 1250L
    0001    002004 1260L      0001    002063 1270L      0001    002122 1280L      0001    002132 1300L      0001    002213 1302L
    0001    002311 1305L      0001    002323 1310L      0001    003531 1311G      0001    003573 1332G      0001    003634 1350G
    0001    003733 1374G      0001    002346 1400L      0001    004005 1410G      0001    004015 1416G      0001    002456 1420L
    0001    004115 1444G      0001    004132 1453G      0001    004146 1464G      0001    000215 150G       0001    004212 1500G
    0001    004220 1506G      0001    002513 1520L      0001    000216 153G       0001    002565 1530L      0001    000267 16L
    0001    002606 1600L      0001    002652 1601L      0001    002677 1602L      0001    002753 1605L      0001    002775 1612L
    0001    003031 1616L      0001    003051 1621L      0001    003073 1630L      0001    003076 1635L      0001    003200 1645L
    0001    000256 167G       0001    003212 1700L      0001    003304 1730L      0001    003312 1735L      0001    003425 1740L
    0001    003457 1750L      0001    003465 1760L      0001    003474 1770L      0001    003505 1780L      0001    003517 1790L
    0001    003553 1800L      0001    003604 1810L      0001    003627 1900L      0001    003666 1905L      0001    003673 1910L
    0001    003677 1920L      0001    003702 1930L      0001    003711 1940L      0001    003745 1950L      0001    003751 1955L
    0001    004063 1960L      0001    004065 1970L      0001    004156 1980L      0001    004240 1990L      0001    004246 1995L
    0001    004250 1997L      0001    000306 204G       0001    000307 207G       0001    000404 211L       0001    000430 221L
    0001    000357 230G       0001    000447 231L       0001    000374 240G       0001    000526 241L       0001    000411 250G
    0001    000506 25L        0001    000435 261G       0001    000473 267G       0001    000475 272G       0001    000605 29L
    0001    000637 30L        0001    000523 306G       0001    000547 314G       0001    000642 32L        0001    001013 375G
    0001    000114 4L         0001    001216 434G       0001    001331 464G       0001    001455 517G       0001    001617 546G
    0001    001712 557G       0001    002040 630G       0001    002056 642G       0001    002270 711G       0001    002377 733G
    0001    002406 741G       0001    002451 750G       0000    000105 9000F      0000    000130 9001F      0000    000144 9002F
    0000    000175 9003F      0000    000204 9005F      0000    000224 9006F      0000    000231 9007F      0000    000263 9008F
    0000    000313 9010F      0000    000343 9011F      0000    000347 9012F      0000    000353 9014F      0000    000440 9015F
    0000    000507 9016F      0000    000510 9017F      0000    000511 9018F      0000    000557 9019F      0000    000631 9020F
    0000    000075 913F       0000 R  000066 APS        0000 R  000033 ASTAR      0000 R  000042 CEPS       0000 R  000036 EPS
```

```
0000 I 000022 I        0000 I 000000 IB       0000 I 000055 ICPSX    0000 I 000053 IMP      0000 I 000021 INVC
0000 I 000005 IP       0000 I 000047 IPHASE   0000 I 000070 IQ       0000 I 000071 IQ1      0000 I 000063 IR
0000 I 000035 ITC      0000 I 000051 ITER     0000 I 000034 ITP      0000 I 000067 IX       0000 I 000045 I1
0000 I 000065 I2       0000 I 000072 I3       0000 I 000073 I4       0000 I 000074 I5       0000 I 000041 J
0000 I 000056 JS       0000 I 000060 J1       0000 I 000061 J2       0000 I 000062 J3       0000 I 000057 J5
0000 I 000064 K        0000 I 000043 L1OFF    0000 I 000044 L2OFF    0000 I 000017 M        0000 I 000031 ME
0000 I 000026 MG       0000 I 000024 ML       0000 I 000050 MOBJ     0000 I 000027 MPN      0000 I 000030 MP1
0000 I 000025 MSMX     0000 I 000020 M2       0000 I 000023 N        0000 I 000046 NA       0000 I 000032 NMAX
0000 R 000054 OBJ      0000 R 000052 PIVM     0000 R 000012 PR       0000 R 000037 XX       0000 R 000040 Y


00101         SUBROUTINE RVSMPX (A,MA,B,MB,MT,NT,L,X,TOLP,INV,KQP)          R10000
00103         DIMENSION A(MA,1),B(MB,1),L(1),X(1),IB(5),IP(5),PR(5)         R10100
00104         M=MT-1                                                         R10200
00105         M2=M+2                                                         R10300
00106         INVC=M/2+5                                                     R10400
00106   C                                                                    R10500
00106   C * * CHECK FOR NEGATIVE RIGHT HAND SIDES.                          R10600
00106   C                                                                    R10700
00107         DO 4 I=1,M                                                     R10800
00112         IF (A(I,NT).GE.0) GO TO 4                                      R10900
00114         WRITE(6,9020)                                                  R11000
00116         L(3)=2                                                         R11100
00117         RETURN                                                         R11200
00120       4 CONTINUE                                                       R11300
00122         N=NT-1                                                         R11400
00123         ML=L(1)                                                        R11500
00124         MSMX=14                                                        R11600
00125         MG=L(2)                                                        R11700
00126         MPN=M+N                                                        R11800
00127         MP1=M+1                                                        R11900
00130         ME=M-ML-MG                                                     R12000
00131         NMAX=N+ML+ME+2*MG                                              R12100
00132         ASTAR='*'                                                      R12200
00133         ITP=N-KQP                                                      R12300
00134         ITC=KQP+ML+MG                                                  R12400
00135         L(2)=0                                                         R12500
00136         IF (INV.GT.0) INV=1                                            R12600
00136   C                                                                    R12700
00136   C * * COMPUTE EPSILON AND CAPITAL EPSILON.                           R12800
00136   C                                                                    R12900
00140         EPS=TOLP                                                       R13000
00141         IF (EPS.GT.0.0)  GO TO 10                                      R13100
00143         XX=1.0E-5                                                      R13200
00144         IF (EPS.LT.0.0)  XX=-EPS                                       R13300
00146         Y=0.0                                                          R13400
00147         DO 5  I=1,M                                                    R13500
00152         DO 5  J=1,N                                                    R13600
00155         EPS=A(I,J)                                                     R13700
00156         IF (EPS.LT.0.0)  EPS=-EPS                                      R13800
00160       5 Y=Y+EPS                                                        R13900
00163         EPS=XX*Y/(M*N)                                                 R14000
00164      10 IF (L(3).EQ.0) GO TO 16                                        R14100
00166      14 DO 15  I=3,MSMX                                                R14200
00171      15 L(I)=0                                                         R14300
00173         L(4)=1                                                         R14400
```

```
      L(5)=1
      L(8)=1
      L(12)=1
      L(13)=1
      L(MSMX)=1
   15 CEPS=10.0*EPS
      L(1)=0
      DO 17 I=1,M
      DO 17 J=1,N
      XX=ABS(A(I,J))
   17 IF (XX.LT.EPS.AND.XX.GT.0.0) L(1)=L(1)+1
      IF (L(4).NE.0)  WRITE (6,9000) EPS,CEPS,L(1)
C
C * * SET UP L1(.)-VARIABLE TYPE INDICATOR.
C
      L1OFF=MSMX
      L2OFF=L1OFF+NMAX
      I=1
      DO 20  J=1,N
      L(L1OFF+I)=0
   20 I=I+1
      IF (ML.EQ.0)  GO TO 211
      DO 21  J=1,ML
      L(L1OFF+I)=1
   21 I=I+1
  211 IF (MG.EQ.0)  GO TO 221
      DO 22 J=1,MG
      L(L1OFF+I)=2
      L(L1OFF+I+MG)=3
   22 I=I+1
  221 IF (ME.EQ.0)  GO TO 231
      DO 23  J=1,ME
      L(L1OFF+I+MG)=3
   23 I=I+1
C
C * * INITIALIZE B(.,.), THE BASIS INVERSE, TO THE IDENTITY MATRIX.
C
  231 DO 26  I=1,M2
      DO 25  J=1,M2
   25 B(I,J)=0.0
      IF (I.GT.M)  GO TO 26
      B(I,I)=1.0
   26 CONTINUE
      IF (INV.GT.0) GO TO 241
C
C * * INITIALIZE THE X'S, B(M2,.)-THE MULTIPLIERS, B(.,M2)-THE VALUES
C * * OF THE BASIC VARIABLES, AND B(M2,M2)-THE ARTIFICIAL OBJ. VALUE.
C
      DO 28 I=1,NMAX
   28 X(I)=-EPS/100.0
  241 J=1
      DO 32 I=1,NMAX
      I1=L(L1OFF+I)
      IF (I1.EQ.0.OR.I1.EQ.2)  GO TO 32
      IF (INV.GT.0) GO TO 29
      X(I)=A(J,NT)
      L(L2OFF+J)=I
   29 B(J,M2)=A(J,NT)
```

```
00326          IF (I1.LT.3)  GO TO 30
00330          B(M2,J)=1.0
00331          B(M2,M2)=B(M2,M2)+B(J,M2)
00332       30 J=J+1
00333       32 CONTINUE
00335          IF (INV.EQ.2) GO TO 1900
00335    C
00335    C * * INITIALIZATIONS FOR PHASE I.
00335    C
00337          NA=0
00340          IPHASE=1
00341          MOBJ=M2
00342          ITER=0
00343          PIVM=1.0E28
00344          IMP=0
00345          OBJ=-B(MOBJ,M2)
00345          IF (INV.EQ.1) GO TO 1900
00346    C
00346    C * * CHECK FOR IMMEDIATE BASIC FEASIBLE SOLUTION.
00345    C
00350          IF (ML.LT.M) GO TO 1100
00350    C
00350    C * * INITIALIZATIONS FOR PHASE II.
00350    C
00352     1050 MOBJ=MP1
00353          IPHASE=2
00354          ITER=0
00355          PIVM=1.0E28
00356          IMP=0
00357          OBJ=-B(MOBJ,M2)
00357    C
00357    C *******************************************
00357    C *** STEPS 1 AND 2-COMPUTE THE C' TO GET S.***
00357    C *******************************************
00357    C
00360     1100 ICPSX=0
00361     1105 NA=0
00362          IF (INV.GT.0) GO TO 1110
00364          JS=0
00365          IF (L(2).EQ.0) GO TO 1110
00367          IF (L(2)/INVC*INVC.EQ.L(2)) INV=2
00371          IF (INV.EQ.2) GO TO 231
00371    C
00371    C * * CALCULATE THE INDICATORS-C'.
00371    C
00373     1110 B(MOBJ,MP1)=0.0
00374          DO 1160 J5=1,NMAX
00377          IF (INV.EQ.0) GO TO 1120
00401          J=JS
00402          GO TO 1130
00403     1120 J=J5
00404          IF (X(J).GE.0.0) GO TO 1160
00406          IF (KOP.EQ.0) GO TO 1130
00410          IF (J.LE.ITC.AND.X(ITP+J).GE.0.0) GO TO 1160
00412          IF (J.GT.ITP.AND.J.LE.ITP+ITC.AND.X(J-ITP).GE.0.0) GO TO 1160
00414     1130 J1=L(L1OFF+J)
00415          IF (J1.EQ.4)  GO TO 1160
00417          IF (J1.EQ.0)  GO TO 1140
```

```
00421         J2=J-N
00422         IF (J.GT.N+ML+MG) J2=J2-MG
00422   C
00422   C * * COMPUTE C' FOR SLACK, SURPLUS, OR ARTIFICIAL VARIABLES.
00422   C
00424         XX=(-1)**(J1+1)*B(MOBJ,J2)
00425         IF (J1.EQ.3) XX=XX-1.0
00427         GO TO 1142
00427   C
00427   C * * COMPUTE C' FOR REAL VARIABLES.
00427   C
00430    1140 XX=0.0
00431         IF (IPHASE.EQ.2) XX=A(MP1,J)
00433         DO 1141 I=1,M
00436    1141 XX=XX+B(MOBJ,I)*A(I,J)
00436   C
00436   C * * TRANSFER IF INVERTING OR PIVOTING OUT ARTIFICIALS.
00436   C
00440    1142 IF (INV.GT.0.OR.ICPSX.EQ.1) GO TO 1143
00440   C
00440   C * * TEST FOR MINIMUM C' IF DOING A NORMAL PIVOT.
00440   C
00442         IF (XX.LE.B(MOBJ,MP1)) GO TO 1160
00444    1143 B(MOBJ,MP1)=XX
00445         JS=J
00446         J3=J2
00447         IF (INV.GT.0) GO TO 1200
00451         IF (ICPSX.EQ.0) GO TO 1160
00451   C
00451   C * * IF TRYING TO PIVOT OUT ARTIFICIALS, CHECK THIS COLUMN.  IF OK TO
00451   C * * ENTER, COMPUTE PIVOT ELEMENT.  IF PIVOT IS BIG ENOUGH, PIVOT.
00451   C * * OTHERWISE, CONTINUE ON TO NEXT COLUMN.
00451   C
00453    1145 IF (XX.LT.-EPS) GO TO 1160
00455         JS=J
00456         IF (J1.EQ.0)  GO TO 1150
00460         B(IR,MP1)=(-1)**(J1+1)*B(IR,J2)
00461         GO TO 1153
00462    1150 B(IR,MP1)=0.0
00463         DO 1151 K=1,M
00466    1151 B(IR,MP1)=B(IR,MP1)+B(IR,K)*A(K,JS)
00470    1153 IF (B(IR,MP1).GT.EPS) GO TO 1200
00472         IF (B(IR,MP1).GT.-EPS) GO TO 1160
00474         GO TO 1200
00475    1160 CONTINUE
00477         IF (ICPSX.EQ.1) GO TO 1635
00501         IF (B(MOBJ,MP1).LT.EPS) GO TO 1600
00501   C
00501   C ********************************
00501   C *** STEP 3-CALCULATE A'(I,S). ***
00501   C ********************************
00501   C
00503    1200 I1=L(L1OFF+JS)
00504         IB(1)=' X(    '
00505         IP(1)=JS
00506         IF (I1.EQ.0) GO TO 1203
00510         IB(1)=' S(    '
00511         IP(1)=JS-N
```

```
00512           IF (I1.LT.3) GO TO 1203
00514           IJ(1)=' A(   '
00515           IP(1)=IP(1)-ML-MS
00516      1203 DO 1225 I=1,MP1
00516     C
00516     C * * COMPUTE REAL OBJECTIVE COST IF NOT ALREADY COMPUTED.
00516     C
00521           IF (I.LT.MP1) GO TO 1205
00523           IF (IPHASE.EQ.2) GO TO 1225
00525           B(I,MP1)=0.0
00526           IF (I1.EQ.0) B(I,MP1)=A(MP1,JS)
00530           IF (I1.EQ.3) B(I,MP1)=-1.0
00532           GO TO 1215
00532     C
00532     C * * GENERATE THE REST OF THE COLUMN.
00532     C
00533      1205 B(I,MP1)=0.0
00534      1207 I2=L(L2OFF+I)
00535           IF (I2.EQ.-1) GO TO 1215
00537           IF (L(L1OFF+I2).EQ.5) GO TO 1225
00541      1215 IF (I1.EQ.0)  GO TO 1220
00543           B(I,MP1)=B(I,MP1)+(-1)**(I1+1)*B(I,J3)
00544           GO TO 1225
00545      1220 DO 1223  K=1,M
00550      1223 B(I,MP1)=B(I,MP1)+B(I,K)*A(K,JS)
00552      1225 CONTINUE
00552     C
00552     C * * IF PIVOTING OUT AN ARTIFICIAL, SET THE VALUE OF THIS ARTIFICIAL
00552     C * * TO ZERO.  IF IT IS GT EPS SET L(3)=1 (COMPUTATIONAL INCONSISTENCY)
00552     C
00554           IF (ICPSX.EQ.0) GO TO 1230
00556           IF (B(IR,M2).GT.EPS) L(3)=1
00560           B(IR,M2)=0.0
00561           GO TO 1300
00561     C
00561     C *************************************
00561     C *** STEP 4-FORM QUOTIENTS TO GET R. ***
00561     C *************************************
00561     C
00562      1230 IR=0
00563           APS=1.0E28
00564           IF (INV.GT.0) APS=0.0
00566           DO 1260 I=1,M
00571           IF (INV.GT.0) GO TO 1240
00573           IF (B(I,MP1).LE.0.0) GO TO 1260
00575           XX=B(I,M2)/B(I,MP1)
00576           IF (XX.GT.APS)  GO TO 1260
00600           IF (XX.LT.APS.OR.IR.EQ.0) GO TO 1250
00602           IF (B(IR,MP1).GE.B(I,MP1)) GO TO 1260
00604           GO TO 1250
00604     C
00604     C * * IF INVERTING, FIND THE LARGEST ABSOLUTE ELEMENT AMONG
00604     C * * THOSE ROWS NOT ALREADY PIVOTED IN.
00604     C
00605      1240 IF (L(L2OFF+I).NE.-1) GO TO 1260
00607           XX=ABS(B(I,MP1))
00610           IF (XX.LE.APS) GO TO 1260
00612      1250 IR=I
```

```
      APS=XX
 1260 CONTINUE
      IF (IR.GT.0) GO TO 1270
C
C * * IF INVERTING, AND THERE IS NO PLACE TO PUT THIS
C * * VARIABLE, SKIP IT AND TRY AGAIN LATER.
C
      IF (INV.GT.0) GO TO 1940
C
C * * THE PROBLEM IS UNBOUNDED.
C
      L(3)=3
      IX=1
      IF (L(MSMX).EQ.0) RETURN
      WRITE(6,9015) (ASTAR,I=1,120),JS
      WRITE(6,9018) ITER,IPHASE,OBJ,(ASTAR,I=1,120)
      GO TO 1735
 1270 XX=ABS(B(IR,MP1))
      IF (XX.GT.CEPS) GO TO 1300
C
C * * IF PIVOT IS TOO SMALL, SET TO ZERO AND TRY AGAIN.
C
      IF (XX.LT.EPS) GO TO 1280
C
C * * IF EPS.LT.PIVOT.LT.CEPS, PRINT WARNING AND CONTINUE.
C
      IF (L(5).NE.0) WRITE(6,9005) ITER,IPHASE,B(IR,MP1)
      GO TO 1300
 1280 B(IR,MP1)=0.0
      GO TO 1230
C
C ******************************************
C *** STEP 5-CALCULATE NEW B(.,MT) AND X. ***
C ******************************************
C
 1300 IF (INV.GT.0) GO TO 1302
      I1=L(L2OFF+IR)
      I2=L(L1OFF+I1)
      X(I1)=-EPS/100.0
      IF (L(L1OFF+I1).EQ.3)  L(L1OFF+I1)=4
      IB(2)=' X(    '
      IP(2)=I1
      IF (I2.EQ.0) GO TO 1302
      IB(2)=' S(    '
      IP(2)=I1-N
      IF (I2.LT.3) GO TO 1302
      IB(2)=' A(    '
      IP(2)=IP(2)-ML-MG
 1302 L(L2OFF+IR)=JS
      B(IR,M2)=B(IR,M2)/B(IR,MP1)
      DO 1310  I=1,M2
      IF (I.EQ.IR)  GO TO 1305
      B(I,M2)=B(I,M2)-B(IR,M2)*B(I,MP1)
      IF (I.GT.M)  GO TO 1310
 1305 IF (INV.GT.0) GO TO 1310
      I1=L(L2OFF+I)
      X(I1)=B(I,M2)
 1310 CONTINUE
```

```
00726         IF (INV.EQ.2) GO TO 1400
00730         X(NMAX+1)=-B(MP1,M2)
00731         OBJ=-B(MOBJ,M2)
00731   C
00731   C ****************************************************************
00731   C *** STEPS 6 AND 7-UPDATE THE INVERSE AND THE MULTIPLIERS. ***
00731   C ****************************************************************
00731   C
00732    1400 DO 1402 K=1,M
00735         B(IR,K)=B(IR,K)/B(IR,MP1)
00736    1402 CONTINUE
00740         DO 1420 I=1,M2
00743         IF (I.EQ.IR)  GO TO 1420
00745         IF (I.EQ.M2.AND.IPHASE.EQ.2) GO TO 1420
00747         DO 1410 K=1,M
00752         B(I,K)=B(I,K)-B(I,MP1)*B(IR,K)
00753    1410 CONTINUE
00755    1420 CONTINUE
00757         IF (INV.GT.0) GO TO 1930
00761         ITER=ITER+1
00762         L(2)=L(2)+1
00762   C
00762   C * * STORE THE MINIMUM PIVOT.
00762   C
00763         IF (B(IR,MP1).GT.PIVM) GO TO 1520
00765         PIVM=B(IR,MP1)
00765         IMP=ITER
00767    1520 I1=5+(IPHASE-1)*4
00767   C
00767   C * * CHECK TO PRINT ITERATION SUMMARY.
00767   C
00770         IF (L(I1+1).EQ.0) GO TO 1530
00772         IF (ITER/L(I1+1)*L(I1+1).NE.ITER) GO TO 1530
00774         WRITE(6,9010) IPHASE,ITER,B(IR,MP1),OBJ,(IB(I),IP(I),I=1,2)
00774   C
00774   C * * CHECK TO PRINT ITERATION BASIS.
00774   C
01007    1530 IF(L(I1+2).EQ.0) GO TO 1100
01011         IF (ITER/L(I1+2)*L(I1+2).NE.ITER) GO TO 1100
01013         IX=2
01014         GO TO 1735
01014   C
01014   C ********************
01014   C *** END OF PHASE. ***
01014   C ********************
01014   C
01015    1600 IX=3
01016         IF ( IPHASE.EQ.2)  GO TO 1700
01020         IQ=0
01020   C
01020   C * * COMPUTE THE NUMBER OF ARTIFICIALS STILL IN THE BASIS.
01020   C
01021         NA=0
01022         DO 1601 J=1,NMAX
01025         IF (L(L1OFF+J).NE.3) GO TO 1601
01027         IF (X(J).GT.CEPS) GO TO 1602
01031         NA=NA+1
01032    1601 CONTINUE
```

```
01034        IF (B(MOBJ,M2).GT.CEPS)  GO TO 1602
01034   C
01034   C * * IF THE ARTIFICIAL OBJECTIVE FUNCTION IS ACCEPTABLE BUT ARTIFICIALS
01034   C * * ARE STILL IN THE BASIS, GO TRY TO PIVOT THEM OUT.
01034   C
01036        IF (NA.GT.0)  GO TO 1612
01036   C
01036   C * * IF NO ARTIFICIALS ARE IN THE BASIS AND THE ARTIFICIAL OBJECTIVE
01036   C * * FUNCTION IS NOT QUITE ZERO, PRINT WARNING OF ROUND-OFF ERROR.
01036   C
01040        IF (B(MOBJ,M2).GT.EPS) GO TO 1605
01042        GO TO 1700
01042   C
01042   C * * THE PROBLEM IS INFEASIBLE.
01042   C
01043   1602 L(3)=2
01044        IF(L(MSMX).EQ.0) RETURN
01046        WRITE(6,9014) (ASTAR,I=1,120),B(MOBJ,MP1)
01055        WRITE(6,9018) ITER,IPHASE,OBJ,(ASTAR,I=1,120)
01056        IX=1
01057        GO TO 1735
01070   1605 IF (L(MSMX).NE.0) WRITE(6,9019) (ASTAR,I=1,240)
01077        L(3)=1
01100        GO TO 1700
01100   C
01100   C * * FIND THE ARTIFICIAL STILL IN THE BASIS.
01100   C
01101   1612 IQ1=0
01102        DO 1615 J=1,NMAX
01105        J1=L(L1OFF+J)
01106        IF (J1.EQ.3)  IQ1=IQ1+1
01110        IF (IQ1.GT.IQ) GO TO 1621
01112   1615 CONTINUE
01114   1616 L(3)=4
01115        IF (L(MSMX).NE.0)  WRITE (6,913)
01120    913 FORMAT ('0SYSTEM ERROR- COMPUTATIONAL IMPOSSIBILITY')
01121        IX=1
01122        GO TO 1735
01122   C
01122   C * * DETERMINE WHICH BASIC VARIABLE IT IS.
01122   C
01123   1621 DO 1625  IR=1,M
01125        I=L(L2OFF+IR)
01127        IF (I.EQ.J)  GO TO 1630
01131   1625 CONTINUE
01133        GO TO 1616
01133   C
01133   C * * SET ARTIFICIALS IN AT ZERO SWITCH, AND GO TRY TO PIVOT IT OUT.
01133   C
01134   1630 ICPSX=1
01135        GO TO 1105
01135   C
01135   C * * INCREMENT IQ, THE COUNTER FOR CURRENT NUMBER OF ARTIFICIALS THAT
01135   C * * CAN'T BE PIVOTED OUT.
01135   C
01136   1635 IQ=IQ+1
01137        IF (NA.GT.IQ) GO TO 1612
01137   C
```

```
01137   C * * COUNT AND CHECK THE ARTIFICIALS STILL IN THE BASIS.
01137   C
01141         JS=0
01142         DO 1645  I=1,M
01145         I1=L(L2OFF+I)
01146         IF (L(L1OFF+I1).NE.3) GO TO 1645
01150         JS=JS+1
01150   C
01150   C * * IF IN AND GT EPSILON, INFEASIBLE. OTHERWISE SET TO 0 AND CONTINUE.
01150   C
01151         IF (B(I,M2).GT.CEPS) GO TO 1602
01153         IF (B(I,M2).GT.EPS) L(3)=1
01155         X(I1)=0.0
01156         L(L1OFF+I1)=5
01157         B(I,M2)=0.0
01160    1640 CONTINUE
01161    1645 CONTINUE
01163         IF (JS.EQ.0)  GO TO 1616
01165         IF (L(3).EQ.1) GO TO 1605
01165   C
01165   C *********************
01165   C *** SUMMARY PRINT.  ***
01165   C *********************
01165   C
01167    1700 I1=5+(IPHASE-1)*4
01170         IF(L(I1+3).EQ.0) GO TO 1730
01172         WRITE(6,9008) (ASTAR,I=1,120),IPHASE,OBJ,ITER
01203         IF(ITER.EQ.0) PIVM=0.0
01205         WRITE(6,9001) PIVM,IMP
01211    1710 IF(IPHASE.EQ.1) WRITE(6,9003) X(NMAX+1)
01215         WRITE(6,9006) (ASTAR,I=1,120)
01215   C
01215   C *******************
01215   C *** BASIS PRINT.  ***
01215   C *******************
01215   C
01223    1730 IF (L(I1+4).EQ.0) GO TO  1810
01225    1735 WRITE(6,9012)
01227         DO 1737 I2=1,5
01232         IB(I2)='      '
01233         PR(I2)=0.0
01234    1737 IP(I2)=0
01235         I3=0
01237         I4=0
01240         IO1=0
01241         DO 1800 I2=1,NMAX
01244         IF (X(I2).LT.0.0.AND.X(I2).GT.-EPS) GO TO 1800
01246         IF (I4.EQ.L(L1OFF+I2)) GO TO 1740
01250         IF (I4.EQ.1.AND.L(L1OFF+I2).EQ.2) GO TO 1740
01252         IF (I4.EQ.3.AND.L(L1OFF+I2).EQ.5) GO TO 1740
01254         IF (L(L1OFF+I2).EQ.4) GO TO 1780
01255         IO1=1
01257         GO TO 1790
01260    1740 I4=L(L1OFF+I2)
01261         IF (I3.EQ.5) GO TO 1790
01263         I3=I3+1
01264         PR(I3)=X(I2)
01265         I5=I4+1
```

```
      GO TO (1750,1760,1760,1770,1780,1770),I5
 1750 IB(I3)=' X(    '
      IP(I3)=I2
      GO TO 1800
 1760 IB(I3)=' S(    '
      IP(I3)=I2-N
      GO TO 1800
 1770 IB(I3)=' A(    '
      IP(I3)=I2-N-ML-MG
      GO TO 1800
 1780 L(3)=4
      IX=1
      WRITE(6,913)
      GO TO 1800
 1790 IF (I3.EQ.0) GO TO 1740
      WRITE(6,9011) (IB(J),IP(J),PR(J),J=1,I3)
      IF (IQ1.EQ.1) WRITE(6,9017)
      IQ1=0
      I3=0
      GO TO 1740
 1800 CONTINUE
      IF (I3.GT.0) WRITE(6,9011) (IB(J),IP(J),PR(J),J=1,I3)
 1810 IF (IX.EQ.1) RETURN
      IF (IX.EQ.2)  GO TO 1100
      IF (IPHASE.EQ.2)  RETURN
      GO TO 1050
C
C ******************************
C *** TOTAL INVERSION ROUTINE.  ***
C ******************************
C
 1900 DO 1910 I=1,M
      I1=L(L2OFF+I)
      IF (L(L1OFF+I1).NE.5) GO TO 1905
      B(I,M2)=0.0
      GO TO 1910
 1905 L(L2OFF+I)=-1
 1910 CONTINUE
 1920 I4=0
      JS=0
      GO TO 1940
C
C * * I4=1 INDICATES THAT A PIVOT WAS PERFORMED.
C
 1930 I4=1
      X(JS)=-EPS/100.0
 1940 I5=JS
      IF (I5.EQ.NMAX) GO TO 1955
C
C * * FOR EACH X(I) THAT IS NON-NEGATIVE AND NOT AN ARTIFICIAL IN AT
C * * A ZERO LEVEL, GO TRY TO PUT IT IN THE BASIS.  DON'T DO ANYTHING
C * * FOR AN ARTIFICIAL IN AT A ZERO LEVEL.
C
      I5=I5+1
      DO 1950 JS=I5,NMAX
      IF (X(JS).LT.0.0) GO TO 1950
      IF (L(L1OFF+JS).EQ.5) GO TO 1930
      GO TO 1100
```

```
01403   1950 CONTINUE
01403   C
01403   C * * IF AT LEAST ONE PIVOT OPERATION WAS DONE, GO THROUGH X AGAIN.
01403   C
01405   1955 IF (I4.EQ.1) GO TO 1920
01407        DO 1957 J=1,NMAX
01412   1957 X(J)=-EPS/100.0
01412   C
01412   C * * SET X TO PROPER VALUES AND CHECK FOR FEASIBILITY.
01412   C
01414        I5=0
01415        DO 1970 I=1,M
01420        IF (B(I,M2).LT.-EPS) I5=1
01422        IF (B(I,M2).LT.0.0.AND.B(I,M2).GE.-EPS) B(I,M2)=0.0
01424        I1=L(L2OFF+I)
01425        IF (I1.LT.0) GO TO 1960
01427        X(I1)=B(I,M2)
01430        GO TO 1970
01431   1960 I5=1
01432   1970 CONTINUE
01434        IF (I5.EQ.0) GO TO 1980
01434   C
01434   C * * INFEASIBLE AFTER INVERSION.
01434   C
01435        L(3)=2
01437        IF (L(MSAX).EQ.0) RETURN
01441        IF (INV.EQ.1) WRITE(6,9007) (ASTAR,I=1,120)
01450        IF (INV.EQ.2) WRITE(6,9002) (ASTAR,I=1,120)
01457        WRITE(6,9018) ITER,IPHASE,OBJ,(ASTAR,I=1,120)
01470        IX=1
01471        GO TO 1735
01471   C
01471   C * * IF INVERTING BECAUSE OF INITIAL BASIS, FIX UP L1(.).
01471   C
01472   1980 IF (INV.EQ.2) GO TO 1997
01474        I5=M+ML+MG+1
01475        IF (I5.GT.NMAX) GO TO 1995
01477        DO 1985 J=I5,NMAX
01502   1985 L(L1OFF+J)=4
01504        I5=0
01505        DO 1990 J=1,M
01510        J1=L(L2OFF+J)
01511        IF (L(L1OFF+J1).NE.4) GO TO 1990
01513        L(L1OFF+J1)=3
01514        I5=1
01515   1990 CONTINUE
01517        IF (I5.EQ.1) GO TO 1997
01521   1995 INV=0
01522        GO TO 1050
01523   1997 INV=0
01524        GO TO 1110
01524   C
01524   C
01525   9000 FORMAT(' EPSILON = 'G13.6,'  CAPITAL EPSILON = 'G13.6,'  'I5,' NON
01525       *-ZERO ENTRIES ARE EFFECTIVELY EQUAL TO ZERO.')
01526   9001 FORMAT(' *'14X'MINIMUM PIVOT WAS'G12.5,' AT ITERATION'I4,'.'57X'*'
01526       *)
01527   9002 FORMAT(/////' '120A1/' *'118X'*'/' *  ERROR - THE PROBLEM IS INFEA
```

```
01527   0626*        *SIBLE.  INFEASIBILITY INDICATED DURING RE-INVERSION OF THE BASIS M  R72500
01527   0627*        *ATRIX.'16X'*')                                                      R72600
01530   0628*    9003 FORMAT(1H+65X'REAL OBJECTIVE FUNCTION ='G15.8)                      R72700
01531   0629*    9005 FORMAT('0* * WARNING * *  SMALL PIVOT ELEMENT AT ITERATION'I4,' OF  R72800
01531   0630*        * PHASE'I2,'. PIVOT ='G14.7)                                         R72900
01532   0631*    9006 FORMAT(' *'118X'*'/' '120A1)                                        R73000
01533   0632*    9007 FORMAT(/////' '120A1/' *'118X'*'/' *  ERROR - THE VARIABLES SPECIF  R73100
01533   0633*        *IED AS COMPRISING AN INITIAL SOLUTION DO NOT FORM A BASIC FEASIBLE  R73200
01533   0634*        * SOLUTION.'12X'*')                                                  R73300
01534   0635*    9008 FORMAT(/////' '120A1/' *'118X'*'/' *'14X'END OF PHASE'I2,'.     OBJ  R73400
01534   0636*        *ECTIVE FUNCTION ='G18.8,4X'THERE WERE'I4,' ITERATIONS.'17X'*')      R73500
01535   0637*    9010 FORMAT (///'0PHASE'I2,' ITERATION',I4,'.  PIVOT=',G13.6,'   OBJEC  R73600
01535   0638*        *TIVE FUNCTION='G15.9,'  ',A3,I3,') ENTERED THE BASIS,',A3,I3,') LE  R73700
01535   0639*        *FT.')                                                               R73800
01536   0640*    9011 FORMAT(5(5XA3,I3,')='F12.6))                                        R73900
01537   0641*    9012 FORMAT('0BASIC VARIABLES'/)                                         R74000
01540   0642*    9014 FORMAT(/////' '120A1/' *'118X'*'/' *  ERROR - THE PROBLEM IS INFEA  R74100
01540   0643*        *SIBLE. THE CONSTRAINTS ASSOCIATED WITH THE ARTIFICIAL VARIABLES BE  R74200
01540   0644*        *LOW ARE INCONSISTENT. *'/' *  IF NONE APPEAR, NUMERICAL DIFFICULTI  R74300
01540   0645*        *ES HAVE BEEN ENCOUNTERED. THE LARGEST ENTRY IN THE OBJECTIVE FUNCT  R74400
01540   0646*        *ION ROW','7X,'*'/' *  IS ',G12.5,101X,'*')                          R74500
01541   0647*    9015 FORMAT(/////' '120A1/' *'118X'*'/' *  ERROR - THE PROBLEM IS UNBOU  R74600
01541   0648*        *NDED. THE VARIABLE X('I3,') CAN ASSUME AN ARBITRARILY LARGE VALUE,  R74700
01541   0649*        * THEREBY YIELDING'7X'*'/' *  AN ARBITRARILY LARGE VALUE OF THE OBJ  R74800
01541   0650*        *ECTIVE FUNCTION.'63X'*')                                            R74900
01542   0651*    9016 FORMAT (1H1)                                                        R75000
01543   0652*    9017 FORMAT (1H )                                                        R75100
01544   0653*    9018 FORMAT(' *'118X'*'/' *  THE ERROR WAS DETECTED AT ITERATION'I4,' O  R75200
01544   0654*        *F PHASE'I2,'. AT THAT TIME THE OBJECTIVE FUNCTION VALUE WAS'G15.8,  R75300
01544   0655*        *4X'*'/' *  AND THE FOLLOWING VARIABLES WERE BASIC.'77X'*'/' *'118X  R75400
01544   0656*        *'*'/' '120A1)                                                       R75500
01545   0657*    9019 FORMAT(///' '120A1/' *'118X'*'/' *  WARNING - COMPUTATIONAL INCONS  R75600
01545   0658*        *ISTENCY INDICATED AT THE END OF PHASE 1. THE ALGORITHM WILL CONTIN  R75700
01545   0659*        *UE WITH PHASE 2,'4X'*'/' *  BUT THE USER IS ADVISED TO CRITICIZE T  R75800
01545   0660*        *HE RESULTS.'67X'*'/' *'118X'*'/' '120A1)                            R75900
01546   0661*    9020 FORMAT('0* *  AT LEAST ONE ELEMENT OF THE RIGHT HAND SIDE COLUMN I  R76000
01546   0662*        *S LESS THAN ZERO. SUBROUTINE TERMINATES.')                          R76100
01547   0663*         END                                                                 R76200

      END OF UNIVAC 1108 FORTRAN V COMPILATION.     0 *DIAGNOSTIC* MESSAGE(S)
PHASE 1 TIME =   1 SEC.
PHASE 2 TIME =   0 SEC.
PHASE 3 TIME =   3 SEC.
PHASE 4 TIME =   0 SEC.
PHASE 5 TIME =   2 SEC.
PHASE 6 TIME =   1 SEC.

TOTAL COMPILATION TIME =   7  SEC
```

APPENDIX C:

LISTING OF SIMPLX

```
BIT FOR SIMPLX,SIMPLX                                                   19 AUG 71          08:46:50.064
UNIVAC 1108 FORTRAN V LEVEL   2206 0018 F50182
THIS COMPILATION WAS DONE ON 19 AUG 71 AT 08:46:50

   SUBROUTINE SIMPLX     ENTRY POINT 003540

   STORAGE USED (BLOCK, NAME, LENGTH)

        0001    *CODE    003704
        0000    *DATA    000763
        0002    *BLANK   000000


   EXTERNAL REFERENCES (BLOCK, NAME)

        0003    NWDU$
        0004    NIO2$
        0005    NIO1$
        0006    NERR2$
        0007    NERR3$


   STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001     000262 10L        0001     002373 1023G       0001     002423 1031G       0001     002517 1051G       0001     000120 11L
    0001     002653 1113G      0001     000074 112G        0001     003004 1147G       0001     000324 12L         0001     003173 1226G
    0001     003201 1233G      0001     003262 1257G       0001     003276 1270G       0001     003343 1305G       0001     000134 132G
    0001     003400 1324G      0001     003413 1334G       0001     003462 1351G       0001     003515 1370G       0001     003530 1400G
    0001     003577 1415G      0001     000214 161G        0001     000217 164G        0001     001106 2000L       0001     001126 2001L
    0001     001140 2002L      0001     001141 2005L       0001     001177 2006L       0001     001155 2007L       0001     001220 2008L
    0001     001246 2009L      0001     001276 2010L       0001     001321 2012L       0001     001324 2015L       0001     000302 202G
    0001     001347 2025L      0001     000303 205G        0001     000405 232G        0001     000421 240G        0001     000446 254G
    0001     000473 263G       0001     000475 267G        0001     000532 302G        0001     000451 32L         0001     000634 325G
    0001     000677 340G       0001     000751 351G        0001     001025 365G        0001     000145 4L          0001     001053 402G
    0001     001071 414G       0001     001225 467G        0001     000237 5L          0001     000515 50L         0001     001364 5000L
    0001     001454 5005L      0001     001476 5010L       0001     001511 5020L       0001     001521 5025L       0001     001524 5030L
    0001     001527 5100L      0001     001572 5110L       0001     001272 513G        0001     001667 5150L       0001     001713 5155L
    0001     001722 5170L      0001     001724 5175L       0001     001725 5300L       0001     002023 5305L       0001     002045 5310L
    0001     002060 5315L      0001     002140 5320L       0001     002206 5330L       0001     002216 5335L       0001     002232 5344L
    0001     002300 5345L      0001     002322 5350L       0001     002326 5400L       0001     002354 5401L       0001     002453 5410L
    0001     002503 5415L      0001     002603 5425L       0001     002624 5428L       0001     002700 5440L       0001     002730 5445L
    0001     002752 5448L      0001     003057 5460L       0001     000573 55L         0001     003016 5500L       0001     003071 5530L
    0001     003073 5535L      0001     001415 554G        0001     003134 5540L       0001     003135 5550L       0001     003147 5600L
    0001     003233 5608L      0001     003241 5609L       0001     003245 5610L       0001     003313 5620L       0001     003352 5625L
    0001     003432 5630L      0001     003471 5635L       0001     003474 5637L       0001     003547 5640L       0001     003606 5645L
    0001     001501 574G       0001     000242 6L          0001     000643 60L         0001     001552 615G        0001     001634 634G
    0001     000737 66L        0001     001761 663G        0001     002050 705G        0001     002066 714G        0001     001000 75L
    0001     002255 763G       0001     002273 775G        0001     001030 85L         0001     001041 90L         0000     000125 9000F
    0000     000150 9001F      0000     000164 9003F       0000     000173 9005F       0000     000213 9006F       0000     000220 9007F
    0000     000250 9008F      0000     000310 9010F       0000     000340 9011F       0000     000344 9012F       0000     000350 9014F
    0000     000435 9015F      0000     000504 9016F       0000     000505 9017F       0000     000506 9018F       0000     000554 9019F
    0000     000626 9020F      0001     001075 91L         0000 R   000040 AA          0000 R   000021 ASTAR       0000 R   000037 CEPS
    0000 R   000033 EPS        0000 I   000026 I           0000 I   000071 IABC        0000 I   000072 IABC1       0000 I   000000 IB
    0000 I   000077 IFOUND     0000 I   000067 IC1         0000 I   000070 IC2         0000 I   000115 IENT        0000 I   000050 IG
    0000 I   000103 ILOOK      0000 I   000116 ILVE        0000 I   000074 IMP         0000 I   000121 IND         0000 I   000044 INOFF
```

```
0000 I 000005 IP        0000 I 000051 IPHASE    0000 I 000063 IPIV      0000 I 000054 IPRINT    0000 I 000114 IS
0000 I 000100 ISPIV     0000 I 000031 ITC       0000 I 000065 ITER      0000 I 000120 ITER1     0000 I 000030 ITP
0000 I 000020 IWARN     0000 I 000032 IX        0000 I 000104 IY        0000 I 000106 I7        0000 I 000055 I1
0000 I 000111 I1X       0000 I 000050 I2        0000 I 000057 I3        0000 I 000056 I4        0000 I 000036 J
0000 I 000112 JCPIV     0000 I 000064 JPIV      0000 I 000101 JSPIV     0000 I 000102 JTRUE     0000 I 000061 JY
0000 I 000052 J1        0000 I 000017 KFLAG     0000 I 000117 KH        0000 I 000041 L1OFF     0000 I 000046 L10
0000 I 000042 L2OFF     0000 I 000047 L20       0000 I 000043 L3OFF     0000 I 000022 M         0000 I 000023 ME
0000 I 000025 MG        0000 I 000024 ML        0000 I 000053 MOBJ      0000 I 000052 MX        0000 I 000027 N
0000 I 000045 NOTOFF    0000 I 000122 NOX       0000 I 000123 N1        0000 I 000124 N2        0000 R 000066 OBJ
0000 R 000076 PIV       0000 R 000073 PIVM      0000 R 000113 PIVX      0000 R 000012 PR        0000 R 000107 Q
0000 R 000105 Q1        0000 R 000075 ROBJ      0000 R 000110 SPIV      0000 R 000034 XMAX      0000 R 000035 Y
```

```
00101     1*        SJBROUTINE SIMPLX (A,MA,MT,NT,L,X,TOLP,KOP)                  S10000
00103     2*        DIMENSION A(MA,1),L(1),X(1),IB(5),IP(5),PR(5)                S10100
00104     3*        KFLAG = 0                                                    S10200
00105     4*        IWARN = 0                                                    S10300
00106     5*        ASTAR = '*'                                                  S10400
00107     6*        M=NT-2                                                       S10500
00110     7*        ME = M-ML-MG                                                 S10600
00111     8*        DO 11 I=1,M                                                  S10700
00114     9*        IF (A(I,NT) .GE. 0.) GO TO 11                                S10800
00115    10*        WRITE (6,9020)                                               S10900
00120    11*        L(3) = 2                                                     S11000
00121    12*        RETJRN                                                       S11100
00122    13*     11 CONTINUE                                                     S11200
00124    14*        ML=L(1)                                                      S11300
00125    15*        MG=L(2)                                                      S11400
00126    16*        L(2) = 0                                                     S11500
00127    17*        IF (L(3) .EQ. 0) GO TO 4                                     S11600
00131    18*        DO 1 I=3,13                                                  S11700
00134    19*      1 L(I)=0                                                       S11800
00135    20*        L(4)=1                                                       S11900
00137    21*        L(5)=1                                                       S12000
00140    22*        L(8) = 1                                                     S12100
00141    23*        L(12) = 1                                                    S12200
00142    24*        L(13)=1                                                      S12300
00143    25*        L(14)=1                                                      S12400
00144    26*      4 N=NT-1                                                       S12500
00145    27*        ITP=N-KOP                                                    S12600
00146    28*        ITC=KOP+ML+MG                                                S12700
00147    29*        IX=1                                                         S12800
00147    30*   C    IF (L(14) .GT. 0) WRITE(6,9016)                              S12900
00150    31*        L(3)=0                                                       S13000
00151    32*        EPS=TOLP                                                     S13100
00152    33*        IF (EPS.GT.0.0)  GO TO 10                                    S13200
00154    34*        XMAX=1.0E-5                                                  S13300
00155    35*        IF (EPS.LT.0)  XMAX=-EPS                                     S13400
00157    36*        Y=0.0                                                        S13500
00160    37*        DO 6 I=1,M                                                   S13600
00163    38*        DO 6 J=1,N                                                   S13700
00166    39*        IF (A(I,J) .LT. 0.) GO TO 5                                  S13800
00170    40*        Y = Y + A(I,J)                                               S13900
00171    41*        GO TO 6                                                      S14000
00172    42*      5 Y = Y-A(I,J)                                                 S14100
00173    43*      6 CONTINUE                                                     S14200
```

```
00176          EPS=XMAX*Y/(M*N)
00177       10 CEPS = 10.*EPS
00200          L(1) = 0
00201          DO 12 I=1,M
00204          DO 12 J=1,N
00207          AA = ABS(A(I,J))
00210          IF (AA .LE. 0 .OR. AA .GT. EPS) GO TO 12
00212          L(1) = L(1) + 1
00213       12 CONTINUE
00216          IF (L(4) .NE. 0) WRITE (6,9000) EPS,CEPS,L(1)
00216    C
00216    C * * CALCULATE L VECTOR OFFSETS AND INITIALIZE L VECTOR.
00216    C
00224          L1OFF=14
00225          L2OFF=L1OFF+N
00226          L3OFF=L2OFF+M
00227          INOFF=L3OFF+MG
00230          NOTOFF=INOFF+M
00231          DO 20  J=1,N
00234          L(L1OFF+J)=J
00235       20 L(NOTOFF+J)=J
00237          DO 25  I=1,M
00242          L(L2OFF+I)=I
00243       25 L(INOFF+I)=I+N
00245          L10=N
00246          L20=M
00247          IF (ML.EQ.M)  GO TO 2000
00251          IF (MG.EQ.0)  GO TO 32
00253          DO 30 IG=1,MG
00256       30 L(L3OFF+IG)=1
00260       32 IPHASE=1
00260    C
00260    C * * CONSTRUCT ARTIFICIAL OBJECTIVE FUNCTION.
00260    C
00261          MX=ML+1
00262          DO 35  J=1,NT
00265          A(MT,J)=0.0
00266          DO 35  I=MX,M
00271       35 A(MT,J)=A(MT,J)+A(I,J)
00274          MOBJ=M+2
00275          IPRINT=6
00276          GO TO 2001
00276    C
00276    C * * THE FOLLOWING SECTION IS ENTERED IN PHASE 1 WHEN ALL COEFFICIENTS
00276    C * * IN THE OBJECTIVE FUNCTION ROW ARE NON-POSITIVE. FEASIBILITY AND
00276    C * * POSSIBLE COMPUTATIONAL INCONSISTENCIES ARE INVESTIGATED.
00276    C
00277       50 IF (A(MT,NT) .GT. CEPS) GO TO 90
00301          DO 55  I1=1,L20
00304          I4=I1
00305          I3=L(L2OFF+I1)
00306          I=L(INOFF+I3)
00307          IF (I.GT.N+ML+MG)  GO TO 60
00311          IF(I.LE.N+ML)  GO TO 55
00313          I2=L(L3OFF+I-N-ML)
00314          IF (I2.EQ.1)  GO TO 60
00315       55 CONTINUE
00320          IF (A(MT,NT) .GT. EPS .OR. IWARN .EQ. 1) L(3) = 1
```

```
      IF (L(3) .EQ. 1 .AND. L(14) .NE. 0) WRITE (6,9019) (ASTAR,
     * I=1,240)
C
C * * COMPUTATIONAL INCONSISTENCY IMPLIES THE ARTIFICIAL OBJECTIVE
C * * FUNCTION AND/OR AT LEAST ONE ARTIFICIAL VARIABLE HAVE ABSOLUTE
C * * VALUE BETWEEN EPS AND CEPS.
C
      IX=4
      GO TO 2007
   60 IF (A(I3,NT) .GT. CEPS .OR. A(I3,NT) .LT. -CEPS) GO TO 90
      IF (L10 .LE. 0) GO TO 66
C
C * * THERE IS AN ARTIFICIAL VARIABLE IN THE BASIS WITH
C * * A VALUE LESS THAN OR EQUAL TO CAPITAL EPSILON.
C * * THERE WILL BE AN ATTEMPT TO FIND AN ACCEPTABLE
C * * PIVOT ELEMENT.
C
   61 DO 65 J=1,L10
      JX=L(L1OFF+J)
      IF (A(I3,JX).GT.EPS.OR.A(I3,JX).LT.-EPS)  GO TO 75
   65 CONTINUE
C
C * * IF THERE EXISTS NO ENTRY GREATER IN MAGNITUDE THAN EPSILON, ROW I3
C * * REPRESENTS A REDUNDANT CONSTRAINT.
C
   66 L20=L20-1
      DO 70 I1=I4,L20
      L(L2OFF+I1)=L(L2OFF+I1+1)
   70 CONTINUE
      IF (A(I3,NT) .GT. EPS) IWARN = 1
      A(I3,NT)=0.0
      GO TO 50
   75 IF (A(I3,JX).GT.0.0)  GO TO 85
      DO 80  J1=1,N
      A(I3,J1)=-A(I3,J1)
   80 CONTINUE
   85 A(I3,NT) = 0.
C
C * * THE ARTIFICIAL CAN AND WILL BE REMOVED FROM THE BASIS.
C
      IPIV = I3
      JPIV=JX
      GO TO 5400
C
C * * THE PROBLEM IS INFEASIBLE.
C
   90 IF (L(14) .EQ. 0) GO TO 91
      WRITE (6,9014) (ASTAR,I=1,120),XMAX
      WRITE (6,9018) ITER,IPHASE,A(MOBJ,NT),(ASTAR,I=1,120)
   91 L(3) = 2
      IX=3
      IF (L(14) .NE. 0) KFLAG = 1
      GO TO 5600
 2000 IPHASE=2
      IX=1
      OBJ=-A(M+1,NT)
      MOBJ=M+1
      IPRINT=10
```

```
      L(2) = L(2)+ 1
 2001 ITER=0
      IC1=1
      IC2=1
      IABC=0
      IABC1=0
      PIVM=1.0 E 8
      IMP=10000
 2002 GO TO 5000
 2005 IF (XMAX.GE.EPS)  GO TO 5100
      IX=IPHASE+1
      IF (IPHASE .EQ. 1) GO TO 50
 2007 IF (IABC.EQ.0.AND.IABC1.EQ.0)  GO TO 2006
      IF (L(5) .EQ. 0) GO TO 2006
      WRITE (6,9007) IPHASE,IABC,IABC1
 2006 IF (L(IPRINT+2) .NE. 0) GO TO 2008
      IF (L(14) .EQ. 0 .OR. L(3) .LE. 1) GO TO 2010
 2008 WRITE (6,9008) (ASTAR,I=1,120),IPHASE,OBJ,ITER
      IF (ITER .EQ. 0) GO TO 2009
      WRITE (6,9001) PIVM,IMP
 2009 ROBJ = -A( I+1,NT)
      IF (IPHASE .EQ. 1) WRITE (6,9003) ROBJ
      WRITE (6,9006) (ASTAR,I=1,120)
 2010 IF (L(IPRINT+3) .NE. 0) KFLAG = 1
      IF(IPHASE.EQ.1.AND.KFLAG.EQ.0) GO TO 5550
      GO TO 5600
 2012 RETURN
 2015 IF (IPIV .EQ. 0) GO TO 5344
      IF (PIV.GT.CEPS)  GO TO 5400
      IF (IBOUND .EQ. 0) GO TO 5300
      IF (PIV .GT. EPS) GO TO 5400
      ISPIV = IPIV
      JSPIV = JPIV
      GO TO 5335
 2025 ITER=ITER+1
      L(2) = L(2)+ 1
      IC1=IC1+1
      IC2=IC2+1
      GO TO 2002
C***********************************************************************
C                                                                      *
C     FIND THE MAXIMUM COEFFICIENT IN THE OBJECTIVE FUNCTION ROW.      *
C                                                                      *
C***********************************************************************
 5000 XMAX=0.0
      JPIV=0
      IF (L10.LT.1) GO TO 5030
      XMAX=-1.0E35
      IBOUND=0
      DO 5025 J=1,L10
      JX=L(L10FF+J)
      IF (A(MOBJ,JX).LE.XMAX) GO TO 5025
      IF (KOP.LE.0) GO TO 5020
      JTRUE=L(NOTOFF+JX)
      IF (JTRUE.GT.ITC) GO TO 5005
      ILOOK=ITP+JTRUE
      GO TO 5010
 5005 IF (JTRUE.LE.ITP.OR.JTRUE.GT.ITP+ITC) GO TO 5020
```

```
      ILOOK=JTRUE-ITP
 5010 DO 5015 I=1,M
      IF (L(INOFF+I).EQ.ILOOK) GO TO 5025
 5015 CONTINUE
 5020 JPIV=JX
      XMAX=A(MOBJ,JX)
 5025 CONTINUE
 5030 JSPIV=JPIV
      GO TO 2005
C***********************************************************************
C                                                                      *
C     FIND THE MINIMUM QUOTIENT.                                       *
C                                                                      *
C***********************************************************************
 5100 IPIV=0
      PIV=0.0
      IF (L20.LT.1)  GO TO 5175
      DO 5105  I=1,L20
      IY = L(L2OFF+I)
      IF (A(IY,JPIV) .GT. 0.0) GO TO 5110
 5105 CONTINUE
      GO TO 5175
 5110 Q1 = A(IY,NT)/A(IY,JPIV)
      IPIV = IY
      PIV=A(IPIV,JPIV)
      IZ=I+1
      IF (IZ.GT.L20)  GO TO 5175
      DO 5170  I=IZ,L20
      IY = L(L2OFF+I)
      IF (A(IY,JPIV) .LE. 0.0) GO TO 5170
      Q = A(IY,NT)/A(IY,JPIV)
      IF (Q.GE.Q1)  GO TO 5150
      Q1=Q
      GO TO 5155
 5150 IF (Q.GT.Q1)  GO TO 5170
      IF (A(IY,JPIV) .LE. A(IPIV,JPIV)) GO TO 5170
 5155 IPIV = IY
      PIV=A(IPIV,JPIV)
 5170 CONTINUE
 5175 GO TO 2015
C***********************************************************************
C                                                                      *
C     FIND AN ALTERNATE PIVOT ELEMENT GREATER THAN CEPS.               *
C                                                                      *
C***********************************************************************
 5300 SPIV=PIV
      ISPIV=IPIV
      JSPIV=JPIV
      DO 5330 J=1,L10
      J1=L(L1OFF+J)
      I1X=0
      Q1=1.0E20
      IF (A(MOBJ,J1).LE.EPS)  GO TO 5330
      IF (KOP.LE.0) GO TO 5315
      JTRUE=L(NOTOFF+J1)
      IF (JTRUE.GT.ITC) GO TO 5305
      ILOOK=ITP+JTRUE
      GO TO 5310
```

```
00701    5305 IF (JTRUE.LE.ITP.OR.JTRUE.GT.ITP+ITC) GO TO 5315             S37500
00703         ILOOK=JTRUE-ITP                                              S37600
00704    5310 DO 5312 I=1,M                                                S37700
00707         IF (L(INOFF+I).EQ.ILOOK) GO TO 5330                          S37800
00711    5312 CONTINUE                                                     S37900
00713    5315 DO 5320 I=1,L20                                              S38000
00715         I1=L(L20FF+I)                                                S38100
00717         IF (A(I1,J1).LE.0.0)  GO TO 5320                             S38200
00721         Q=A(I1,NT)/A(I1,J1)                                          S38300
00722         IF (Q.GT.Q1)  GO TO 5320                                     S38400
00724         IF (Q.GE.Q1.AND.A(I1,J1).LE.PIV)  GO TO 5320                 S38500
00726         Q1=Q                                                         S38600
00727         I1X=I1                                                       S38700
00730    5320 CONTINUE                                                     S38800
00732         IF (I1X.EQ.0.OR.A(I1X,J1).LE.PIV)  GO TO 5330                S38900
00734         PIV=A(I1X,J1)                                                S39000
00735         IPIV=I1X                                                     S39100
00735         JPIV=J1                                                      S39200
00737         Y=PIV/EPS                                                    S39300
00740         IF (PIV.GT.CEPS)  GO TO 5350                                 S39400
00742    5330 CONTINUE                                                     S39500
00744         IF (PIV.GT.EPS)  GO TO 5345                                  S39600
00746    5335 A(ISPIV,JSPIV) = 0.                                          S39700
00747         IBOUND = 1                                                   S39800
00750         JPIV = JSPIV                                                 S39900
00751         GO TO 5100                                                   S40000
00751   C                                                                  S40100
00751   C * * THE PROBLEM IS UNBOUNDED.                                    S40200
00751   C                                                                  S40300
00752    5344 L(3) = 3                                                     S40400
00753         IX=3                                                         S40500
00754         JPIV = JSPIV                                                 S40600
00755         JCPIV = L(NOTOFF+JPIV)                                       S40700
00756         IF (L(14) .EQ. 0) GO TO 5600                                 S40800
00760         KFLAG = 1                                                    S40900
00761         WRITE (6,9015) (ASTAR,I=1,120),JCPIV                         S41000
00770         WRITE (6,9018) ITER,IPHASE,OBJ,(ASTAR,I=1,120)               S41100
01001         GO TO 5600                                                   S41200
01002    5345 IF (PIV.GT.CEPS)  GO TO 5350                                 S41300
01004         IF (L(5) .NE. 0) WRITE (6,9005) ITER,IPHASE,PIV              S41400
01012         IABC1=IABC1+1                                                S41500
01013    5350 IABC = IABC+1                                                S41600
01014         GO TO 5400                                                   S41700
01014   C******************************************************************** S41800
01014   C                                                                  * S41900
01014   C      EXCHANGE A BASIC AND A NON-BASIC VARIABLE, I.E. PIVOT.      * S42000
01014   C                                                                  * S42100
01014   C******************************************************************** S42200
01015    5400 PIVX=1.0/A(IPIV,JPIV)                                        S42300
01016         IF (A(IPIV,JPIV).GE.PIVM) GO TO 5401                         S42400
01020         PIVM=A(IPIV,JPIV)                                            S42500
01021         IMP=ITER+1                                                   S42600
01022    5401 DO 5415 I=1,MOBJ                                             S42700
01025         IF (I.EQ.IPIV)  GO TO 5415                                   S42800
01027         A(I,JPIV) = -PIVX*A(I,JPIV)                                  S42900
01030         DO 5410 JX = 1,L10                                           S43000
01033         J = L(L10FF+JX)                                              S43100
01034         IF (J.EQ.JPIV)  GO TO 5410                                   S43200
```

```
01036          A(I,J) = A(I,J)+A(I,JPIV)*A(IPIV,J)
01037     5410 CONTINUE
01041          A(I,NT) = A(I,NT) + A(I,JPIV)*A(IPIV,NT)
01042          IF (A(I,NT).GE.0.0)  GO TO 5415
01044          IF (I.LE.M)  A(I,NT)=0.0
01046     5415 CONTINUE
01050          DO 5420 JX = 1,L10
01053          J = L(L1OFF+JX)
01054          IF (J.NE.JPIV)  A(IPIV,J)= PIVX*A(IPIV,J)
01056     5420 CONTINUE
01060          A(IPIV,NT) = PIVX*A(IPIV,NT)
01061          OBJ=-A(NOBJ,NT)
01062          A(IPIV,JPIV)=PIVX
01063          IS=L(INOFF+IPIV)
01064          L(INOFF+IPIV)=L(NOTOFF+JPIV)
01065          L(NOTOFF+JPIV)=IS
01066          IENT = L(INOFF+IPIV)
01067          ILVE = IS
01070          IB(1) = ' X(    '
01071          IF (IENT .LE. N) GO TO 5425
01073          IB(1) = ' S(    '
01074          IENT = IENT-N
01075     5425 IF (IPHASE .EQ. 1) GO TO 5428
01077          IB(2) = ' X(    '
01100          IF (ILVE .LE. N) GO TO 5460
01102          IB(2) = ' S(    '
01103          ILVE = ILVE-N
01104          GO TO 5460
01105     5428 IF (IS .LE. N+ML+MG) GO TO 5440
01105     C
01105     C * * TRANSFER FOR NON-EQUALITY CONSTRAINT.
01105     C
01107          IB(2) = ' A(    '
01110          ILVE = ILVE-N-ML
01111          J=1
01112          DO 5430 I=1,L10
01115          I1=L(L1OFF+I)
01116          IF (I1.EQ.JPIV)  J=J+1
01120          L(L1OFF+I)=L(L1OFF+J)
01121     5430 J=J+1
01123          L10=L10-1
01124          GO TO 5460
01125     5440 IB(2) = ' S(    '
01126          IF (IS .LE. N) IB(2) = ' X(    '
01130          IF (IS .GT. N+ML) GO TO 5445
01132          IF (IS .GT. N) ILVE = ILVE-N
01134          GO TO 5460
01134     C
01134     C * * TRANSFER FOR SLACK.
01134     C
01135     5445 KH = L(NOTOFF+JPIV)-ML-N
01136          IF (L(L3OFF+KH) .NE. 0) GO TO 5448
01140          ILVE = ILVE-N
01141          GO TO 5460
01142     5448 IB(2) = ' A(    '
01143          ILVE = ILVE-N-ML
01144          L(L3OFF+KH)=0
01145          A(MT,JPIV)=A(MT,JPIV)+1
```

```
      DO 5450  I=1,MT
      A(I,JPIV)=-A(I,JPIV)
 5450 CONTINUE
 5460 IF (IX .NE. 2) IX = 1
      GO TO 5500
C*********************************************************************
C                                                                    *
C     PRINT ITERATION SUMMARIES.                                     *
C                                                                    *
C*********************************************************************
 5500 IF (L(IPRINT) + L(IPRINT+1) .EQ. 0) GO TO 5550
      ITER1 = ITER+1
      IND=0
      IF (IC1 .LT. L(IPRINT) .OR. L(IPRINT) .EQ. 0) GO TO 5530
      IC1=0
      WRITE (6,9010) IPHASE,ITER1,PIV,OBJ,IB(1),IENT,IB(2),ILVE
      GO TO 5535
 5530 IND=1
 5535 IF (IC2 .LT. L(IPRINT+1) .OR. L(IPRINT+1) .EQ. 0) GO TO 5550
      KFLAG = 1
      IC2=0
      IF (IND.EQ.0)  GO TO 5540
 5538 WRITE (6,9010) IPHASE,ITER1,PIV,OBJ,IB(1),IENT,IB(2),ILVE
 5540 GO TO 5600
 5550 GO TO (2025,50,2012,2000),IX
C*********************************************************************
C                                                                    *
C     GET X AND PRINT BASIS IF KFLAG NE 0.                           *
C                                                                    *
C*********************************************************************
 5600 NOX = N+M+MG
      DO 5605 I2=1,NOX
 5605 X(I2) = -.001*EPS
      DO 5610 I2=1,M
      IND = L(INDFF+I2)
      IF (IND .LE. N+ML) GO TO 5609
      IF (IND .GT. N+ML+MG) GO TO 5608
      KH = IND-N-ML
      IF (L(L3OFF+KH) .EQ. 0) GO TO 5609
 5608 X(IND+MG) = A(I2,NT)
      GO TO 5610
 5609 X(IND) = A(I2,NT)
 5610 CONTINUE
      X(NOX+1) = OBJ
      IF (KFLAG .EQ. 0) GO TO 5550
      KFLAG = 0
      DO 5606 I=1,5
      IB(I) = ' X(    '
 5606 CONTINUE
      WRITE (6,9012)
      I3 = 0
      DO 5625 I2=1,N
      IF (X(I2) .LT. 0.) GO TO 5620
      I3 = I3+1
      PR(I3) = X(I2)
      IP(I3) = I2
 5620 IF (I3 .LT. 5 .AND. I2 .LT. N) GO TO 5625
      IF (I3 .EQ. 0) GO TO 5625
```

```
      WRITE (6,9011) (IB(J),IP(J),PR(J),J=1,I3)
      I3 = 0
 5625 CONTINUE
      N1 = N+1
      N2 = N+ML+MG
      IF (N2 .LT. N1) GO TO 5637
      I3 = 0
      DO 5628 I=1,5
      IB(I) = ' S(    '
 5628 CONTINUE
      WRITE (6,9017)
      DO 5635 I2=N1,N2
      IF (X(I2) .LT. 0.) GO TO 5630
      I3 = I3+1
      PR(I3) = X(I2)
      IP(I3) = I2-N1+1
 5630 IF (I3 .LT. 5 .AND. I2 .LT. N2) GO TO 5635
      IF (I3 .EQ. 0) GO TO 5635
      WRITE (6,9011) (IB(J),IP(J),PR(J),J=1,I3)
      I3 = 0
 5635 CONTINUE
 5637 N1 = N2+1
      N2 = NOX
      IF (N2 .LT. N1) GO TO 5550
      I3 = 0
      DO 5638 I=1,5
      IB(I) = ' A(    '
 5638 CONTINUE
      WRITE (6,9017)
      DO 5645 I2=N1,N2
      IF (X(I2) .LT. 0.) GO TO 5640
      I3 = I3+1
      PR(I3) = X(I2)
      IP(I3) = I2-N1+1
 5640 IF (I3 .LT. 5 .AND. I2 .LT. N2) GO TO 5645
      IF (I3 .EQ. 0) GO TO 5645
      WRITE (6,9011) (IB(J),IP(J),PR(J),J=1,I3)
      I3 = 0
 5645 CONTINUE
      GO TO 5550
 9000 FORMAT(' EPSILON = 'G13.6,'  CAPITAL EPSILON = 'G13.6,'  'I5,' NON
     *-ZERO ENTRIES ARE EFFECTIVELY EQUAL TO ZERO.')
 9001 FORMAT(' *'14X'MINIMUM PIVOT WAS'G12.5,' AT ITERATION'I4,'.'57X'*'
     *)
 9003 FORMAT(1H+65X'REAL OBJECTIVE FUNCTION ='G15.8)
 9005 FORMAT('0* * WARNING * *  SMALL PIVOT ELEMENT AT ITERATION'I4,' OF
     * PHASE'I2,'. PIVOT ='G14.7)
 9006 FORMAT(' *'118X'*'/' '120A1)
 9007 FORMAT(//'0* * WARNING * *  IN PHASE'I2,', THERE HAVE BEEN'I3,' AT
     *TEMPTS TO FIND AN ALTERNATE PIVOT ELEMENT.'//19X'OF THESE,'I3,' FO
     *UND NO ALTERNATE ELEMENT GREATER THAN CAPITAL EPSILON.')
 9008 FORMAT(/////' '120A1/' *'118X'*'/' *'14X'END OF PHASE'I2,'.    OBJ
     *ECTIVE FUNCTION ='G18.8,4X'THERE WERE'I4,' ITERATIONS.'17X'*')
 9010 FORMAT (///'0PHASE',I2,' ITERATION',I4,'.  PIVOT=',G13.6,'   OBJEC
     *TIVE FUNCTION='G15.8,'   ',A3,I3,') ENTERED THE BASIS,',A3,I3,') LE
     *FT.')
 9011 FORMAT(5(3XA3,I3,')='F12.6))
 9012 FORMAT('0BASIC VARIABLES'/)
```

```
01441   9014 FORMAT(/////' '120A1/' *'118X'*'/' *  ERROR - THE PROBLEM IS INFEA  S60700
01441       *SIBLE. THE CONSTRAINTS ASSOCIATED WITH THE ARTIFICIAL VARIABLES BE  S60800
01441       *LOW ARE INCONSISTENT. *'/' *  IF NONE APPEAR, NUMERICAL DIFFICULTI  S60900
01441       *ES HAVE BEEN ENCOUNTERED. THE LARGEST ENTRY IN THE OBJECTIVE FUNCT  S61000
01441       *ION ROW','7X,'*'/' *  IS ',G12.5,101X,'*')                          S61100
01442   9015 FORMAT(/////' '120A1/' *'118X'*'/' *  ERROR - THE PROBLEM IS UNBOU  S61200
01442       *NDED. THE VARIABLE X(''I3,') CAN ASSUME AN ARBITRARILY LARGE VALUE,  S61300
01442       * THEREBY YIELDING'7X'*'/' *  AN ARBITRARILY LARGE VALUE OF THE OBJ  S61400
01442       *ECTIVE FUNCTION.'63X'*')                                            S61500
01443   9016 FORMAT (1H1)                                                         S61600
01444   9017 FORMAT (1H )                                                         S61700
01445   9018 FORMAT(' *'118X'*'/' *  THE ERROR WAS DETECTED AT ITERATION'I4,' O  S61800
01445       *F PHASE'I2,'. AT THAT TIME THE OBJECTIVE FUNCTION VALUE WAS'G15.8,  S61900
01445       *4X'*'/' *  AND THE FOLLOWING VARIABLES WERE BASIC.'77X'*'/' *'118X  S62000
01445       *'*'/' '120A1)                                                       S62100
01446   9019 FORMAT(///' '120A1/' *'118X'*'/' *  WARNING - COMPUTATIONAL INCONS  S62200
01446       *ISTENCY INDICATED AT THE END OF PHASE 1. THE ALGORITHM WILL CONTIN  S62300
01446       *UE WITH PHASE 2,'4X'*'/' *  BUT THE USER IS ADVISED TO CRITICIZE T  S62400
01446       *HE RESULTS.'67X'*'/' *'118X'*'/' '120A1)                            S62500
01447   9020 FORMAT('0* *  AT LEAST ONE ELEMENT OF THE RIGHT HAND SIDE COLUMN I  S62600
01447       *S LESS THAN ZERO. SUBROUTINE TERMINATES.')                          S62700
01450        END                                                                 S62800

  END OF UNIVAC 1108 FORTRAN V COMPILATION.      0 *DIAGNOSTIC* MESSAGE(S)
PHASE 1 TIME =   1 SEC.
PHASE 2 TIME =   0 SEC.
PHASE 3 TIME =   2 SEC.
PHASE 4 TIME =   0 SEC.
PHASE 5 TIME =   1 SEC.
PHASE 6 TIME =   2 SEC.

TOTAL COMPILATION TIME =   6  SEC
```

APPENDIX D: TIMING CONSIDERATIONS

In order to provide the reader with some idea of the time involved in solving linear programming problems with RVSMPX and SIMPLX, we include the following table of problems that were run on the National Bureau of Standards' UNIVAC 1108. Each of the problems was randomly generated in such a way as to be bounded and feasible; and for each problem, all constraints were "greater than" constraints, so that a full Phase I was required.

Note that, as n increases from 20 to 120 while m is held fixed at 20, RVSMPX becomes faster than SIMPLX. The same is true for m held fixed at 50 and n increasing from 75 to 200. This illustrates the concept alluded to in section 1, and discussed at length in [7]. Basically, the concept is that for $n > 3m$, the revised simplex method unequivocally appears to be better computationally. Futhermore, in [7], Wagner argues that even for smaller n the revised simplex method is more desirable than the standard simplex method.

TABLE I: RESULTS OF SAMPLE RUNS

		RVSMPX				SIMPLX		
m	n	Elapsed Time	Number of Iterations	Time per Iteration		Elapsed Time	Number of Iterations	Time per Iterations
10	20	.3984	29	.0137	*	.3686	29	.0127
10	20	.3876	28	.0138		.3338	28	.0119
15	20	.5826	29	.0201	*	.4580	29	.0158
20	20	1.0838	44	.0246	*	.7898	44	.0180
20	30	1.9954	74	.0270	*	1.6608	74	.0224
20	60	2.5174	74	.0340		2.9472	74	.0398
20	80	4.3960	112	.0393	*	5.6848	112	.0508
20	120	3.3164	66	.0502	*	5.0066	66	.0759
50	75	43.5036	322	.1351		37.1222	329	.1128
50	150	58.7630	335	.1754	*	73.0950	339	.2156
50	200	87.8642	459	.1914	*	133.0888	474	.2808

Note: the times listed above are in seconds.

DEPARTMENT OF COMMERCE

NATIONAL BUREAU OF STANDARDS