THE SOLUTION OF LARGE SYSTEMS OF ALGEBRAIC EQUATIONS

BY

JOHN M. PAVKOVICH

TECHNICAL REPORT NO. 33 (2)
DECEMBER 6, 1963

PREPARED UNDER CONTRACT Nonr-225(37)
(NR-044-211)
FOR
OFFICE OF NAVAL RESEARCH

COMPUTER SCIENCE DIVISION
School of Humanities and Sciences
STANFORD UNIVERSITY

THE SOLUTION OF LARGE SYSTEMS OF ALGEBRAIC EQUATIONS

by

John M. Pavkovich

TECHNICAL REPORT NO. 33
December 6, 1963

PREPARED UNDER CONTRACT Nonr-225(37)
(NR-044-m)
OFFICE OF NAVAL RESEARCH

COMPUTER SCIENCE DIVISION
SCHOOL OF HUMANITIES AND SCIENCES
STANFORD UNIVERSITY

THE SOLUTION OF LARGE SYSTEMS OF ALGEBRAIC EQUATIONS

by

John M. Pavkovich

The solution of a system of linear algebraic equations using a computer is not a difficult problem as long as the equations are not ill-conditioned and all of the coefficients can be stored in the computer. However, when the number of coefficients is so large that supplemental means of storage, such as magnetic tape, are required, the problem of solving the system in an efficient manner increases considerably. This paper describes a method of solution whereby such systems of equations can be solved in an efficient manner. The problems associated with ill-conditioned systems of equations are not discussed.

The method described on the following pages was implemented on the IBM 7090 at Stanford for equations with complex coefficients. Although all figures quoted related to tape movement and arithmetic speed are for this computer, the ideas behind the method are applicable to any computer which has the ability to read tape, write tape, and compute simultaneously.

Consider the system of equations

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N &= y_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N &= y_2 \\ &\vdots \\ a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N &= y_n \end{aligned} \qquad (1)$$

The first step in the solution is to normalize the system, i.e., to multiply each equation by a factor which makes the magnitude of the largest coefficient in that equation approximately equal to 1. In the case of a binary machine, this factor should be a power of two so that no significant figures are lost during this process. The reason for normalizing the system of equations is to increase the effectiveness of pivoting (interchanging equations during the solution process) and thus minimize the difficulties associated with roundoff error.

The method used to solve the system of equations is basically Gauss's method with partial pivoting. Briefly, this method is performed as follows. The first column of the system of equations is scanned to find the largest coefficient of x_1 in absolute value. The equation containing this coefficient is then interchanged with the first equation (or row). A suitable multiple of this new first equation is then subtracted from each of the other equations in order to eliminate x_1 from each of them. This process is then repeated using the coefficients of x_2 and Eqs. 2 through N. Coefficients a_{22} through a_{N2} are examined to determine the largest in absolute value. The equation containing this coefficient is interchanged with the second equation and a suitable multiple is subtracted from each of the remaining equations. This same process of eliminating one variable at a time from all succeeding equations is repeated again and again until a system of equations is obtained in which the i-th equation contains only the unknowns x_i through x_N. Such a system of equations is

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1N}x_N &= y_1 \\
a_{22}x_2 + a_{23}x_3 + \cdots + a_{2N}x_N &= y_2 \\
a_{N-1,N-1}x_{N-1} + a_{N-1,N}x_N &= y_{N-1} \\
a_{NN}x_N &= y_N
\end{aligned} \tag{2}$$

This so-called reduced system can now be solved by starting at the bottom and solving first for x_N, then x_{N-1} using Eq. (N-1) and the now known value of x_N. This process, known as backsolving, is continued until the entire solution is obtained.

Gauss's method as described above is inefficient when applied to a system of equations too large to fit in core storage. The reason is that as each variable is eliminated from all subsequent equations, all the coefficients of these equations must be read from tape and the new coefficients written on tape. Moreover, while each coefficient is in core storage, it is used in only one arithmetic operation. What is needed is a method whereby numbers which must be read from tape and written on tape many times, are used in many arithmetic operations while they are in core storage. This can be accomplished by applying Gauss's method in a more subtle fashion in which successive columns of the reduced system are obtained rather than successive rows. This is achieved as follows.

Consider again the system of equations (1). Assume that it has been normalized. As with the ordinary form of Gauss's method, the first column of coefficients is examined to locate the first pivotal element, i.e., the numerically largest coefficient of x_1. The location of this element is recorded and this coefficient is interchanged with a_{11}. It is necessary to remember this interchange since this same interchange must be performed in all subsequent columns to accomplish the interchange of the first equation with the equation containing the largest coefficient of x_1. After the interchange has been performed, a_{11} is the first column of the reduced system of equations, and will not be involved in any more numerical operations until the backsolving is performed. The remaining elements in column 1, i.e., a_{21} through a_{N1}, are now divided by a_{11}. The result of this operation will be denoted by b_{i1} and these numbers will be referred to as multipliers since in terms of Gauss's method, b_{i1} represents the factor by which the first equation is multiplied when it is used to eliminate x_1 from the i-th equation.

Thus, we have

$$b_{i1} := \frac{a_{i1}}{a_{11}} \qquad (i := 2, 3, \ldots, N) \quad . \qquad (3)$$

The b_{i1}'s will be used in processing all of the remaining columns.

The reduction of the second column begins by performing the interchange associated with column 1. The coefficients a_{22} through a_{N2} are then processed using the relation

$$a_{i2}^{NEW} := a_{i2}^{OLD} - b_{i1}a_{12} \qquad (i := 2, \ldots, N) \quad . \qquad (4)$$

In terms of Gauss's method, these are the calculations which occur in the second column when x_1 is eliminated from Eqs. 2 through N. The elements a_{22} through a_{N2} are now examined to find the second pivotal element. Its location is recorded and it is interchanged with a_{22}. Elements a_{12} and a_{22} are now the second column of the reduced system of equations. Elements a_{32} through a_{N2} are divided by a_{22} to obtain the second column of multipliers. The second set of multipliers will be used to process all remaining columns.

The pattern for obtaining the successive columns of the reduced systems of equations is now established. Each column is taken in order and reduced using the interchanges and multiplier columns associated with all of the previous columns. The operation reducing the k-th column with the j-th column of multipliers is

$$a_{ik}^{NEW} := a_{ik}^{OLD} - b_{ij}a_{jk} \qquad (j < k;\ i := j + 1,\ j + 2,\ \ldots\ N) \qquad (5)$$

Note that (4) is just a special case of (5) with $j := 1$ and $k := 2$. After the k-th column has been processed using multiplier columns 1 through $k - 1$, elements a_{kk} through a_{Nk} are examined to find the k-th pivotal element. Its location is recorded and it is interchanged with a_{kk}. Elements $a_{k+1,\,k}$ through a_{Nk} are then divided by a_{kk} to obtain the k-th column of multipliers.

To obtain the complete reduced system of equations, the operations indicated above are carried out until all the columns have been reduced. The right-hand side is reduced in exactly the same way that the N-th column is reduced. The result of these operations will be a reduced system of equations of structure (2). Note that columns of multipliers will form a lower triangular matrix:

$$\begin{matrix} b_{21} & & & & \\ b_{31} & b_{32} & & & \\ b_{41} & b_{42} & b_{43} & & \\ \vdots & & & & \\ b_{N1} & b_{N2} & b_{N3} & \cdots & b_{N,N-1} \end{matrix}$$

Associated with each column of multipliers is an interchange which must be performed before that column of multipliers is applied.

It should be clear from the preceding discussion that the numbers which are used again and again in performing the reduction are the multipliers. Thus these numbers must be repeatedly read from tape if there is insufficient room for all of them in core storage. However, a little thought will show that it is permissible to process more than one column at a time with the same column of multipliers. This means that while a column of multipliers is in core storage, we should process enough columns with it to allow the next column of multipliers to be read from tape. With a judicious choice of the number of columns one chooses to reduce simultaneously, it is possible to overlap almost all tape movement with computing and still keep the amount of core storage required to a minimum.

For the IBM 7090, the number of columns, K, to be processed simultaneously can be arrived at as follows: The time required to perform arithmetic operations and to read or write tape are as follows:

Floating Multiplication	24μs (microseconds)
Floating Addition or Subtraction	14μs
Read and Write Tape (729-IV Tape Drive 556 Characters/Inch)	100μs/word
Pass a Record Gap	7300μs

From (5) it can be seen that it is necessary to perform one multiplication and one subtraction per multiplier element per column being processed. Here we are assuming the coefficients to be real. If each column of multipliers is written as one record, then the following relation is the criterion we wish to satisfy:

$$\text{Compute Time} \geq \text{Tape Read Time} \tag{7a}$$

$$K(M + 1)\,(24\mu s + 14\mu s) \geq M \cdot 100\mu s + 7300\mu s \quad , \tag{7b}$$

where K = numbers of columns being reduced simultaneously, and M = length of multiplier column being read. By solving for K, we find

$$K \geq 2.63\,(M/M{+}1) + 192/(M{+}1) \;. \tag{8}$$

From the above relation, we see that K should certainly be 3 or larger.

A choice of 6 or 8 would probably be the most reasonable since the record gaps would introduce some lost time only when the length of multipliers became less than 30 or 40. In the case of complex coefficients the calculation is quite similar. For the program written at Stanford, 4 columns were used.

If the ideas put forth thus far are implemented in a program, it would proceed as follows. Three tapes are required which will be denoted as below:

IT = Input Tape. This tape contains the matrix describing the system of equations to be solved. It is assumed that the system of equations has been normalized and that the matrix is stored by columns on this tape.

MT = Multiplier Tape. This tape will contain all of the multipliers at the conclusion of the reduction process.

RST = Reduced System Tape. This tape will contain the columns of the reduced system at the conclusion of the reduction process.

It will be seen that the program as described below possesses one major difficulty, namely, that there may be some delay while the MT tape rewinds. A method of overcoming this difficulty will be described subsequently. The program proceeds as follows:

1: Read the first K columns of the system of equations from IT into core storage.

2: Reduce these K columns until the first K columns of the reduced system and the first K columns of multipliers are obtained.

3: Write the K columns of multipliers on MT and rewind it.

4: Write the K columns of the reduced system on RST.

5: Read the next K columns of the system of equations from IT into core storage.

6: Reduce these K columns using the multipliers stored on MT. During this process, all of the multipliers which have been previously written on MT will be read.

7: Further reduce these K columns to obtain K more columns of the reduced system and K more columns of multipliers.

8: Write the K new columns of multipliers on MT and rewind it.
9: Write the K new columns of the reduced system on RST.
10: If more columns remain on IT, go to step 5.

To solve the system of equations for some particular right-hand side, one reduces this right-hand side by processing it with all of the multipliers on MT. The reduced system is then backsolved with this reduced RHS to obtain the solution. During the backsolution process it is necessary to backspace RST before reading each column since they are required in the reverse order from that on the tape. If one has to backsolve the system many times for many different right-hand sides, it is wise to write a tape of the reduced system matrix with the columns in the order in which they are required during backsolving. This can be done the first time the system is backsolved.

As stated previously, the program described above wastes considerable time waiting for the MT to rewind. However, this difficulty can be overcome by using extra multiplier tapes in such a fashion that a tape is always available with the correct column of multipliers ready to be read into core storage. One possible way of doing this using a total of three multiplier tapes will be described here. These three tapes are denoted MT, MT1, MT2. To be effective, this scheme requires two channels.

Tapes MT and MT1 are on Channel A and MT2 is on Channel B. Table I describes how the tapes are used. Here K, the number of columns reduced at one time, is 4. By studying Table I, it will be seen that MT contains approximately one-half of the multiplier columns. The remaining columns of multipliers are on either MT1 or MT2. Consider line 10 in Table I. At this point columns 1-12 have been reduced. Multiplier columns 1-8 are on MT and multiplier columns 9-12 are on MT1. Columns 13-16 are now read from the input tape and processed using multiplier columns 1-8. When this is complete, the rewinding of MT is initiated. Columns 13-16 are then further processed using multiplier columns 9-12. While each of these multiplier columns is in core storage, it is copied onto MT2. Since MT1 and MT2 are on different channels it is possible to read multipliers from MT1, write multipliers on MT2, and compute, all simultaneously. After multiplier columns 9-12 have been used, columns 13-16 are further processed to obtain columns 13-16 of the reduced system and multiplier columns 13-16. The columns of the reduced system are written on RST and multiplier columns 13-16 are written on MT2. The MT1 and MT2 are now rewound. At this point, the configuration of the tapes is that shown on line 13 of Table I.

The program is now ready to begin processing columns 17-20. These 4 columns are read from IT and processed using multiplier columns 1-8 from MT. Multipliers from MT2 are now used to process the 4 columns in core. While each multiplier column 9, 10, 11, and 12 is in core, it is copied onto MT. Since MT and MT2 are on different channels, there is no delay in the program. As soon as multiplier 12 has been written on MT, it is rewound. While multiplier columns 13-16 are in core, they are written on MT1. Columns 17-20 are then further processed to obtain 4 more columns of the reduced system and 4 more columns of multipliers. The 4 columns of the reduced system are written on RST and the 4 columns of multipliers are written on MT1. Tapes MT1 and MT2 are then rewound and the tapes are in the configurations indicated on line 16 of Table I. The reader should now be able to make his way through Table I.

When all the columns on IT have been processed, it is necessary to copy the multipliers from MT1 or MT2 onto MT if one wants one tape with all of the multiplier columns on it. This will delay the program slightly, but the delay is of little significance when compared to the time required for the entire reduction process.

The program written at Stanford performs the reduction as described above. It also has the capability to compute residues using double precision and iterate the solution to obtain more accurate results. Timing experiments were performed using this program and some representative results are indicated in Table II. A millisecond core clock on the IBM 7090 was used to measure the elapsed time so the measurements are quite accurate. It must be confessed, however, that the results are not exactly reproducible. The reasons for this are related to tape. The start and stop times of various tapes are probably not reproducible from one experiment to another. Also, any tape error further introduces differences since the program is delayed while the tape error is corrected.

TABLE I

	Columns Being Processed	Channel A MT	Channel A MT1 Reading	Channel A MT1 Writing	Channel B MT2 Reading	Channel B MT2 Writing
1		---		---	---	
2	1-4					
3		1-4		---	---	
4		1-4	---			---
5	5-8					
6		1-4	---			5-8
7		1-4		---	5-8	
8	9-12					
9		1-8		9-12	5-8	
10		1-8	9-12			---
11	13-16					
12		1-8	9-12			9-16
13		1-8		---	9-16	
14	17-20					
15		1-12		13-20	9-16	
16		1-12	13-16			---
17	21-24					
18		1-12	13-16			13-24
	ETC.					

Arrangement of Tape Storage During the Reduction Process

TABLE II

N	Reduction and Solution for 1 RHS (no Iterations)	Solution for 2nd RHS (no Iteration)	Iteration
40	11.9 sec	0.44 sec	0.70 sec
80	52.8 sec	2.47 sec	5.32 sec
120	149.6 sec	4.59 sec	10.93 sec
160	327.8 sec	7.33 sec	18.60 sec
320	2398.2 sec	24.57 sec	---

From the results of the timing experiments, it was possible to construct polynomials which give reasonably good estimates of the running time for solving a system of N equations. These polynomials are as follows:

(1) Reduction and solution for 1 RHS with no iterations:

$$T = (0.000068)N^3 + (0.0012)N^2 + (0.125)N + (0.425) \qquad (9a)$$

(2) Solution for a second RHS with no iterations:

$$T = (0.000195)N^2 + (0.0144)N + (0.078) \qquad (9b)$$

(3) Each iteration:

$$T = (0.000629)N^2 + (0.015)N + (0.095) \qquad (9c)$$

Using polynomial (9a), one estimates that the time required to solve 1000 simultaneous equations with complex coefficients would be about 19 hours. A program to solve equations with real coefficients would require about 30% of this time. Although the numerical operations would require only one-fourth as much time, there is no decrease in the amount of bookkeeping required.

In principle, the program written at Stanford is capable of solving 1000 or more simultaneous equations. However, the use of the present program to solve a system larger than about two or three hundred is rather risky, since the tape routines are not very sophisticated. In their present form, the tape routines make 10 attempts to correct writing errors and 10 attempts to correct reading errors. If the routines are unsuccessful in correcting the tape error, the program halts and the whole computation must be restarted from the beginning. It would be better if the program were able to salvage as much of the computation as possible after encountering bad tape. This could be accomplished by using an extra tape on which a copy of all multipliers and reduced columns was written, so that if any tape failures occurred, the program could continue after new tapes were mounted. This would increase the running time slightly, but it would be well justified by the increased reliability.

SUBROUTINE GAUSS and its associated subroutines are now described briefly and listed in either FORTRAN or FAP.

SUBROUTINE GAUSS

(NSYS, ISOLVD, KITER, EPS, ANS1, ANS2, RHS1, RHS2, KCOEF, KCOPY, KMULT, KT1, KT2, KT3, ISING)

Subroutine Gauss solves a system of up to 500 simultaneous algebraic equations with complex coefficients. The limit of 500 is determined only by the array size in the subroutine and one could increase the maximum allowable size by simply changing the DIMENSION statement and the IF statement which checks to see that the array size is not exceeded.

The arguments of subroutine Gauss are as follows:

NSYS = size of the system to be solved. If NSYS exceeds the array size, a message is printed on-line to save the tape containing the system of equations. The subroutine then rewinds the tape containing the system of equations and then pauses before calling EXIT.

ISOLVD = an integer variable used to indicate whether the system has been previously solved and that only a new right side is to be considered. If ISOLVD is equal to 1, the reduction process is not performed and the program assumes that the reduced system of equations and the multiplier matrix are available on tapes KT1 and KMULT respectively. If any iterations are required, the program also assumes that a copy of the matrix is available on tape KCOPY. If ISOLVD is not equal to 1, the entire reduction process is performed.

KITER = the maximum number of times the program is permitted to iterate and correct the solution. During the iteration process, the error is measured by the maximum change in any unknown divided by the maximum of the unknowns, i.e.,

$$\text{error} = \frac{\text{maximum } |\Delta X_i|}{\text{maximum } |X_i|} .$$

The iteration process stops as soon as either (1) the error is less than EPS, the accuracy criteria, (2) the error for the last iteration is greater than for the previous iteration, or (3) KITER iterations have been performed.

EPS = accuracy criteria for the iteration process. See KITER .

ANS1, ANS2 = one-dimensional arrays which represent the real and imaginary parts of the answer respectively.

RHS1, RHS2 = one-dimensional arrays which represent the real and imaginary part of the right-hand side respectively.

The following 6 arguments of SUBROUTINE GAUSS are logical tape numbers and the purpose of each is described below. In order to perform the reduction efficiently, the program requires that tape KT1 be on a channel different from the channel to which KMULT, KT2, and KT3 are attached. The program also requires that KCOEF and KCOPY be on different channels. If these restrictions are not met, the program prints out a message (off-line) and returns with ISING equal to 4.

KCOEF = the logical tape which contains the matrix describing the system of equations to be solved. The program assumes that the matrix has been previously normalized and that the matrix is stored on this tape by columns. Each column is written as one logical record by a statement of the form

```
WRITE TAPE NCOEF, (A1(K),A2(K),K=1,NSYS,1)
```

where A1 and A2 are the real and imaginary parts respectively of one column of matrix elements.

KCOPY = a logical tape on which SUBROUTINE GAUSS writes a copy of the matrix contained on tape KCOEF. During the iteration process, the entire matrix must be used in computing the residues. In order to overlap the tape reading with computing, it is necessary to have a copy of the matrix written by the I/O routines used by SUBROUTINE GAUSS.

KMULT = a logical tape used during the reduction process. At the conclusion of the reduction process, this tape will contain the multiplier matrix. This lower triangular matrix will have been written by the I/O routines used by GAUSS and thus this tape cannot be read by FORTRAN tape statements.

KT1 = a logical tape used during the reduction process. At the conclusion of the reduction process and the initial backsolving, this tape will contain the reduced system matrix. Again, this tape has been written by the I/O routines associated with SUBROUTINE GAUSS.

KT2, KT3 = logical tapes used by SUBROUTINE GAUSS. These tapes are used as scratch tapes during the reduction process.

ISING = an integer variable used to indicate the result achieved by SUBROUTINE GAUSS. ISING will normally be equal to 0. However, if during the reduction process a pivotal element is encountered which is less than 1.0 x 10^{-15} or greater than 1.0 x10^{+15}, ISING is set equal to 1 or 2 respectively and control is returned to the calling program. Also, as indicated previously, ISING is set equal to 4 if the channel requirements for the tapes are not met.

Several subroutines are used by SUBROUTINE GAUSS in solving the system of equations. The function, name, and argument list of each is as follows:

SUBROUTINE SAVEIT

This subroutine has no arguments and is called by GAUSS whenever an uncorrectable tape error occurs. The subroutine should be written by the user and could call EXIT or take any other action deemed appropriate.

SUBROUTINE RSTART (NRUN)

Since SUBROUTINE GAUSS may run for a considerable length of time, it should possess the capability to be interrupted and restarted. This can be achieved by writing a routine called RSTART. If sense switch 6 is down, RSTART is called periodically by GAUSS. On returning to GAUSS, all tapes are repositioned if NRUN is equal to zero. If NRUN is positive, tapes are not repositioned before resuming the reduction process.

SUBROUTINE MDIVID (N, NA, IMAX, A1, A2)

This subroutine performs the division necessary to compute a new column of multipliers. A1 and A2 are one-dimensional arrays, N elements in length, representing the real and imaginary parts respectively of one column of the matrix. MDIVID first interchanges element NA with element IMAX. Elements NA + 1 through N are then divided by element NA.

SUBROUTINE REDUCE (N, NA, IMAX, A1, A2, AM1, AM2)

This subroutine is used to perform the reduction of one column of the matrix with the NA^{th} column of multipliers. In the argument list above, N represents the order of the system, NA indicates which column of multipliers is being used, and IMAX indicates. which element is to be interchanged with element NA before processing. A1 and A2 are one-dimensional arrays representing the real and imaginary parts of the matrix column respectively, and AM1 and AM2 are one-dimensional arrays representing the real and imaginary parts respectively of the column of multipliers.

SUBROUTINE DETER (D1, D2, DET1, DET2, NB2)

As the system of equations is reduced, the determinant is computed by multiplying together the diagonal elements of the reduced system. This subroutine is used in performing this operation. DET1 and DET2 are the real and imaginary parts of the accumulated product and D1 and D2 are the real and imaginary parts of the next factor to be used. Because such an' extended product may exceed the range of floating point numbers the computer can handle, this subroutine carries the power of 2 separately as ND2 in order to prevent any overflow or underflow.

SUBROUTINE BSOLVE (K, RHS1, RHS2, COL1, COL2, ANS1, ANS2)

This subroutine is used during the backsolving operation. As with the reduction procedure, the backsolving is carried out by columns. K is an integer which indicates the particular element of the answer that is being obtained. RHS1 and RHS2 are one-dimensional arrays representing the real and imaginary parts respectively of the right-hand side. COL1 and COL2 are one-dimensional arrays representing the real and imaginary parts of column K of the reduced system. ANS1 and ANS2 are one-dimensional arrays representing the real and imaginary parts of the answer.

SUBROUTINE DPSET (NSYS, REMS, IMMS, RELS, IMLS)

SUBROUTINE DPRES (RECOL, IMCOL, ANS1, ANS2)

These two subroutines are the two entry points to the FAP coded subroutine used in the double precision calculation of the residues. As with the reduction and the backsolving, the residue calculation is performed by columns. The first entry point DPSET is used to indicate the size of the system, NSYS, and the location for the arrays for the most significant and least significant parts of the real and imaginary parts of the residue. The second entry DPRES is used during the calculation of the residues. RECOL and IMCOL are one-dimensional arrays containing the real and imaginary parts of one column of the matrix. ANS1 and ANS2 are the real and imaginary parts of the component of the answer associated with the column being processed.

CHAN (NT1, NT2, NOK)

This FAP coded subroutine is used to check that the channel requirements for the tapes are satisfied. NT1 and NW are two logical tape numbers. If these tapes are on different channels, NOK is set equal to 1. If they are on the same channel, NOK is set equal to zero.

BSET (NTAPE)

BSPACE

These two subroutines are two entry points to the FAP coded subroutine used to backspace logical tape NTAPE one physical record. BSET is used to set up the backspace instruction for logical tape NTAPE, Thereafter, each time BSPACE is called, tape NTAPE is backspaced one physical record.

RSETA (NTAPE, N, NRET, IQUIT)

READA (NREAD, A1, A2)

RCHKA

These three subroutines are the three entry points to one of the tape reading routines used by GAUSS. A call to RSETA initializes the routine to read records from tape NTAPE. N is the size of the system being solved. NRET is obtained from an ASSIGN statement and is used to construct a transfer instruction which is later executed if an uncorrectable tape ermr is encountered. IQUIT is an integer parameter which is used to indicate the nature of the trouble encountered if a return is made using NRET. A call to READA then initiates the tape reading. One physical record is read which should contain the last NREAD elements of each of the one-dimensional arrays A1 and A2, i.e., elements N-READ + 1 through N. A later call to RCHKA checks to see that the reading was completed satisfactorily. If an error has occurred, the tape is backspaced and the record is read again. Up to 5 attempts are made to read the tape correctly. If the routine is unsuccessful, IQUIT is set equal to 2 to indicate an uncorrectable reading error and a return is made using the NRET transfer instruction.

WSETA (NTAPE, N, NRET, IQUIT)

WRITEA (NWRITE, A1, A2)

WCHKA

These three subroutines are the three entry points to one of the tape writing routines used by GAUSS. The arguments above are analogous to those for RSETA, READA, and RCHKA and the execution of the routine is similar except for the following: Before writing a record, the end of tape indicator is interrogated. If it is on, IQUIT is set equal to 3 and a return is made using the NRET transfer instruction. If a tape redundancy check occurs, the tape is backspaced and blank tape is written before attempting to write the record again. Up to 10 attempts are made to write the record correctly. If the routine is unsuccessful, IQUIT is set equal to 1 and a return is made using the NRET transfer instruction.

RSETB (NTAPE, N, NRET, IQUIT)

READB (NREAD, A1, A2)

RCHKB

WSETB (NTAPE, N, NRET, IQUIT)

WRITEB (NWRITE, A1, A2)

WCHKB

These subroutines are identical to the above routines ending in A-except for the name.

RSETC (NTAPE, NRET, IQUIT)

READC (NREAD, A1, A2)

RCHKC

WSETC (NTAPE, NRET, IQUIT)

WRITEC (NWRITE, A1, A2)

WCHKC

These subroutines are almost identical to the routines ending in A, the difference being that these mu-tines read or write the first NREAD or NWRITE elements respectively of the arrays Al and A2.

```
      SUBROUTINE GAUSS(NSYS,ISOLVD,KITER,EPS,ANS1,ANS2,RHS1,RHS2,
     1          KCOEF,KCOPY,KMULT,KT1,KT2,KT3,ISING)
C        THIS SUBROUTINE SOLVES A SYSTEM OF UP TO 500 SIMULTANEOUS
C        ALGEBRAIC EQUATIONS WITH COMPLEX COEFICIENTS USING GAUSS
C        REDUCTION.   THE MATRIX FLEMENTS ARE STORED ON TAPE KCOEF
C        BY COLUMNS .
         DIMENSION A1(500),A2(500),B1(500),B2(500),
     1             C1(500),C2(500),D1(500),D2(500),
     2             AM1(500),AM2(500),BM1(500),BM2(500),
     3             RHS1(500),RHS2(500),ANS1(500),ANS2(500),IORDER(500)
            N = NSYS
            NCOEF = KCOEF
            NCOPY = KCOPY
            NMULT = KMULT
            NT1=KT1
            NT2=KT2
            NT3=KT3
            IF  (500 - N)  4,8,8
    4             WRITE OUTPUT TAPE 6, 6, N
   60             FORMAT(1H0,10X,29HARRAY SIZE EXCEEDED IN GAUSS./
     1                 1H0,10X,24HARRAY SIZE=500      N =I4/
     2                 1H0,10X,21HEXECUTION TERMINATFD.)
                  PRINT 7,  NCOEF
   70             FORMAT(1H0,25HPLEASE SAVE LOGICAL TAPE I2,
     1                 42H.  THIS PROGRAM WILL PAUSE WHEN COMPLETED.)
                  REWIND NCOES
                  PAUSE
                  CALL EXIT
    8       NITER = KITER
            NTERR=0
            REWIND NCOPY
            REWIND YMULT
            REWIND NT1
            IF(ISOLVD - 1)  50,10,50
   10             ITER = 0
                  DO  20  K  =1,N,1
                       C1(K) = RHS1(K)
                       C2(K) = RHS2(K)
                       ANS1(K)=0.0
   20                  ANS2(K)=0.0
                  EHOLD1=1.0
                  GO TO 3400
   50       ISING =  0
            EPSA= I.OE-15
            EPSB = 1.0E+15
            NSAVE=NT1
C        CHECK COMPATABILITY OF TAPE ASSIGNMENTS.
            IQUIT = 0
            CALL CHAN(NT1,NMULT,NOK) .
            IF(NOK)  80,80,90
   80             WRITE OUTPUT TAPE 6,160,  NT1,NMULT
                  IQUIT =  1
   90       CALL CHAN(NT1,NT2,NOK)
            IF(NOK)  100,100,110
  100             WRITE OUTPUT TAPE 6, 165,  NT1,NT2
                  IQUIT =  1
  110       CALL  CHAN(NT1,NT3,NOK)
            IF  (NOK)  120,120,130
  120             WRITE OUTPUT TAPE 6, 170,  NT1,NT3
                  IQUIT =  1
  130       CALL CHAN(NCOEF,NCOPY,NOK)
            IF(NOK)  140,140,150
  140             WRITE OUTPUT TAPE 6, 175,  NCOEF,NCOPY
```

```
                 IQUIT = 1
150          IF (IQUIT)  100)0,20(,155
155              ISING = 4
                 WRITE OUTPUT TAPE 6, 180
                 RETURN
1600FORMAT (1H0,25X,14HLOGICAL TAPES I2,5H AND I2,57H HAVE BEEN ASSIGN
   1ED T   O KT1 AND KMULT RFSPECTFULLY.   THFSE/1H,35X,43HLOGICAL TAPF U
   2NITS ARE ON THE SAME CHANNEL.)
1650FORMAT(1H0,25X,14HLOGICAL TAPES I2,5H AND I2,55H HAVE BEEN ASSIGN
   1ED TO KT1 AND KT2 RESPECTFULLY..  THESE/1H ,35X,43HLOGICAL TAPE UNI
   2TS ARE ON THE SAME CHANNEL.)
1700FORMAT(1H0,25X,14HLOGICAL TAPES I2,5H AND I2,55H HAVE BEEN ASSIGN
   1ED TO KT1 AND KT3 RESPECTFULLY.  THESE/1H,35X,43HLOGICAL TAPE UNI
   2TS ARE ON THE SAME CHANNEL.)
1750FORMAT(1H0,25X,14HLOGICAL TAPES I2,5H AND I2,59H HAVE BEEN ASSIGN
   1ED TO KCOEF AND KCOPY RFSPECTFULLY.  THFSE/1H ,?5X,43HLOGICAL TAPE
   2 UNITS ARF ON THE SAME CHANNFL.)
1800FORMAT(1H0,25X,79HTHIS PROGRAM REQUIRES THAT TAPE KT1 3E ON A CHA
   1NNEL DIFFERENT FROM THAT USED BY/1H,35X,68HKMULT, YT2, OR KT3. A
   2LSO TAPES KCOEF AND KCOPY MUST BE ON DIFFERENT/1H,35X,9HCHANNELS.
   3)
200          REWIND NCOEF
             REWIND NT2
             REWIND NT3
C        THE FOLLOWING STATEMENTS ARE NECESSARY TO MAKE THE
C        COMPILER HAPPY.
             IQUIT = 4
220          ASSIGN 6000 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
725          ASSIGN 6200 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
330          ASSIGN 6400 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
235          ASSIGN 6600 TO NRET
             GO TO NRET, (600),6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8?00,8400,8600,8800)
240          ASSIGN 6800 TO NRET
             GO T O NRET, (60)0,62(( ,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
245          ASSIGN 7000 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
?50          ASSIGN 7200 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
255          ASSIGN 7400 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
260          ASSIGN 7600 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
265          ASSIGN 7800 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
270          ASSIGN 3300 TO NRET
             GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
275          ASSIGN 8200 TO NRET
             GO TO NRFT, (6000,6200,6400,6600,6800,7000,7200,7400,
   1                7600,7800,8000,8200,8400,8600,8800)
```

```
  280         ASSIGN 8400 TO NRET
              GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
     1              7600,7800,8000,8200,8400,8600,8800)
  785         ASSIGN 8600 TO NRET
              GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
     1              7600,7800,8000,8200,8400,8600,8800)
  290         ASSIGN 8800 TO NRET
              GO TO NRET, (6000,6200,6400,6600,6800,7000,7200,7400,
     1              7600,7800,8000,8200,8400,8600,8800)
  295         CONTINUE
C        'THE COMPUTATION AND RZDUCTION CAN NOW BEGIN.
              NA = 1
              NB = 2
              NC = 3
              ND = 4
              DFT1 = 1.0
              DET2 = 0.0
              NB2 = 0
              DMAX = 0.0
              DMIN = 1.C
              WRITE TAPE NT3, (RHS1(K),RHS2(K), K = 1,N,1)

C        READ IN AND COPY ON TAPE NCOPY FOUR COLUMNS.   IF FEWER
C        THAN FOUR COLUMNS REMAIN, READ IN AND COPY THE REMAINING
C        COLUMNS.
  400         ASSIGN 6000 TO NRET
              CALL WSETA(NCOPY,N,NRET,IQUIT)
              NWRITE = N
              IF (NA - N)  420,420,10000
  420              READ TAPE NCOEF,(A1(K),A2(K),K=1,N,1)
                   NTEST = 1
                   CALL WRITEA(NWRITE,A1,A2)
              IF   (NB - N)  440,440,500
  440              READ TAPE NCOEF,(B1(K),B2(K),K=1,N,1)
                   CALL WCHKA
                   NTEST = 2
                   CALL WRITEA(NWRITE,B1,B2)
              IF  (NC   -N)  460,460,500
  460              READ TAPE NCOFF,(C1(K),C2(K),K=1,N,1)
                   CALL WCHKA
                   NTEST = 3
                   CALL WRITEA(NWRITE,C1,C2)
              IF  (ND   -N)  480,480,500
  480              READ TAPE NCOEF,(D1(K),D2(K),K=1,N,1)
                   CALL WCHKA
                   NTEST = 4
                   CALL WRITEA(NWRITE,D1,D2)
  500         CALL WCHKA
              IF   (NA - 1)  10000,2000,600
C        BEGIN REDUCTION OF COLUMNS USING MULTIPLIERS STORED
C        ON TAPE.
  600         ASSIGN 6200 TO NRET
              CALL RSETA(NMULT,N,NRET,IQUIT)
              CALL WSETA(NMULT,N,NRET,IQUIT)
              ASSIGN 6400 TO NRET
              CALL RSETB(NT1,N,NRET,IQUIT)
              CALL WSETB(NT2,N,NRET,IQUIT)
              NREAD = N - 1
              MI = 4*((NA - 4)/8) + 4
              M2 = 4*(NA/8) + 5
              MLAST = NA - 1
              CALL READA(NREAD,AM1,AM2)
              CALL RCHKA
```

```
            NREAD = NREAD - 1
            CALL READA(NREAD,BM1,BM2)
            IMULT = 1
C     REDUCTION USING MULTIPLIERS IN AM1 AND AM2.
  700       IMAX = IORDER(IMULT)
            CALL REDUCE(N,IMULT,IMAX,A1,A2,AM1,AM2)
            IF (NB - N) 720,720,800
  720       CALL REDUCE(N,IMULT,IMAX,B1,B2,AM1,AM2)
            IF (NC - N) 740,740,800
  740       CALL REDUCE(N,IMULT,IMAX,C1,C2,AM1,AM2)
            IF (ND - N) 760,760,800
  760       CALL REDUCE(N,IMULT,IMAX,D1,D2,AM1,AM2)
C     INITIATE TAPE READING AND WRITING OF MULTIPLIER AND
C     REDUCTION TAPES
  800       IMULT = IYULT + 1
            NWRITE = NREAD
            NREAD = NREAD - 1
            IF (IMULT - M1) 820,840,880
  820          CALL RCHKA
               CALL READA(NREAD,AM1,AM2)
               GO TO 1200
  840          IF (IMULT - MLASTI 845,860,10000
  845             CALL RCHKA
                  CALL READB(NREAD,AM1,AM2)
                  GO TO 1200
  860             CALL RCHKA
                  IF (ND - N) 870,1200,1200
  870                REWIND NYULT
                     GO TO 1200
  880       IF (IMULT - MLAST) 885,980,1040
  885          CALL RCHKB
               IF (IMULT - M2) 900,940,970
  900             IF (IMULT - MI - 1) 10000,905,920
  905                CALL WRITEA(NWRITE,BM1,BM2)
                     CALL READB(NREAD,AM1,AM2)
                     GO TO 1200
  920                CALL WCHKA
                     CALL WRITEA(NWRITE,BM1,BM2)
                     CALL READB(NREAD,AM1,AM2)
                     GO TO 1200
  940             IF (M1 + 1 - M2) 945,950,10000
  945                CALL WCHKA
  950                IF (ND - N) 955,960,960
  955                   REWIND NMULT
  960                CALL WRITEB(NWRITE,BM1,BM2)
                     CALL READB(NREAD,AM1,AM2)
                     GO TO 1200
  970             CALL WCHKB
                  CALL WRITEB(NWRITE,BM1,BM2)
                  CALL READB(NREAD,AM1,AM2)
                  GO TO 1200
  980          CALL RCHKB
               REWIND NT1
               IF(MLAST - 8) 10000,1000,1020
 1000             CALL WCHKA
                  CALL WRITEA(NWRITE,BM1,BM2)
                  GO TO 1200
 1020             CALL WCHKB
                  CALL WRITEB(NWRITE,BM1,BM2)
                  GO TO 1200
 1040          IF (MLAST-4) 10000,2000,1050
 1050          IF (MLAST-8) 10000,1060,1100
 1060             CALL WCHKA
```

```
                  IF   (ND - N)   1080,2000,2000
 1080                   REWIND NMULT
                        GO TO 2000
 1100             CALL WCHKB
               GO TO 2000
C        REDUCTION USING MULTIPLIERS- IN BM1 AND BM2.
 1200        IMAX = IORDER(IMULT)
             CALL REDUCE(N,IMULT,IMAX,A1,A2,BM1,BM2)
             IF  (NB - N)   1220,1220,1400
 1720        CALL REDUCE(N,IMULT,IMAX,B1,B2,BM1,BM2)
             IF  (NC  - N)  1240,1240,1400
 1240        CALL REDUCE(N,IMULT,IMAX,C1,C2,BM1,BM2)
             IF   (ND - N)  1260,1260,1400
 1260        CALL REDUCE(N,IMULT,IMAX,D1,D2,BM1,BM2)
C        INITIATE TAPE READING AND WRITING OF MULTIPLIER ANC
C        REDUCTION TAPES.
 1400        IMULT = IMULT + 1
             NWRITE = NREAD
             NREAD  =  NREAD  - 1
             IF (IMULT-M1)   1420,1440,1480
 1420             CALL RCHKA
                  CALL READA(NREAD,BM1,BM2)
                  GO TO 700
 1444             IF (IMULT-MLAST)   1445,1460,10000
 1445                  CALL RCHKA
                       CALL READB(NREAD,BM1,BM2)
                       GO TO 700
 1460                  CA! L RCHKA
                       IF (ND -  NJ)   1470,700,700
 1470                        REWIND NMULT
                             GO TO 700
 1480        IF (IMULT-MLAST)   1485,1580,1640
 1485             CALL RCHKB
                  IF (IMULT-M2)   1500,1540,1570
 1500                  IF   (IMJLT -  MI  - 1)   10000,1505,1520
 1505                        CALL WRITEA(NWRITE,AM1,AM2)
                             CALL READB(NREAD,BM1,BM2)
                             GO TO 700
 1520                        CALL WCHKA
                             CALL WRITEA(NWRITE,AM1,AM2)
                            'CALL READB(NREAD,BM1,BM2)
                             GO TO 700
 1540                  IF   (M1 + 1 - M2)   1545,1550,10000
 1545                        CALL WCHKA
 1550                        IF   (ND - N)   1555,1560,1560
 1555                             REWIND NMULT
 1560                        CALL WRITEB(NWFITE,AM1,AM2)
                             CALL READB(NREAD,BM1,BM2)
                             GO TC 700
 1570                  CALL WCHKB
                       CALL WRITEB(NWRITE,AM1,AM2)
                       CALL READB(NREAD,BM1,BM2)
                       GO TO 700
 1580             CALL RCHKB
                  REWIND NT1
                  I  F(MLAST - 8)   10000,1600,1620
 1600                  CALL WCHKA
                       CALL WRITEA(NWRITE,AM1,AM2)
                       GO TO 700
 1620                  CALL WCHKB
                       CALL WRITEB(NWRITE,AM1,AM2)
                       GO TO 700
 1640             IF (MLAST-4)   100(0,2000,1650
```

```
1650          IF(MLAST - 8)  10000,1660,1700
1660             CALL WCHKA
                 IF  (ND - N)  1680,2000,2000
1680                REWIND NMULT
                    GO TO 2000
1700             CALL WCHKB
C        REDUCTION OF COLUMNS AFTER PROCESSING WITH MULTIPLIERS.
2000       IF  (ND - N)  2010,2005,2005
2005          REWIND NT2
2010       ASSIGN 7000 TO NRET
           CALL WSETC(NT3,NRET,ICUIT)
           PMAX=0.0
           IMAX = NA
           DO 2040 K = NA,N,1
              COMP = ABSF(A1(K)) + ABSF(A2(K))
              IF(PMAX - COMP)  2020,2040,2040
2020             PMAX = COMP
                 IMAX = K
2040          CONTINUE
           IORDER(NA) = IMAX
           NTEST = NA
           IF(EPSA - PMAX)  2060,2060,4000
2060       IF(PMAX - EPSB)  2070,2070,4000
2070       CALL DETER(A1(IMAX),A2(IMAX),DET1,DET2,NB2)
           DMAX = MAX1F(DMAX,PMAX)
           DMIN = MIN1F(DMIN,PMAX)
           NWRITE=NA
           IF  (NA - N)  2080,2400,10000
2080       CALL MDIVID(N,NA,IMAX,A1,A2)
           CALL WRITEC(NWRITE,A1,A2)
           CALL REDUCE(N,NA,IMAX,RHS1,RHS2,A1,A2)
           CALL REDUCE(N,NA,IMAX,B1,B2,A1,A2)
           CALL WCHKC
           PMAX = 0.0
           IMAX = NB
           DO 2140 K = NB,N,1
              COMP = ABSF(B1(K)) + ABSF(B2(K))
              IF(PMAX - COMP)  2120,2140,2140
2120             PMAX = COMP
                 IMAX = K
2140          CONTINUE
           IORDER(NB) = IMAX
           NTEST=NB
           IF(EPSA - PMAX)  2160,2160,4000
2160       IF(PMAX - EPSB)  2170,2170,4000
2170       CALL DETER(B1(IMAX),B2(IMAX),DET1,DET2,NB2)
           DMAX = MAX1F(DMAX,PMAX)
           DMIN = MIN1F(DMIN,PMAX)
           NWRITE=NB
           IF  (NB - N)  2180,2420,10000
2180       CALL MDIVID(N,NB,IMAX,B1,B2)
           CALL WRITEC(NWRITE,B1,B2)
           CALL REDUCE(N,NB,IMAX,RHS1,RHS2,B1,B2)
           CALL REDUCE(N,NA,IORDER(NA),C1,C2,A1,A2)
           CALL REDUCE(N,NB,IORDER(NB),C1,C2,B1,B2)
           CALL WCHKC
           PMAX = 0.0
           IMAX = NC
           DO 2240 K = NC,N,1
              COMP = ABSF(C1(K)) + ABSF(C2(K))
              IF(PMAX - COMP)  2220,2240,2240
2220             PMAX = COMP
                 IMAX = K
```

```
 2240          CONTINUE
          IORDER(NC) = IMAX
          NTEST = NC
          IF (EPSA - PMAX)  2260,2260,4000
 2260     IF(PMAX - EPSB)  2270,2270,4000
 2270     CALL DETER(C1(IMAX),C2(IMAX),DET1,DET2,NB2)
          DMAX = MAX1F(DMAX,PMAX)
          DMIN = MIN1F(DMIN,PMAX)
          NWRITE = NC
          IF  (NC  - N)  2280,2440,10000
 2280     CALL MDIVID(N,NC,IMAX,C1,C2)
          CALL WRITEC(NWRITE,C1,C2)
          CALL REDUCE(N,NC,IMAX,RHS1,RHS2,C1,C2)
          CALL REDUCE(N,NA,IORDER(NA)  ,D1,D2,A1,A2)
          CALL REDUCE(N,NB,IORDER(NB),D1,D2,B1,B2)
          CALL REDUCE(N,NC,IORDER(NC),D1,D2,C1,C2)
          CALL WCHKC
          PMAX = 0.0
          IMAX = ND
          DO 2340 K = ND,N,1
               COMP = ABSF(D1(K)) + ABSF(D2(K))
               IF(PMAX - COMP)  2320,2340,2340
 2320               PMAX = COMP
                    IMAX = K
 2340          CONTINUE
          IORDER(ND) = IMAX
          NTEST = ND
          IF(EPSA - PMAX)  2360,2360,4000
 2360     IF(PMAX - EPSB)  2370,2370,4000
 2370     CALL DETER(D1(IMAX),D2(IMAX),DET1,DET2,NB2)
          DMAX = MAX1F(DMAX,PMAX)
          DMIN = MIN1F(DMIN,PMAX)
          NWRITE = ND
          IF  (ND - N)  2380,2460,10000
 2380     CALL MDIVID(N,ND,IMAX,D1,D2)
          CALL WRITEC(NWRITE,D1,D2)
          CALL REDUCE(N,ND,IMAX,RHS1,RHS2,D1,D2)
          CALL WCHKC
          GO TO 2500
 2400     CALL WRITEC(NWRITE,A1,A2)
          CALL WCHKC
          GO TO 2700
 2420     CALL WRITEC(NWRITE,B1,B2)
          CALL WCHKC
          GO TO 2700
 2440     CALL WRITEC(NWRITE,C1,C2)
          CALL WCHKC
          GO TO 2700
 2460     CALL WRITEC(NWRITE,D1,D2)
          CALL WCHKC
          GO TO 2700
C      WRITE NEW MULTIPLIERS ON TAPE.
 2500     IF  (NA - 1)  10000,2505,2520
 2505          ASSIGN 6600 TO NRET
               CALL WSETA(NMULT,N,NRET,IQUIT)
               GO TO 2540
 2520          ASSIGN 6800 TO NRET
               CALL WSETA(NT2,N,NRET,IQUIT)
 2540     NWRITE = N - NA
          CALL WRITEA(NWRITE,A1,A2)
          CALL WCHKA
          NWRITE = N - NB
          CALL WRITEA(NWRITE,B1,B2)
```

```
      CALL WCHKA
      NWRITE = N - NC
      CALL WRITEA(NWRITE,C1,C2)
      CALL WCHKA
      NWRITE = N-ND
      CALL WRITEA(NWRITE,D1,D2)
      CALL WCHKA
C     CALL RSTART IF RESTART IS DESIRED.
 2600 IF (SENSE SWITCH 6) 2610,2670
 2610    CALL RSTART(NRUN)
      IF(NRUN) 10000,2620,2670
 2620    REWIND NCOEF
         RE JIND NCOPY
         REWIND NMULT
         ASSIGN' 8600 TO NRET
         CALL RSETA(NCOPY,N,NRET,IQUIT)
         NREAD = N
         DO 2640 J = 1,ND,1
            CALL READA(NREAD,A1,A2)
            READ TAPE NCOEF,(B1(K),B2(K),K=1,N,1)
 2640       CALL RCHKA
         READ TAPE NT3, (A1(K),A2(K),K =1,N,1)

         ASSIGN 8800 TO NRET
         CALL RSFTC(NT3,NRET,IQUIT)
         NREAD=0
         DO 2660 J=1,ND,1
            NREAD = NREAD + 1
            CALL RFADC(NREAD,A1,A2)
 2660       CALL RCHYC
 2670 REWIND NT1
      REWIND NT2
      REWIND NMULT
      NTEMP = NT1
      NT1 = NT2
      NT2 = NTEMP
      NA = NA + 4
      NB = NA + 1
      NC = NB + 1
      ND = NC + 1
      GO TO 400
C     THE REDUCTION IS COMPLETE.  PRINT THE VALUE OF THE DETERMINANT
C     AND THE MAXIMUM AND MINIYUM PIVOTAL ELEMENTS.
 2700 TEMP =1.0
      DO 2720 K = 1,N,1
         IF(IORDER(K) - K) 2715,2720,2715
 2715       TEMP =-TEMP
 2720    CONTINUE
      FNB2 = FLOATF(NB2)
      KE1 = FNB2/3.3219281
      KE2 = KE1
      EXPON= MODF(FNB2,3.3219281)
      AMPL = TEMP*(2.0**EXPON)
      DET1 = AMPL*DET1
      DET2 = AMPL*DET2
      IF(DET1) 2745,2740,2745
 2740    KE1 = 0
         GO TO 2755
 2745 IF(ABSF(DET1)-1.01 27469275592750
 2746    DET1 = 10.0*DET1
         KE1 = KE1- 1
         GO TO 2745
 2750 IF(10.0.- ABSF(DET1)) 2751,2751,2755
```

```
2751            DET1 = DET1/10.0
                KE1 = KE1 + 1
                GO TO 2750
2755        IF(DET2)  2760,2756,2760
2756            KE2 = 0
                GO TO 2770
2760        IF(ABSF(DET2)-1.0)  2761,2770,2765
2761            DET2 = 10.0*DET2
                KE2 = KE2 - 1
                GO TO 2760
2765        IF(10.0- ABSF(DET2))  2766,2770,2770
2766            DET2 = DET2/10.0
                KE2 = KE2 + 1
                CO TO 2765
2770        WRITE OUTPUT TAPE 6, 2775,  DET1,KE1,DET2,KE2,DMAX,DMIN
27750FORMAT(1H0,25X,70HTHE GAUSSIAN REDUCTION IS COMPLETED.   THE DETER
    1MINANT OF THE MATRIX IS/1H,45X,18HREAL PART         = F9.5,1HE,I4/
    21H,45X,18HIMAGINARY PART    =F9.5,1HE,14/1H0,25X,62HTHE MAGNITUDE
    30F THE LARGEST PIVOTAL ELEMENT IS APPROXIMATELY 1PE9.2,1H./1H0,
    425X,63HTHE MAGNITUDE OF THE SMALLEST PIVOTAL ELEMENT IS APPROXIMAT
    5ELY 1PE9.2 , 1H. )
C         COPY ALL MULTIPLIERS ON TO THE MULTIPLIER TAPE.
2800        ASSIGN 7200 TO NRET
            CALL WSETA(NMULT,N,NRET,IQUIT)
            IF (N-   121  2860,2860,2820
2820        ASSIGN 7400 TO NRET
            CALL RSETB(NT2,N,NRET,IQUIT)
            IMULT = M2
2840        NWDS= N - IMULT
            CALL READB(NWDS,AM1,AM2)
            CALL RCHKB
            CALL WRITEA(NWDS,AM1,AM2)
            CALL WCHKA
            IMULT = IMULT + 1
            IF(IMULT - NA)  2840,2860,2860
2860        NWRITE = N - NA
            IF(NWRITE)  10000,2940,2880
2880        CALL WRITEA(NWRITE,A1,A2)
            CALL WCHKA
            NWRITE = N -  NB
            IF (NWRITE)  10000,2940,2900
2900        CALL WRITEA(NWRITE,B1,B2)
            CALL WCHKA
            NWRITE=N-NC
            IF(NWRITE)  10000,2940,2920
2920        CALL WRITEA(NWRITE,C1,C2)
            CALL WCHKA
2940        REWIND NT1
            REWIND NT2
            REWIND NMULT
            REWIND NCOEF
            REWIND NCOPY
            NT1= NSAVE
C         BACKSOLVE THE REDUCED SYSTEM OF EQUATIONS.   ALSO WRITE THE
C         REDUCED COLUMNS ON TAPE NT1 IN THE REQUIRED ORDER.
3000        CALL BSET(NT3)
            ASSIGN 7600 TO NRET
            CALL RSETC(NT3,NRET,IQUIT)
            ASSIGN 7800 TO NRET
            CALL WSETC(NT1,NRET,IQUIT)
            NREAD = N
            CALL BSPACE
            CALL READC(NREAD,AM1,AM2)
```

```
             CALL RCHKC
             CALL BSPACE
 3040        NWDS = NREAD
             NREAD=NREAD-1
             CALL WRITEC(NWDS,AM1,AM2)
             IF(NREAD)  10000,3080,3(60
 3060             CALL BSPACE
                  CALL READC(NREAD,BM1,BM2)
 3080        CALL BSOLVE(NWDS,RHS1,RHS2,AM1,AM2,ANS1,ANS2)
             CALL WCHKC
             IF(NREAD)  10000,3180,3100
 3100        CALL RCHKC
             CALL BSPACE
             NWDS = NREAD
             NREAD=NREAD-1
             CALL WRITEC(NWDS,BM1,BM2)
             IF (NREAD)  10000,3140,3120
 3120             CALL BSPACE
                  CALL READC(NREAD,AM1,AM2)
 3140        CALL BSOLVE(NWDS,RHS1,RHS2,BM1,BM2,ANS1,ANS2)
             CALL WCHKC
             IF (NREAD)  10000~3180~3160
 3160        CALL RCHKC
             CALL BSPACE
             GO TO 3040
 3180        REWIND NT1
             REWIND NT3
             ITER  = 0
             EHOLD1=1.0
             READ TAPE NT3,(C1(K),C2(K),K= 1,N,1)
             REWIND NT3
             IF(NITER)  3190,3190,3200
 3190             RETURN
C         COMPUTE RFSIDUES USING  DOUBLE PRECISION.
 3300        ITER = ITER + 1
             ASSIGN 8000 TO NRET
             CALL RSETA(NCOPY,N,NRET,IQUIT)
             CALL DPSET(N,RHS1,RHS2,D1,D2)
             DO  3220  K=1,N,1
                  RHS1(K) = C1(K)
                  RHS2(K) = C2(K)
                  D1(K) = 0.0
 3220             D2(K) = 0.0
             NREAD= N
             CALL READA(NREAD,AM1,AM2)
             ICOL = 1
 3240        'IF (ICOL - N)  3260,3250,3320
 3250             CALL RCHKA
                  GO TO 3370
'3260        CALL RCHKA
             CALL READA(NREAD,BM1,BM2)
 3270        CALL DPRES(AM1,AM2,ANS1(ICOL),ANS2(ICOL))
             ICOL = ICOL + 1
             IF (ICOL - N)  3290,3280,3320
 3280             CALL RCHKA
                  GO TO 3300
 3290        CALL RCHKA
             CALL READA(NREAD,AM1,AM2)
 3300        CALL DPRES(BM1,BM2,ANS1(ICOL),ANS2(ICOL))
             ICOL = ICOL + 1
             GO TO 3240
 3320        REWIND NCOPY
C         REDUCE  THE  NEW RIGHT HAND SIDE.
```

```
 3400          ASSIGN  8200 TO NRET
               CALL RSETA(NMULT,N,NRET,IQUIT)
               NREAD = N - 1
               CALL READA(NREAD,AM1,AM2)
 3420          IMULT = N - NREAD
               NREAC = NREAD - 1
               IF (NREAD)  3520,3440,3450
 3440               CALL RCHKA
                    GO TO 3460
 3450          CALL RCHKA
               CALL READA(NREAD,BM1,BM2)
 3460          CALL REDUCE(N,IMULT,IORDER(IMULT),RHS1,RHS2,AM1,AM2)
               IMULT = N - NREAD
               NREAD = NREAD - 1
               IF(NREAD)  3520,3480,3490
 3480               CALL RCHKA
                    GO TO 3500
 3490          CALL RCHKA
               CALL READA(NREAD,AM1,AM2)
 3500          CALL REDUCE(N,IMULT,IORDER(IMULT),RHS1,RHS2,BM1,BM2)
               GO TO 3420
 3520          REWIND NMULT
C         BACK-SOLVE AND CORRECT SOLUTION.
 3600          ASSIGN  8400 TO NRET
               CALL RSETC(NT1,NRET,IQUIT)
               NREPD = N
               CALL READC(NREAD,AM1,AM2)
 3620          NWDS = NREAD
               NREAD = NREAD - 1
               IF(NREAD)  3720,3640,3650
 3640               CALL RCHKC
                    GO TO 3660
 3650          CALL RCHKC
               CALL READC(NREAD,BM1,BM2)
 3660          CALL BSOLVE(NWDS,RHS1,RHS2,AM1,AM2,A1,A2)
               NWDS = NREAD
               NREAD = NREAD - 1
               IF (NREAD)  3720,3680,3690
 3680               CALL RCHKC
                    GO TO 3700
 3690          CALL RCHKC
               CALL.READC(NREAD,AM1,AM2)
 3700          CALL BSOLVE(NWDS,RHS1,RHS2,BM1,BM2,A1,A2)
               GO TO 3620
 3720          REWIND NT1
               DO  3740  K = 1,N,1
                    B1(K) = ANS1(K)
                    B2(K) = ANS2(K)
                    ANS1(K) = ANS1(K) + A1(K)
 3740               ANS2(K) = ANS2(K) + A2(K)
               IF (NITER)  3760,3760,3780
 3760               RETURN
 3780          IF(ITER)  3760,3200,3800
 3800          RMAX = 0.0
               EMAX = 0.0
               DO  3840  K= 1,N,1
                    RMAX = MAX1F(RMAX,ABSF(ANS1(K)),ABSF(ANS2(K)))
 38400              EMAX = MAX1F(EMAX,ABSF(ANS1(K) - B1(K)),
     1                             ABSF(ANS2(K) - B2(K)))
               RERR = EMAX/RMAX
               IF  (RERR-EPS)  3860,3860,3880
 3860               WRITE OUTPUT TAPE 6, 3940,  ITER,RERR,EPS
                    RETURN
```

```
 3880        IF(EHOLD1 - RERR)   3890,3890,3900
 3890           WRITE OUTPUT TAPE 6, 3945,   ITER,RERR,EHOLD1,EPS
                RETURN
 3900        IF(NITER - ITER)   3910,3910,3920
 3910           WRITE OUTPUT TAPE 6, 3950,   ITER,RERR,EPS,EHOLD1
                RETURN
 3920        EHOLD1 = RERR
             GO TO 3200
 39400FORMAT (1H0,25X,48HTHE  ACCURACY  DESIRED HAS BEEN ESTABLISHED AFTER
     1  I2,26H ITERATIONS.   THE RELATIVE/1H,35X,9HERROR   IS   1PE9.2,
     234H.   THE RELATIVE ERROR DESIRED WAS 1PE9.2,1H.)
 39450FORMAT (1H0,25X,44HTHE ITERATIVE PROCEDURE IS NOT CONVERGING. I2,
     131H ITERATIONS HAVE BEEN COMPLETED/1H  ,35X,38HBUT THE RELATIVE ERR
     2OR INCREASED FROM 1PE9.2,4H TO 1PE9.2,7H  DURING/1H,35X,52HTHE LAS
     3T  ITERATION.   THE RELATIVE ERROR DESIRED WAS 1PE9.2,1H.)
 39500FORMAT (1H0,25X,4HTHE I2,65H ITERATIONS ALLOWED HAVE BEEN COMPLETE
     1D BUT THE RELATIVE ERROR IS/1H  ,35X,6HSTILL 1PE9.2,24H WHILE THAT
     2DESIRED WAS 1PE9.2,15H.   THE RELATIVE/1H,35X,37HERROR FOR THE PRE
     3VIOUS ITERATION WAS 1PE9.2,1H.)
 4000        WRITE OUTPUT TAPE 6, 4040,   NTEST,PMAX,EPS3,EPSB
             ISING = 1
             IF(EPS3 - PMAX)   4020,4020,4030
 4020           ISING = 2
 4030        RETURN
 40400FORMAT(1H1,25X,68HTHE BOUNDS FOR PIVOTAL ELEMENTS WERE EXCEEDED I
     1N DETERMINING PIVOTAL/1H,35X,8HELEMENT I3,43H.   THE MAGNITUDE OF
     2THIS PIVOTAL ELEMENT IS/1H,35X,14HAPPROXIMATELY  1PE9.2,33H.   THIS
     3 PROGRAM REQUIRES THAT ALL/1H,35X,29HPIVOTAL ELEMENTS LIE BETWEEN
     4 1PE9.2,5H AND 1PE9.2,1H.)
C        PROCEDURES FOR CORRECTING TAPE ERRORS.   IQUIT ASSUMES THE
C        VALUES 1,2, OR 3 ACCORDING TO THE DIFFICULTY ENCOUNTERED.
C                  IQUIT=  1        UNCORRECTABLE WRITING ERROR
C                  IQUIT =  2       UNCORRECTABLE READING ERROR
C                  IQUIT  =  3      SHORT TAPE
C        WRITING ERROR ON TAPE NCOPY DURING COPYING OF INPUT COLUMNS.
 6000        NGO= IQUIT
             MTERR = NCOPY
             GO TO(9500,10000,9700,225),NGO
C        READING OR WRITING ERROR ON TAPE NMULT DURING REDUCTION.
 6700        NGO = IQUIT
             MTERP  = NMULT
             GO TO(9500,9600,9700,230),NGO
C        READING OR WRITING ERROR ON TAPE NT1   OR NT2 DURING REDUCTION.
 6400        NGO = IQUIT
             MTERR = NT1
             NTERR = NT2
             GO TO(9500,9600,9700,235),NGO
C        WRITING ERROR ON TAPE NMULT DURING MULTIPLIER WRITING.
 6600        NGO = IQUIT
             YTERR = NMULT
             GO TO(9500,10000,9700,240),NGO
C        WRITING ERROR ON TAPE NT2 DURING MULTIPLIER WRITING.
 6800        NGO = IQUIT
             MTERR = NT2
             GO TO(9500,10000,9700,245),NGO
C        WRITING ERROR ON TAPE NT3 DURING REDUCEC COLUMN WRITING.
 7000        NGO = IQUIT
             MTERR = NT3
             GO TO(9500,10000,9700,250),NGO
C        WRITING ERROR ON TAPE NMULT DURING MULTIPLIER COPYING.
 7200        NGO = IQUIT
             MTERR = NMULT
             GO TO(9500,10000,9700,255),NGO
```

```
C          READING ERROR ON TAPE NT2 DURING MULTIPLIER COPYING.
 7400            NGO = IQUIT
                 MTERR = NT2
                 GO TO(10000,9600,10000,260),NGO
C          READING ERROR ON TAPE NT3 DURING INITIAL BACKSOLVING.
 7600            NGO = IQUIT
                 MTERR= NT3
                 GO TO(10000,9600,10000,265),NGO
C          WRITING ERROR ON TAPE NT1 DURING INITIAL BACKSOLVING.
 7800            NGO=IQUIT
                 MTERR =NT1
                 GO TO(9500,10000,9700,270),NGO
C          READING FRROR ON TAPE NCOPY DURING ITERATION.
 8000            NGO = IQUIT
                 MTERR = NCOPY
                 GO TO(10000,9600,10000,275),NGO
C          READING ERROR ON TAP" NMUL' DURING RHS REDUCTION.
 8300            NGO = IQUIT
                 MTERR = NMULT
                 GO TO(10000,9600,10000,280),NGO
C          READING ERROR ON TAPE NT1 DURING BACKSOLVING.
 8400            NGC=IQUIT
                 MTERR = NT1
                 GO TO(10000,9600,10000,285),NGO
C          READING FRROR ON TAPE NCOPY DURING RESTARTING.
 8600            NGO=IQUIT
                 MTERR = NCOPY
                 GO TO(10000,9600,10000,290),NGO
C          READING ERROR ON TAPE NT3 DURING RESTARTING.
 8800            NGO = IQUIT
                 MTERR = NT3
                 GO TO(10000,9600,10000,295),NGO
 9500            IF(NTERR) 9530,9505,9530
 9505            WRITF OUTPUT TAPE 6, 9515, MTERR
                 PRINT 5515, MTERR
 95150           FORMAT(1H0,40HREPEATED REDUNDANCIES IN WRITING LOGICAL,
      1                6H TAPE,I2,1H.)
                 GO TO 9660
 9530            WRITE OUTPUT TAPE 6, 9545, MTERR,NTERR
                 PRINT 9545, MTERR,NTERR
 95450           FORMAT(1H0,40HREPEATED REDUNDANCIES IN WRITING LOGICAL,
      1                6H TAPF,I2,21H AND/OR LOGICAL TAPF,I2,1H.)
                 GO TO 9660
 9600            IF(NTERR) 9630,9605,9630
 9605            WRITF OUTPUT TAPE 6, 9615, MTERR
                 PRINT 9615, MTERR
 96150           FORMAT(1H0,40HREPEATED REDUNDANCIES IN READING LOGICAL,
      1                6H TAPE,I2,1H.)
                 GO TO 9660
 9630            WRITF OUTPUT TAPF 6, 9645, MTERR,NTERR
                 PRINT 9645, MTERR,NTERR
 96450           FORMAT(1H0,40HREPEATED REDUNDANCIES IN READING LOGICAL,
      1                6H TAPE,I2,21H AND/OR LOGICAL TAPE,I2,1H.)
 9660            PRINT 9675
 9675            FORMAT(1H0,28HINSPECT TAPE AND TAPF DRIVE.)
                 GO TO 9800
 9700            IF (NTERR) 9730,9705,9730
 9705            WRITF OUTPUT TAPE 6, 9715, MTERR
                 PRINT 9715, MTERR
 97150           FORMAT(1H0,40HEND OF TAPF ENCOUNTERED WHILE WRITING ON,
      1                14H LOGICAL TAPE,I2,1H.)
                 GO TO 9760
 9730            WRITF OUTPUT TAPE 6, 9745, MTERR,NTERR
```

```
              PRINT 9745,  MTERR,NTERR
97450         FORMAT(1H0,40HEND OF TAPE FNCOUNTERED WHILE WRITING ON,
    1              14H LOGICAL TAPE,I2,17H OR LOGICAL TAPE,I2,1H.)
 9760         PRINT 9775
 9775         FORMAT(1H0,34HMOUNT A LONGER TAPE ON THIS DRIVE.)
 9800         PRINT 9815
 9815         FORMAT(1H0,41HPRESS START TO RESUME PROCESSING THIS JOB)
              PAUSE
              REWIND NCOEF
              REWIND NMULT
              REWIND NCOPY
              REWIND NT1
              REWIND NT2
              REWIND NT3
              CALL SAVEIT
10000         WRITE OUTPUT TAPE 6, 10020
              REWIND NMULT
              REWIND NCOEF
              REWIND NCOPY
              REWIND NT1
              REWIND NT2
              REWIND NT3
              CALL DUMP
100200FORMAT(1H1,123HEITHER A MACHINE ERROR HAS OCCURRED OR THERE IS AN
     1 ERROR IN SUBROUTINE GAUSS OR ITS ASSOCIATED SUBROUTINES.  DUMP HA
     2S BEEN/1H,29HCALLED TO TERMINATE THIS JOB.)
          END
```

```
      SUBROUTINE SAVEIT
         CALL CHAIN(3,B3)
      END
```

```
      SUBROUTINF RSTART(NRUN)
         NRUN = 1
         RETURN
      END
```

```
      SUBROUTINE MDIVID(N,NA,IMAX,A1,A2)
C     THIS SUBROUTINE PERFORMS THE DIVISION NECESSARY IN
C     COMPUTING A NEW COLUMN OF MULTIPLIERS.
      DIMENSION A1(500),A2(500)
         T1 = A1(IMAX)
         T2 = A2(IMAX)
         A1(IMAX) = A1(NA)
         A2(IMAX) = A2(NA)
         A1(NA) = T1
         A2(NA) = T2
         IF(ABSF(T1) - ABSF(T2))  100,120,120
100         TEMP = T1/T2
            R2 = -1.0/(T2*(1.0 + TEMP**2))
            R1 = -TEMP*R2
            GO TO 140
120         TEMP = T2/T1
            R1 =  1.0/(T1*(1.0 + TEMP**2))
            R2 = -TEMP*R1
140      KS = NA + 1
         DO 160 K = KS,N,1
            TEMP = A1(K)
            A1(K) = R1*TEMP - R2*A2(K)
160         A2(K) = R1*A2(K) + R2*TEMP
         RETURN
      END
```

```
      SUBROUTINE REDUCE(N,NA,IMAX,A1,A2,AM1,AM2)
C     THIS SUBROUTINE PERFORMS THE REDUCTION OF ONE COLUMN WITH
C     ONE COLUMN OF MULTIPLIERS.  THE NECESSARY INTERCHANGE IS
C     PERFORMED.
      DIMENSION A1(500),A2(500),AM1(500),AM2(500)
         KA = NA
         KMAX=IMAX
         T1 = A1(KMAX)
         T2 = A2(KMAX)
         A1(KMAX) = A1(KA)
         A2(KMAX) = A2(KA)
         A1(KA) =T1
         A2(KA)=T2
         KS= KA + 1
         DO 100 K =KS,N,1
            A1(K) = A1(K) - AM1(K)*T1 + AM2(K)*T2
100         A2(K) = A2(K) - AM1(K)*T2 - AM2(K)*T1
         RETURN
      END
```

```
      SUBROUTINE DETER(D1,D2,DET1,DET2,NB2)
C         THIS SUBROUTINE IS USED IN COMPUTING THE DETERMINANT OF
C         THE MATRIX.
              T1 = DET1
              T2 = DET2
              DET1 = T1*D1 - T2*D2
              DET2 = T1*D2 + T2*D1
              COMP = MAX1F(ABSF(DET1),ABSF(DET2))
              NADD = LOGF(COMP)/0.69314718
              AYPL= 2.0**NADD
              DET1 = DET1/AMPL
              DET2 = DET2/AMPL
              NB2 = NB2 + NADD
              RETURN
          END
```

```
      SUBROUTINE BSOLVE(K,RHS1,RHS2,COL1,COL2,ANS1,ANS2)
C         THIS SUBROUTINE IS USED TO OBTAIN THE SOLUTION OF THE
C         REDUCED SYSTEM OF EQUATIONS.
          DIMENSION RHS1(500),RHS2(500),COL1(500),COL2(500),
    1                  ANS1(500),ANS2(500)
              N = K
              T1 = COL1(N)
              T2 = COL2(N)
              IF(ABSF(T1) - ABSF(T2)) 10,20,20
   10             TEMP = T1/T2
                  R2 = -1.0/(T2*(1.0 + TEMP**2))
                  R1 = -TEMP*R2
                  GO TO 30
   20             TEMP = T2/T1
                  R1 = 1.0/(T1*(1.0 + TEMP**2))
                  R2 = -TEMP*R1
   30         T1 = R1*RHS1(N) - R2*RHS2(N)
              T2 = R1*RHS2(N) + R2*RHS1(N)
              ANS1(N) = T1
              ANS2(N)=T2
              KS = N - 1
              IF(KS) 50,50,60
   50             RETURN
   60         DO 80 K = 1,KS,1
                  RHS1(K) = RHS1(K) - T1*COL1(K) + T2*COL2(K)
   80             RHS2(K) = RHS2(K) - T1*COL2(K) - T2*COL1(K)
              RETURN
          END
```

```
*       DOUBLE PRECISION SUBROUTINE
        LBL     DPRES
        COUNT   100
        ENTRY   DPSET                   (NSYS,REMS,IMMS,RELS,IMLS)
        ENTRY   DPRES                   (RECOL,IMCOL,ANS1,ANS2)
DPSET   CLA*    1,4
        STD     NSYS
        CCA     2,4
        ADD     =1
        STA     REMS1
        STA     REMS2
        STA     REMS3
        STA     REMS4
        CLA     3,4
        ADD     =1
        STA     IMMS1
        STA     IMMS2
        STA     IMMS3
        STA     IMMS4
        CLA     4,4
        ADD     =1
        STA     RELS1
        STA     RELS2
        STA     RELS3
        STA     RELS4
        CLA     594
        ADD     =l
        STA     IMLS1
        STA     IMLS2
        STA     IMLS3
        STA     I MLS4
        TRA     6,4
DPRES   SXA     X4,4
        CLA     1,4
        ADD     =1
        STA     RECOL1
        STA     RECOL2
        CLA     2,4
        ADD     =1
        STA     IMCOL1
        STA     IMCOL2
        CLA*    ?,4
        STO     ANS1
        CLA*    4,4
        STO     ANS2
        LXD     =0000001000000,4
REPEAT  LDQ     ANS1
RECOL1  FMP     **,4
        STQ     PROD?
        CHS
REMS1   FAD     **,4
        STO     TEMP
        XCA
RELS1   UFA     **,4
        UFS     PROD2
        FAD     TEMP
REMS2   STO     **,4
RELS2   STQ     **,4
        LDQ     ANS2
IMCOL1  FMP     **,4
        STQ     PROD2
REMS3   FAD     **,4
        STO     TEMP
```

```
        XCA
RELS3   UFA     **,4
        UFA     PROD?
        FAD     TEMP
REMS4   STO     **,4
RELS4   STQ     **,4
        LDQ     ANS1
IMCOL2  FMP     **,4
        STQ     PROD?
        CHS
IMMS1   FAD     **,4
        STO     TEMP
        XCA
IMLS1   UFA     **,4
        UFS     PROD2
        FAD     TEMP
IMMS2   STO     **,4
IMLS2   STQ     **,4
        LDQ     ANS2
RECOL2  FMP     **,4
        STQ     PROD2
        CHS
IMMS3'  FAD     **,4
        STO     TEMP
        XCA
IMLS3   UFA     **,4
        UFS     PROD?
        FAD     TEMP
IMMS4   STO     **,4
IMLS4   STQ     **,4
        TXI     *+1,4,1
NSYS    TXL     REPEAT,4,**
x4      AXT     ,4
        TRA     5,4
ANS1    PZE
ANS2    PZE
PROD1   PZE
PROD2   PZE
TEMP    PZE
        END
```

```
*       CHANNEL COMPATABILITY SUBROUTINE
        LBL     CHAN
        COUNT   20
        ENTRY   CHAN                    (NT1,NT2,NOK)
CHAN    SXA     X4,4
        CLA*    1,4
        CALL    (IOS)
        CLA*    $(ETT)
        STO     TEMP
        LXA     X4,4
        CLA*    2,4
        CALL    (IOS)
        CLA*    $(ETT)
        SUB     TEMP
X4      AXT     ,4
        TZE     OUT
        CLA     =000000C1000000
        STO*    394
        TRA     4,4
OUT     STZ*    3,4
        TRA     4,4
TEMP    PZE
        END
```

```
*       TAPE BACKSPACE SUBROUTINE
        LBL     BSPACE
        COUNT   13
        ENTRY   BSET                    (NTAPE)
        ENTRY   BSPACE
BSET    SXA     X4,4
        CLA*    1,4
        ADM     =020
        CALL    (IOS)
        CAL*    $(BSR)
        STO     BSPACE
X4      AXT     ,4
        TRA     2,4
BSPACE  PZE
        TRA     1,4
        END
```

```
*        SUBROUTINE READA
         LBL      READA
         COUNT    63
         ENTRY    QSETA                      (NTAPE,N,NRET,IQUIT)
         ENTRY    READA                      (NREAD,A1,A2)
         ENTRY    RCHKA
RSETA    SXA      X4,4
         CAL*     1,4
         ACL      =O20
         CALL     (IOS)
         LDQ*     $(TRC)
         SLQ      TRC
         LDQ*     $(TCO)
         SLQ      TCO
         LDQ*     $(RCH)
         SLQ      RCH
         CAL*     $(RDS)
         SLW      RDS
         STA      BSR
x 4      AXT      ,4
         CLA*     2,4
         ARS      18
         SUB      =1
         STO      TEMP
         CLA*     3,4
         STA      NRET
         CLA      4,4
         STA      IQUIT
         TRA      5,4
READA    CLA*     194
         STD      RIO
         STD      RIO+1
         CLA      2,4
         SUB      TEMP
         STA      RIO
         CLA      394
         SUR      TEMP
         STA      RIO+1
RDS      PZF
RCH      PZE      RIO
         TRA      4,4
RIO      IOCP     **,,**
         IOCD     **,,**
RCHKA    STZ      NBAD
TCO      PZE      *
TRC      PZE      ERROR
         TRA      1,4
ERROR    SXA      S4,4
         LXD      NBAD,4
         TXI      *+1,4,1
         TXH      QUIT,4,5
         SXD      NBAD,4
BSR      BSR      **
         XEC      RDS
         XEC      RCH
S4       AXT      **,4
         TRA      TCO
QUIT     CLA      =O000002000000
IQUIT    STO      **
         CAL      RDS
         STA      *+1
         REW      **
NRET     TRA      **

NBAD     PZE
TEMP     PZE
         END
```

```
*        SUBROUTINE WRITEA
         LBL       WRITCA
         COUNT     80
         ENTRY     WSETA                   (NTAPE,N,NRET,IQUIT)
         ENTRY     WPITFA                  (WRITE ,A1 ,A2 )
         ENTRY     WCHKA
WSFTA    SXA       X4,4
         CAL*      1,4
         ACL       =O20
         CALL      (ICS)
         LDQ*      $(TRC)
         SLQ       TRC
         LDQ*      $(TCO)
         SLQ       TCO
         LDQ*      $(RCH)
         SLQ       RCH
         CAL*      $(ETT)
         SLW       ETT
         SLW       ETTOFF
         CAL*      $(WRS)
         SLW       WRS
         STA       BSR
X4       AXT       ,4
         CLA*      2,4
         ARS       18
         SUR       =1
         STO       TEMP
         CLA*      3,4
         STA       NRET1
         STA       NRET2
         CLA       4,4
         STA       IQUIT1
         STA       IQUIT2
ETTOFF   PZF
         NOP
         TRA       5,4
WRITEA   CLA*      1,4
         STD       WIO
         STD       WIO+1
         CLA       2,4
         SUB       TEMP
         STA       WIO
         CLA       3,4
         SUB       TEMP
         STA       WIO+1
ETT      PZE
         TRA       SHORT
WRS      PZE
RCH      PZF       WIC
         TRA       4,4
WIO      IOCP      **,,**
         IOCD      **,,**
SHORT    CLA       =O000003000000
IQUIT1   STG       **
         CAL       WRS
         STA       *+1
         REW       **
NRET1    TRA       **
WCHKA    STZ       NBAD
TCO      PZE       *
TRC      PZE       FRROR
         TRA       1,4
ERROR    SXA       S4,4
```

```
        LXD    NBAD,4
        TXI    *+1,4,1
        TXH    QUIT,4,10
        SXD    NBAD,4
BSR     BSR    **
        XEC    WRS
        XEC    WRS
        XEC    WRS
        XEC    RCH
54      AXT    **,4
        TRA    TCO
QUIT    CLA    =0000001000000
IQUIT2  STO    **
        CAL    WRS
        STA    *+1
        REW    **
NRET2   TRA    **
NBAD    PZE
TEMP    PZE
        END
```

```
*       SUBROUTINE  READB
        LBL     READB
        COUNT   63
        ENTRY   RSETB                       (NTAPE,N,NRET,IQUIT)
        ENTRY   READB                       (NREAD,A1,A2)
        ENTRY   RCHKB
RSETB   SXA     X4,4
        CAL*    1,4
        ACL     =O20
        CALL    (IOS)
        LDQ*    $(TRC)
        SLQ     TRC
        LDQ*    $(TCO)
        SLQ     TCO
        LDQ*    $(RCH)
        SLQ     RCH
        CAL*    $(RDS)
        SLW     RDS
        STA     BSR
x 4     AXT     ,4
        CLA*    2,4
        ARS     18
        SUB     =1
        STO     TEMP
        CLA*    3,4
        STA     NRET
        CLA     4,4
        STA     IQUIT
        TRA     5,4
READB   CLA*    1,4
        STD     RIO
        STD     RIO+1
        CLA     2,4
        SUB     TEMP
        STA     RIO
        CLA     3,4
        SUB     TEMP
        STA     RIO+1
RDS     PZE
RCH     PZF     RIO
        TRA     4,4
RIO     IOCP    **,,**
        IOCD    **,,**
RCHKB   STZ     NBAD
TCO     PZE     *
TRC     PZE     ERROR
        TRA     1,4
ERROR   SXA     S4,4
        LXD     NBAD,4
        TXT     *+1,4,1
        TXH     QUIT,4,5
        SXD     NBAD,4
BSR     BSR     **
        XEC     RDS
        XEC     RCH
S4      AXT     **,4
        TRA     TCO
QUIT    CLA     =O00000C z o o o o o o .
IQUIT   STO     **
        CAL     RDS
        STA     *+1
        REW     **
NRET    TRA     **

NBAD'   PZE
TEMP    PZE
        END
```

```
*       SUBROUTINE     WRITEB
        LBL        WRITFB
        COUNT      80
        ENTRY      WSETB                    (NTAPE,N,NRET,IQUIT)
        ENTRY      WRITFB                   (NWRITE,A1,A2)
        ENTRY      WCHKR
WSETB   SXA        X4,4
        CAL *      1,4
        ACL        =020
        CALL       (IOS)
        LDQ*       $(TRC)
        SLQ        TRC
        LDQ*       $(TCO)
        SLQ        TCO
        LDQ*       $(RCH)
        SLQ        RCH
        CAL*       $(ETT)
        SLW        FTT
        SLW        ETTOFF
        CAL*       $(WRS)
        SLW        WRS
        STA        RSR
x 4     AXT        ,4
        CLA*       2,4
        ARS        18
        SUR        =1
        STO        TEMP
        CLA*       3,4
        STA        NRET 1
        STA        NRET2
        CLA        4,4
        STA        IQUIT1
        STA        IQUI T2
FTTOFF  PZF
        NOP
        TRA        5,4
WRITEB  CLA*       1,4
        STD        WIO
        STD        WIO+1
        CLA        2,4
        SUB        TEMP
        STA        WI0
        CLA        3,4
        SUB        TEMP
        STA        WIO+1
ETT     PZF
        TRP        SHORT
WRS     PZE
RCH     PZF        wI0
        TRA        4,4
WI0     IOCP       **,,**
        IOCD       **,,**
SHORT   CLA        =0000003000000
IQUI T1 STO        **
        CAL        WRS
        STA        *+1
        REW        **
NRET1   TRA        **
WCHKB   ST7        NRAD
TCO     PZE        *
TRC     PZE        FRROR
        TRA        1,4
ERROR   SXA        S4,4
```

```
        LXD     NBAD,4
        TXI     *+1,4,1
        TXH     QUIT,4,10
        SXD     NBAD,4
BSR     BSR     **
        XEC     WRS
        XEC     WRS
        XEC     WRS
        XEC     RCH
54      AXT     **,4
        TRA     TCO
QUIT    CLA     =0000001000000
IQUIT2  STO     **
        CAL     WRS
        STA     *+1
        REW     **
NRFT2   TRA     **
NBAD    PZF
TEMP    PZE
        END
```

```
*        SURROUTINF R=ADC
         LBL      REPDC
         COUNT    62
         ENTRY    RSETC                     (NTAPE,NRET.IQUIT)
         ENTRY    READC                     (NREAD,A1,A2)
         ENTRY    RCHKC
RSETC    SXA      X4,4
         CAL*     1,4
         ACL      =020
         CALL     (IOS)
         LDQ*     $(TRC)
         SLQ      TRC
         LDQ*     $(TCO)
         SLQ      TCO
         LDQ*     $(RCH)
         SLQ      RCH
         CAL*     $(RDS)
         SLW      RDS
         STA      RSR
x 4      AXT      ,4
         CLA*     2,4
         STA      NRET
         CLA      3,4
         STA      IQUIT
         TPA      4,4
READC    CLA*     1,4
         STD      R IO
         STD      RIO+1
         ARS      18
         SUB      =1
         STO      TEMP
         CLA      2,4
         SUB      TEMP
         STA      RIO
         CLA      3,4
         SUB      TEYP
         STA      RIO+1
RDS      PZE
RCH      PZE      RIO
         TRA      4,4
RIO      IOCP     **,,**
         IOCD     **,,**
RCHKC    STZ      NRAD
TCO      PZF      *
TRC      PZE      FRROR
         TRA      1,4
ERROR    SXA      S4,4
         LXD      NBAD,4
         TXI      *+1,4,1
         TXH      QUIT,4,5
         SXD      NBAD,4
BSR'     BSR      **
         XEC      RDS
         XEC      RCH
s 4      AXT      **,4
         TRA      TCO
QUIT     CLA      =0000002000000
IQUIT    STO      **
         CAL      RDS
         STA      *+1
         REW      **
NRET     TRA      **
NBAD     PZE
TEMP     PZE
         END
```

```
*         SUBRCUTINE WRITEC
          LBI       WRITFC
          COUNT     79
          ENTRY     WSFTC                           (NTAPF,NRET,IQUIT)
          ENTRY     WRITEC                          (NWRITE,A1,A2)
          ENTRY     WCHKC
WSETC     SXA       X4,4
          CAL"      1,4
          ACL       =02C
          CALL      (IOS)
          LDQ*      $(TRC)
          SLQ       TRC
          LDQ*      $(TCO)
          SLQ       TCO
          LDQ*      $(RCH)
          SLQ       RCH
          CAL*      $(ETT)
          SLW       ETT
          SLW       ET TOFF
          CAL*      $(WRS)
          SLW       WRS
          STA       BSR
x 4 -     AXT       ,4
          CLA*      2,4
          STA       NRET1
          STA       NRET2
          CLA       3,4
          STA       IQUIT1
          STA       IQUIT2
ETTOFF    PZE
          NOP
          TRA       4,4
WRITEC    CLA*      1,4
          STD       WIO
          STD       WIO+1
          ARS       18
          SUB       =1
          STO       TEMP
          CLA       2,4
          SUB       TEMP
          STA       WIO
          CLA       3,4
          SUB       TEMP
          STA       WIO+1
ETT       PZE
          TRA       SHORT
WRS       PZE
RCH       PZE       WIO
          TRA       4,4
WIO       IOCP      **,,**
          IOCD      **,,**
SHORT     CLA       =0000003000000
IQUIT1    STO       **
          CAL       WRS
          STA       *+1
          REW       **
NRET1     TRA       **
WCHKC     STZ       NBAD
TCO       PZE       *
TRC       PZF       ERROR
          TRA       1,4
ERROR     SXA       $4,4
          LXD       NBAD,4
```

```
        TXI    *+1,4,1
        TXH    QUIT,4,10
        SXD    NBAD,4
BSR     BSR    **
        XEC    WRS
        XEC    WRS
        XEC    WRS
        XEC    RCH
S4      AXT    **,4
        TRA    TCC
QUIT    CLA    =0000001000000
IQUIT2  STO    **
        CAL    WRS
        STA    *+1
        REW    **
NRFT2   TRA    **
NBPD    PZE
TEMP    PZE
        END
```