IBM Application Program

H20-0205-3

System/360 Scientific Subroutine Package

(360A-CM-03X) Version III

Programmer's Manual

The Scientific Subroutine Package (SSP) is a collection of over 250 FORTRAN subroutines divided, for the sake of presentation, into two groups: statistics and mathematics. Also, over 200 subroutines are presented in both single and double precision mode. SSP is a collection of input/output-free computational building blocks that can be combined with a user's input, output, or computational routines to meet his needs. The package can be applied to the solution of many problems in industry, science, and engineering.

Version 3 of the Scientific Subroutine Package for System/360 incorporates and extends the capabilities of the original SSP/360. This version provides over 40 new mathematical and statistical subroutines 24 of which are in both single- and double-precision FORTRAN. Examples of the new capabilities are the use of the QR iteration for obtaining eigenvalues of a matrix, and the nonparametric test of Kolmogorov-Smirnov.

This manual contains sufficient information to permit the reader to understand and use all of the subroutines of the Scientific Subroutine Package.

DOI number: 10.3247/SL2Soft08.001 DOI link: http://dx.doi.org/10.3247/sl2soft08.001

Fourth Edition

This is a major revision of, and obsoletes H20-0205-2.

New machine listings have been provided for all routines.

Specifications contained herein are subject to change from time to time.
Any such change will be reported in subsequent revisions or Technical Newsletters.

Copies of this and other IBM publications can be obtained through IBM branch offices. Address comments concerning the contents of this publication to:
IBM, Technical Publications Department, 112 East Post Road, White Plains, N. Y. 10601

© International Business Machines Corporation 1966, 1967, 1968.

CONTENTS

Purpose and Objectives of the Package . . . 1
Areas of Application 1
Statistics 1
Mathematics 1
Characteristics 1
Overall Rules of Usage 3
General Rules 3
Matrix Operations 3
Variable Dimensioning 3
Storage Compression 4
Matrix Element References 5
Program Modification 7
Advanced FORTRAN Features 7
Optimization of Time 7
Double Precision 8
Format of the Documentation 9
Subroutine Descriptions 9
Sample Program Descriptions 9
Operating Notes 10
Categorical Guide to Subroutines and Sample Programs 11
Alphabetic Guide to Subroutines with Storage Requirements 19
Subroutine Descriptions and Listings 27
Statistics 27
Mathematics 94
Appendix A. Accuracy of Subroutines 386
Appendix B. Sample Program Descriptions . 400

This programming package has been developed with the cooperation and assistance of IBM Germany and IBM France.

PURPOSE AND OBJECTIVES OF THE PACKAGE

The Scientific Subroutine Package is a set of basic computational subroutines intended to aid the user in the development of his own FORTRAN subroutine library. While this package may provide many of the tools necessary to solve the more commonly encountered problems in engineering and science, there is no intent to imply that these subroutines represent the current state of the art in statistics or numerical analysis. As with all tools, the user should understand their capabilities and their application to his functional requirements before deciding to use them.

AREAS OF APPLICATION

Individual subroutines, or a combination of them, can be used to carry out the listed functions in the following areas:

Statistics

- Probit analysis
- Analysis of variance (factorial design)
- Correlation analysis
- Multiple linear regression
- Stepwise regression
- Polynomial regression
- Canonical correlation
- Factor analysis (principal components, varimax)
- Discriminant analysis (many groups)
- Time series analysis
- Data screening and analysis
- Nonparametric tests
- Random number generation (uniform, normal)
- Distribution functions

Mathematics

- Inversion
- Eigenvalues and eigenvectors
- Simultaneous linear algebraic equations
- Transpositions
- Matrix arithmetic (addition, product, etc.)
- Matrix partitioning
- Matrix tabulation and sorting of rows or columns
- Elementary operations on rows or columns of matrices
- Matrix factorization
- Integration and differentiation of given or tabulated functions
- Solution of systems of first-order differential equations
- Fourier analysis of given or tabulated functions
- Bessel and modified Bessel function evaluation
- Gamma function evaluation
- Jacobian elliptic functions
- Elliptic, exponential, sine cosine, Fresnel integrals
- Finding real roots of a given function
- Finding real and complex roots of a real polynomial
- Polynomial arithmetic (addition, division, etc.)
- Polynomial evaluation, integration, differentiation
- Chebyshev, Hermite, Laguerre, Legendre polynomials
- Minimum of a function
- Approximation, interpolation, and table construction

CHARACTERISTICS

Some of the characteristics of the Scientific Subroutine Package are:

- All subroutines are free of input/output statements.
- Subroutines do not contain fixed maximum dimensions for the data arrays named in their calling sequences.
- All subroutines are written in FORTRAN.
- Many matrix manipulation subroutines handle symmetric and diagonal matrices (stored in economical, compressed formats) as well as general matrices. This can result in considerable saving in data storage for large arrays.
- The use of the more complex subroutines (or groups of them) is illustrated in the program documentation by sample main programs with input/output.
- All subroutines are documented uniformly.

The subroutines in SSP have been programmed in the subset of the FORTRAN IV language, which is compatible with all of the FORTRAN processors announced for System/360. Many of the larger functions such as those in statistics have been programmed as a series or sequence of subroutines.

An example of the use of sequences of subroutines is the statistical function called factor analysis. Factor analysis is a method of analyzing the intercorrelations within a set of variables. It determines whether the variance in the original set of variables can be accounted for adequately by a

smaller number of basic categories; namely, factors. In the Scientific Subroutine Package, factor analysis is normally performed by calling the following five subroutines in sequence:

1. CORRE - to find means, standard deviations, and correlation matrix
2. EIGEN - to compute eigenvalues and associated eigenvectors of the correlation matrix
3. TRACE - to select the eigenvalues that are greater than or equal to the control value specified by the user
4. LOAD - to compute a factor matrix
5. VARMX - to perform varimax rotation of the factor matrix

The multiple use of subroutines is illustrated by the fact that subroutine CORRE is also utilized in the multiple linear regression and canonical correlation. Subroutine EIGEN is used in canonical correlation as a third-level subroutine.

OVERALL RULES OF USAGE

GENERAL RULES

All subroutines in the Scientific Subroutine Package (SSP) are entered by means of the standard FORTRAN CALL statement. These subroutines are purely computational in nature and do not contain any references to input/output devices. The user must therefore furnish, as part of his program, whatever input/output and other operations are necessary for the total solution of his problem. In addition, the user must define by DIMENSION statements all matrices to be operated on by SSP subroutines as well as those matrices utilized in his program. The subroutines contained in SSP are no different from any user-supplied subroutine. All of the normal rules of FORTRAN concerning subroutines must therefore be adhered to, with the exception that the dimensioned areas in the SSP subroutine are not required to be the same as those in the calling program.

The CALL statement transfers control to the subroutine and replaces the dummy variables in that subroutine with the value of the actual arguments that appear in the CALL statement if the argument is a constant or a variable. When the argument is an array or function subprogram name, the address of the array or subprogram is transmitted to the called subroutine.

The arguments in a CALL statement must agree in order, number, and type with the corresponding arguments in the subroutine. A number may be passed to a subroutine either as a variable name in the argument list or as a constant in the argument list. For example, if the programmer wishes to add matrix AR1 to matrix AR2 in order to form matrix AR3 using the SSP subroutine GMADD, and if AR1 and AR2 are both matrices with ten rows and twenty columns, either of the two following methods can be used:

Method 1

```
.
CALL GMADD(AR1,AR2,AR3,10,20)
.
```

Method 2

```
.
N = 10
M= 20
CALL GMADD(AR1,AR2,AR3,N,M)
.
```

Many of the subroutines in SSP require the name of a user function subprogram or a FORTRAN-supplied function name as part of the argument list in the CALL statement. If the user's program contains such a CALL, the function name appearing in the argument list must also appear in an EXTERNAL statement at the beginning of that program.

For example, the SSP subroutine RK2 integrates a function furnished by the user. The user must therefore program the function and give the name of the function to RK2 as a parameter in the CALL statement. If the user wishes to integrate the function $\frac{dy}{dx} = 3.0\,x + 2.0Y$, his main program may appear as:

```
EXTERNAL DERY
.
.
CALL RK2(DERY, .....)
.
.
RETURN
END
```

His function subprogram can be:

```
FUNCTION DERY (X, Y)
DERY= 3.0*X+2.0*Y
RETURN
END
```

The user's main program gives the name of the programmed function to RK2 by including that name in the CALL statement and in an EXTERNAL statement. RK2, in turn, goes to the function DERY each time it requires a value for the derivative. The subroutine RK2 is not modified by the programmer. The dummy function name FUN in subroutine RK2 is, in effect, replaced by the name appearing in the user's CALL statement during execution of the subroutine.

MATRIX OPERATIONS

Special consideration must be given to the subroutines that perform matrix operations. These subroutines have two characteristics that affect the format of the data in storage--variable dimensioning and data storage compression.

Variable Dimensioning

Those subroutines that deal with matrices can operate on any size array, limited in most cases only by the available core storage and numerical analysis considerations. The subroutines do not contain fixed maximum dimensions for data arrays named in their calling sequence. The variable dimension capability

has been implemented in SSP by using a vector storage approach. Under this approach, each column of a matrix is immediately followed in storage by the next column. Vector storage and two-dimensional storage result in the same layout of data in core, so long as the number of rows and columns in the matrix are the same as those in the user's dimension statement. If, however, the matrix is smaller than the dimensioned area, the two forms of storage are not compatible.

Consider the layout of data storage when operating on a 5 by 5 array of numbers in an area dimensioned as 10 by 10. If the programmer has been using double-subscripted variables in the normal FORTRAN sense, the 25 elements of data will appear as shown in Figure 1. FORTRAN stores double-subscripted data by column based on the column length specified in the DIMENSION statement. Thus, in the example, sequential core locations will contain data elements 1 to 5, five blank locations, data elements 6 to 10, five blank locations, and so on. The matrix subroutines take a vector approach in storing arrays by column, which means that they assume the data is stored as shown in Figure 2.

		Column									
		1	2	3	4	5	6	7	8	9	10
	1	(1)	(6)	(11)	(16)	(21)					
	2	(2)	(7)	(12)	(17)	(22)					
	3	(3)	(8)	(13)	(18)	(23)					
	4	(4)	(9)	(14)	(19)	(24)					
Row	5	(5)	(10)	(15)	(20)	(25)					
	6										
	7										
	8										
	9										
	10										

Figure 1. Double-subscripted data storage

(1)	(11)	(21)
(2)	(12)	(22)
(3)	(13)	(23)
(4)	(14)	(24)
(5)	(15)	(25)
(6)	(16)	
(7)	(17)	
(8)	(18)	
(9)	(19)	
(10)	(20)	

Figure 2. Vector storage

As stated previously, if the dimensioned area is the same as the matrix size, the two approaches will have the same data storage layout, and the user can proceed in a regular double-subscripted fashion. If, however, he is operating in a mode where the dimensioned area is larger than the arrays, and if he wishes to use the SSP subroutines, he must be certain that his data is stored in the vector fashion illustrated by Figure 2. A subroutine called ARRAY is available in SSP to change from one form of storage to the other. In addition, a subroutine called LOC is available to assist in referencing elements in an array stored in the vector fashion.

Storage Compression

Many subroutines in SSP can operate on compressed forms of matrices as well as on the normal form. By using this capability, which is called "storage mode", considerable savings in data storage can be obtained for special forms of large arrays. The three modes of storage are termed general, symmetric, and diagonal. In this context, general mode is one in which all elements of the matrix are in storage. Symmetric mode is one in which only the upper triangular portion of the matrix is retained columnwise in sequential locations in storage. (The assumption is made that the corresponding elements in the lower triangle have the same value.) Diagonal mode is one in which only the diagonal elements of the matrix are retained in sequential locations in storage. (The off-diagonal elements are assumed to be zero.) This capability has been implemented using the vector storage approach.

Figure 3 illustrates the effect of the storage mode capability. A symmetric matrix is shown in Figure 3A. If this array is to be manipulated using the SSP matrix subroutines with storage mode capability, the array may be stored as shown in Figure 3B. This is the upper triangular portion of the array and corresponds to a storage mode code of 1. Symmetric matrices of order N may be stored in a vector of only N*(N+1)/2 locations rather than N*N locations. For larger matrices, this will be a saving of almost one half.

The effect of storage mode when dealing with diagonal matrices is even more pronounced. Diagonal matrices of order N may be stored in a vector only N locations long. Figure 3C shows a 3 by 3 diagonal matrix. If this array is to be manipulated using the SSP matrix subroutines with storage mode capability, only the diagonal elements of the array need be stored. This is shown in Figure 3D and corresponds to a storage mode code of 2.

General matrices of order N by M require a vector N*M long and use a storage mode code of 0.

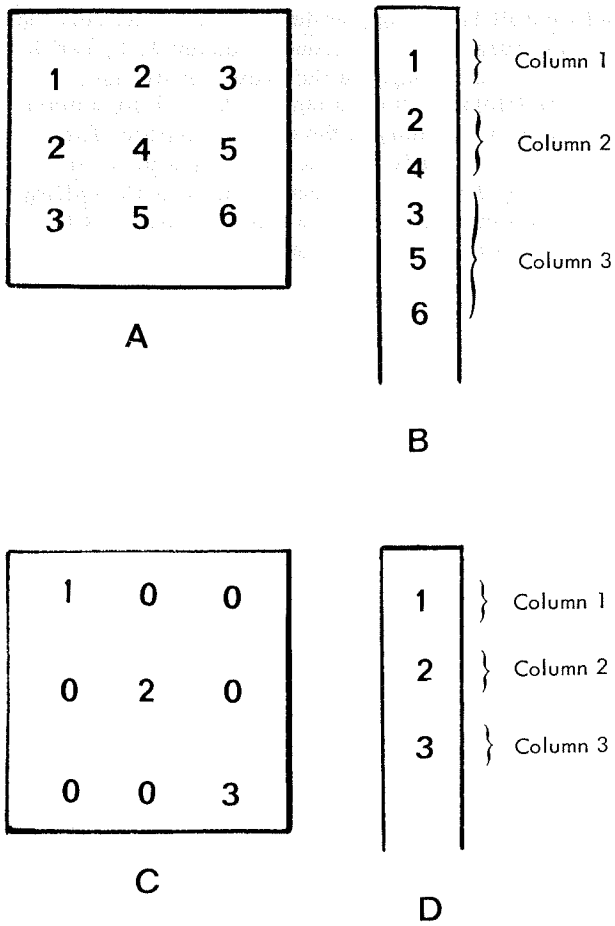


Figure 3. Storage mode

Thus, if the programmer wishes to use SSP subroutines on matrix A, which is general, matrix B, which is symmetric, and matrix C, which is diagonal, and all matrices are 10 by 10 or smaller, the dimension statement in his program can be:

```
DIMENSION A(100), B(55), C(10)
```

Matrix Element References

Subroutine LOC in the Scientific Subroutine Package may be used to reference elements within a matrix stored in a vector fashion, and may involve storage mode compression. The calling sequence for LOC is:

```
CALL LOC (I, J, IJ, N, M, MS)
```

The capabilities of subroutine LOC are as follows. If reference is required to the element at row I and column J of matrix A whose dimensions are N by M, and if the storage mode code is MS, a CALL to the LOC subroutine as shown above will result in the computation of the subscript IJ such that A(IJ) is the desired element. The parameters represented by I, J, N, M, MS can be either integer variables or integer constants. The parameter represented by IJ is an integer variable. Note that the user must dimension the array A as a single-subscripted variable to meet the restrictions of some FORTRAN systems. To illustrate the use of LOC, if reference is required to the element at row 2, column 2 of the 3 by 3 symmetric matrix illustrated in Figure 3A and stored as shown in Figure 3B (storage mode code 1), the sequence may be:

```
CALL LOC (2, 2, IJ, 3, 3, 1)
```

The value of IJ computed by LOC will be 3, meaning that the proper element is the third element in the specially stored symmetric matrix (Figure 3B). If the storage mode code is for a symmetric matrix where only the upper triangular portion is retained in storage, and if I and J refer to an element in the lower triangular portion, IJ will contain the subscript for the corresponding element in the retained upper triangle. Thus if the user wanted the element in row 3, column 1 of the matrix shown in Figure 3A, and the array was stored as in Figure 3B, the statement:

```
CALL LOC (3, 1, IJ, 3, 3, 1)
```

would result in IJ having the value of 4; that is, the fourth element in Figure 3B. If a matrix is stored as shown in Figure 3D (storage mode 2), and LOC is used to compute the subscript for an off-diagonal element (I not equal to J), the result in IJ will be zero. This is due to the fact that the element does not exist in storage. In this situation, the user must not utilize IJ as a subscript. Following is an illustration of how to take care of this condition and also handle the case where the current storage mode is unknown.

If the user wishes to set a variable X equal to the element in the third row and fourth column of a 10 by 10 array named A, for either a symmetric, diagonal, or general matrix, the required program can be implemented for any storage mode MS as follows:

```
      CALL LOC (3, 4, IJ, 10, 10, MS)
      X = 0.0
      IF(IJ)20, 30, 20
   20 X = A(IJ)
   30 --------
```

(MS is assumed to have been set at 0, 1, or 2 at some earlier point in the program.) This sequence will then set the proper value for X, given any storage mode that may be encountered. The second and third statements take care of the off-diagonal condition for a matrix with a storage mode of 2.

As a special case, LOC can be used to compute the total length of an array in storage with a statement such as:

```
CALL LOC (N, M, IJ, N, M, MS)
```

For example, if the user has a 3 by 3 matrix whose storage mode is 1 (Figure 3B), the statement:

```
CALL LOC (3, 3, IJ, 3, 3, 1)
```

will result in IJ being set to 6. This is not only the proper subscript to reference element 3, 3, but is also the actual length of the vector in storage.

The information contained in the fifth parameter (number of columns) in the calling sequence for LOC is not actually used in the calculations performed by LOC. It has been included in the calling sequence in case the user wishes to expand LOC to cover other forms of data storage.

PROGRAM MODIFICATION

ADVANCED FORTRAN FEATURES

The user may wish to take advantage of more advanced FORTRAN compiler features. For example, the larger FORTRAN systems include a subroutine multiple entry point capability. Some of the subroutines in SSP can be combined by using this feature. To illustrate this, GMADD and GMSUB (addition and subtraction of general matrices) can be combined as shown in Figure 4. This will in no way affect the programs using these subroutines but will reduce the total program storage requirements.

```
      SUBROUTINE GMADD (A,B,R,N,M)
      DIMENSION A(1),B(1),R(1)
      K=0
      GO TO 10
      ENTRY GMSUB (A,B,R,N,M)
      K=1
10    NM=N*M
      DO 40 I=1,NM
      IF(K) 20,20,30
20    R(I)=A(I)+B(I)
      GO TO 40
30    R(I)=A(I)-B(I)
40    CONTINUE
      RETURN
      END
```

Figure 4. Multiple entry point

OPTIMIZATION OF TIME

The subroutines in SSP are designed to conserve storage for data. If the user wishes to exchange space for time, there are several ways in which SSP may be modified to effect this end. For example, many of the subroutines in SSP make use of the LOC subroutine to handle vector storage and storage mode referencing. The execution time of these subroutines can be substantially reduced by implementing LOC in Assembler Language. (The distributed version of LOC is implemented in FORTRAN.) Another approach is to incorporate the function of LOC within each subroutine and thus avoid the "set-up" costs of repeated calls to LOC. This has the effect of reducing execution time but at some cost in subroutine storage and in the ease with which other modes of storage, such as triangular matrix storage or storage by row rather than by column, can be implemented. Figure 5 shows how matrix addition and the LOC capabilities can be implemented within the same subroutine.

```
      SUBROUTINE MADX(A,B,R,N,M,MSA,MSB)
      DIMENSION A(1),B(1),R(1)
C
C        TEST FOR SAME STORAGE MODE
C
      IF(MSA-MSB) 30,10,30
C
C        COMPUTE VECTOR LENGTH
C
   10 ND=N*M
      IF(MSA-1) 24,22,23
   22 ND=(ND+N)/2
      GO TO 24
   23 ND=N
C
C        ADD MATRICES OF SAME STORAGE MODE
C
   24 DO 25 I=1,ND
   25 R(I)=A(I)+B(I)
      RETURN
C
C        GET STORAGE MODE OF OUTPUT MATRIX
C
   30 MTEST=MSA*MSB
      MSR=0
      IF(MTEST) 35,35,32
   32 MSR=1
   35 DO 60 J=1,M
      DO 60 I=1,N
C
C        LOCATE ELEMENT IN OUTPUT MATRIX
C
      KX=-1
      MS=MSR
      GO TO 65
   40 IJR=IR
C
C        LOCATE ELEMENT IN MATRIX A
C
      KX=0
      MS=MSA
      GO TO 65
   45 IJA=IR
      AEL=0.0
      IF(IJA) 46,48,46
   46 AEL=A(IJA)
C
C        LOCATE ELEMENT IN MATRIX B
C
   48 KX=1
      MS=MSB
      GO TO 65
   50 IJB=IR
      BEL=0.0
      IF(IJB) 55,60,55
   55 BEL=B(IJB)
C
C        ADD MATRICES OF DIFFERENT STORAGE MODES
C
   60 R(IJR)=AEL+BEL
      RETURN
C
C        IN LINE LOC
C
   65 IF(MS-1) 70,75,90
   70 IR=N*(J-1)+I
      GO TO 95
   75 IF(I-J) 80,85,85
   80 IR=I+(J*J-J)/2
      GO TO 95
   85 IR=J+(I*I-I)/2
      GO TO 95
   90 IR=0
      IF(I-J) 95,92,95
   92 IR=I
   95 IF(KX) 40,45,50
      END
```

Figure 5. Inline LOC

DOUBLE PRECISION

The accuracy of the computations in many of the SSP subroutines is highly dependent upon the number of significant digits available for arithmetic operations. Matrix inversion, integration, and many of the statistical subroutines fall into this category. The user may, therefore, wish to use double-precision versions of these subroutines.

Many of the subroutines in SSP have been provided in both single- and double-precision versions. In others, instructions for converting to double precision are included as part of the subroutine description. FORTRAN double-precision statements have been included in each of these subroutines in the form of a comments card. In most cases, the double-precision version of the subroutine can be obtained merely by removing the C from column 1 of the double-precision statement card(s) before compilation. In a few cases, additional instructions are given for changing references to FORTRAN-furnished functions; for example, SQRT to DSQRT and ABS to DABS; in other cases, new constants are given.

The use of double-precision subroutines requires a detailed knowledge of the FORTRAN rules concerning double precision. A few of the more basic ones are as follows:

1. Any real variable, vector, or array name contained in the argument list of a CALL to a double-precision subroutine must be declared as double precision in the calling program. For example, if the SSP matrix inversion subroutine MINV has been modified for double precision, and if the user's program contains the statement:

```
CALL MINV (AR1, N, DET, MY1, MY2)
```

where:

AR1 is the array name (real, double-precision array)
N is the dimension (integer variable)
DET is the determinant (real, double-precision variable)
MY1 is a work vector (integer array)
MY2 is a work vector (integer array)

then the program must also have at its beginning the specification statement:

```
DOUBLE PRECISION AR1, DET
```

The other variables are integer variables and should not be included in the specification statement.

2. Any user-supplied function named in the CALL statement for a double-precision SSP subroutine must be programmed as a double-precision function. For example, the sample program for Runge-Kutta integration performs the integration on a sample function called FUN using subroutine RK2. This function is programmed as:

```
FUNCTION FUN (X, Y)

FUN = 1.1*X

RETURN

END
```

If a double-precision version of RK2 is used, the function must also be double precision. The sample function will then be:

```
DOUBLE PRECISION FUNCTION FUN (X, Y)

DOUBLE PRECISION X, Y

FUN = 1.1*X

RETURN

END
```

FORMAT OF THE DOCUMENTATION

The major portion of this manual consists of the documentation for the individual subroutines and the sample programs.

SUBROUTINE DESCRIPTIONS

Subroutine and sample program guides, both categorial and alphabetic, designed to aid in locating any particular subroutine are given in the pages that follow.

Each of the subroutine descriptions consists of a program listing and, in some cases, a mathematical description. The first part of the program listing is a set of comment cards containing the subroutine name, its calling sequence, description of parameters, remarks, names of other subroutines required, and method. References to books and periodicals will be found under the method section of the description. The mathematical description pages do not, in most cases, indicate the derivation of the mathematics. They are intended to indicate what mathematical operations are actually being performed in the subroutines. Some of the major statistical functions are performed by a sequence of SSP subroutines. An explanation of this sequence will be found just before the description of the first subroutine that is specific to this function.

SAMPLE PROGRAM DESCRIPTIONS

The sample program listings are given in Appendix B.

Each sample program consists of a detailed description including information on the problem, the program, input, output, program modification, operating instructions, error messages, timing and machine listings of the programs, input data and output results. The sample programs have been chosen to (1) illustrate a sequence of SSP subroutines, (2) illustrate the use of a complex subroutine, or (3) show the way in which one member (such as matrix addition) of a large set of simple subroutines might be used.

As part of the development of the sample programs, some special sample subroutines have been implemented that may prove useful to the programmer. These include:

HIST - print a histogram of frequencies

MATIN - read an input matrix into storage in vector form for use by SSP matrix subroutines

PLOT - plot several variables versus a base variable

MXOUT - print a matrix stored in the SSP vector format

Listings of the above subroutines are included after the sample program documentation in this manual. The source decks are part of the sample program source decks on the distributed magnetic tape.

The sample programs have been implemented for execution on a 32K byte (8K word) System/360 using Basic Programming Support FORTRAN (Tape). Instructions for modifying the sample programs for different data capacities are included in the documentation. In addition, those sample programs that illustrate potentially double-precision subroutines include double-precision statements in the form of comment cards. These comment cards are contained in the sample program source decks.

OPERATING NOTES

It is recommended that those SSP subroutines that will be frequently used in an installation be compiled and the relocatable decks placed on the FORTRAN system's residence device. In the case of Basic Programming Support FORTRAN (Tape), this will be the library portion of the system tape. Information on the method for updating the system to include user-supplied subroutines will be found in the appropriate FORTRAN programmer's guide. SSP subroutines are handled in the same manner as user-supplied subroutines. If the subroutines are not placed on the FORTRAN system's residence device, those required by a particular program will have to be included in that program each time it is run.

CATEGORIAL GUIDE TO SUBROUTINES AND SAMPLE PROGRAMS

(Subroutines added in Version III are marked with an asterisk)

STATISTICS

Data Screening

TALLY--totals, means, standard deviations, minimums, and maximums 27

BOUND--selection of observations within bounds 27

SUBST--subset selection from observation matrix 28

ABSNT--detection of missing data 28

TAB1--tabulation of data (one variable) 29

TAB2--tabulation of data (two variables) 30

SUBMX--building of subset matrix 31

Correlation and Regression (See Smoothing, Factorization)

CORRE--means, standard deviations, and correlations 32

*MISR--means, standard deviations, third and fourth moments, correlations, simple regression coefficients and their standard errors; considers that data may be missing 33

ORDER--rearrangement of intercorrelations 36

MULTR--multiple linear regression 37

GDATA--data matrix generation for polynomial regression 39

*STPRG--stepwise multiple linear regression 41

*PROBT--probit analysis 44

CANOR--canonical correlation 47

Design Analysis (See Smoothing, Regression, Factorization)

AVDAT--data storage allocation 49

AVCAL--Σ and Δ operation 50

MEANQ--mean square operation 57

Discriminant Analysis

DMATX--means and dispersion matrix 52

DISCR--discriminant functions 53

Factor Analysis (See Eigenanalysis)

TRACE--cumulative percentage of eigenvalues 55

LOAD--factor loading 56

VARMX--varimax rotation 56

Time Series (See Smoothing)

AUTO--autocovariances 59

CROSS--cross covariances 60

SMO--application of filter coefficients (weights) 61

EXSMO--triple exponential smoothing 62

Nonparametric Statistics

*KOLMO--Kolmogorov-Smirnov one-sample test 63

*KOLM2--Kolmogorov-Smirnov two-sample test 65

*SMIRN--Kolmogorov-Smirnov limiting distribution values 66

CHISQ--χ^2 test for contingency tables 68

KRANK--Kendall rank correlation 69

*MPAIR--Wilcoxin's signed ranks test 70

QTEST--Cochran Q-test 71

RANK--rank observations 71

*SIGNT--sign test 72

SRANK--Spearman rank correlation 73

TIE--calculation of ties in ranked observations 74

TWOAV--Friedman two-way analysis of variance statistic 74

UTEST--Mann-Whitney U-test 75

WTEST--Kendall coefficient of concordance 76

Generation of Random Variates - Distribution Functions

RANDU--uniform random deviates 77

GAUSS--normal deviates 77

*NDTR--normal distribution function 78

*BDTR--beta distribution function 78

*CDTR--X^2 distribution function 81

*NDTRI--inverse of normal distribution function 83

Elementary Statistics and Miscellany

MOMEN--first four moments 85

TTEST--test on population means 86

*BISER--biserial correlation coefficient 87

*PHI--phi coefficient 88

*POINT--point-biserial correlation coefficient 89

*TETRA--tetrachoric correlation coefficient 90

*SRATE--survival rates 92

Matrices: Storage

MCPY--matrix copy 94

RCPY--copy row of matrix into vector 94

CCPY--copy column of matrix into vector 95

DCPY--copy diagonal of matrix into vector 95

XCPY--copy submatrix from given matrix 96

MSTR--storage conversion 96

LOC--location in compressed-stored matrix 97

CONVT--single-precision/double-precision conversion 97

ARRAY--vector storage/double-dimensioned storage conversion 98

Matrices: Operations

GMADD--add two general matrices 98

GMSUB--subtract two general matrices 99

GMPRD--product of two general matrices 99

GMTRA--transpose of a general matrix 100

GTPRD--transpose product of two general matrices 100

MADD--add two matrices 101

MSUB--subtract two matrices 101

MPRD--matrix product (row into column) 102

MTRA--transpose a matrix 102

TPRD--transpose product 103

MATA--transpose product of matrix by itself 103

SADD--add scalar to matrix 104

SSUB--subtract scalar from a matrix 104

SMPY--matrix multiplied by a scalar 105

SDIV--matrix divided by a scalar 105

SCLA--matrix clear and add scalar 106

DCLA--replace diagonal with scalar 106

RADD--add row of one matrix to row of another matrix 107

CADD--add column of one matrix to column of another matrix 107

SRMA--scalar multiply row and add to another row 108

SCMA--scalar multiply column and add to another column 108

RINT--interchange two rows 109

CINT--interchange two columns 109

RSUM--sum the rows of a matrix 110

CSUM--sum the columns of a matrix 110

RTAB--tabulate the rows of a matrix 111

CTAB--tabulate the columns of a matrix 112

RSRT--sort matrix rows 112

CSRT--sort matrix columns 113

RCUT--partition by row 113

CCUT--partition by column 114

RTIE--adjoin two matrices by row 114

CTIE--adjoin two matrices by column 115

*MPRC, DMPRC--permute rows or columns 115

MFUN--matrix transformation by a function 117

RECP--reciprocal function for MFUN 117

Matrices: Inversion, Systems of Linear Equations and Related Topics

MINV--matrix inversion 118

*SINV, DSINV--invert a symmetric positive definite matrix 119

SIMQ--solution of simultaneous linear, algebraic equations 120

GELG, DGELG--system of general simultaneous linear equations by Gauss elimination 121

*RSLMC--solution of simultaneous linear equations with iterative refinement 124

*FACTR--triangular factorization of a nonsingular matrix 126

MFGR, DMFGR--matrix factorization and rank determination 127

GELS, DGELS--system of general simultaneous linear equations with symmetric coefficients 133

GELB, DGELB--system of general simultaneous linear equations with band-structured coefficients 137

*MTDS, DMTDS--divide a matrix by a triangular matrix 142

*MLSS, DMLSS--solution of simultaneous linear equations with symmetric positive semidefinite matrix 145

*MCHB, DMCHB--triangular factorization of a symmetric positive definite band matrix 148

*MFSS, DMFSS--triangular factorization and rank determination of a symmetric positive semidefinite matrix 152

*MFSD, DMFSD--triangular factorization of a symmetric positive definite matrix 158

LLSQ, DLLSQ--solution of linear least-squares problems 160

Matrices: Eigenanalysis and Related Topics

EIGEN--eigenvalues and eigenvectors of a real, symmetric matrix 164

NROOT--eigenvalues and eigenvectors of a special nonsymmetric matrix 166

*ATEIG--eigenvalues of a real almost triangular matrix 167

*HSBG--reduction of a real matrix to almost triangular form 169

Polynomials: Operations

PADD--add two polynomials 171

PSUB--subtract one polynomial from another 171

PMPY--multiply two polynomials 172

PDIV--divide one polynomial by another 172

PCLA--replace one polynomial by another 173

PADDM--multiply polynomial by constant and add to another polynomial 173

PVAL--value of a polynomial 174

PVSUB--substitute variable of polynomial by another polynomial 174

PILD--evaluate polynomial and its first derivative 175

PDER--derivative of a polymonial 175

PINT--integral of a polynomial 176

PQSD--quadratic synthetic division of a polynomial 176

PCLD--complete linear synthetic division 177

PGCD--greatest common divisor of two polynomials 177

PNORM--normalize coefficient vector of polynomial 178

PECN, DPECN--economization of a polynomial for symmetric range 178

PECS, DPECS--economization of a polynomial for unsymmetric range 180

Polynomials: Roots

POLRT--real and complex roots of a real polynomial 181

PRQD, DPRQD--roots of a real polynomial by QD algorithm with displacement 183

*PRBM, DPRBM--roots of a real polynomial by Bairstow's algorithm 189

*PQFB, DPQFB--determine a quadratic factor of a real polynomial 193

Polynomials: Special Types

CNP, DCNP--value of Nth Chebyshev polynomial 198

CNPS, DCNPS--value of series expansion in Chebyshev polynomials 199

TCNP, DTCNP--transform series expansion in Chebyshev polynomials to a polynomial 200

CSP, DCSP--value of Nth shifted Chebyshev polynomial 201

CSPS, DCSPS--value of series expansion in shifted Chebyshev polynomials 202

TCSP, DTCSP--transform series expansion in shifted Chebyshev polynomials to a polynomial 203

HEP, DHEP--value of Hermite polynomial 205

HEPS, DHEPS--value of series expansion in Hermite polynomials 206

THEP, DTHEP--transform series expansion in Hermite polynomials to a polynomial 207

LAP, DLAP--value of Laguerre polynomial 208

LAPS, DLAPS--value of series expansion in Laguerre polynomials 209

TLAP, DTLAP--transform series expansion in Laguerre polynomials to a polynomial 210

LEP, DLEP--value of Legendre polynomial 212

LEPS, DLEPS--value of series expansion in Legendre polynomials 213

TLEP, DTLEP--transform a series expansion in Legendre polynomials to a polynomial 214

Roots of Nonlinear Equations

RTWI, DRTWI--refine estimate of root by Wegstein's iteration 215

RTMI, DRTMI--determine root within a range by Mueller's iteration 217

RTNI, DRTNI--refine estimate of root by Newton's iteration 220

Extremum of Functions

FMFP, DFMFP--unconstrained minimum of a function of several variables--Davidon method 221

FMCG, DFMCG--unconstrained minimum of a function of several variables--conjugate gradiant method 225

Permutations

*PPRCN--composition of permutations 231

*PERM--operations with permutations and transpositions 232

Sequences: Sums and Limits

TEAS, DTEAS--limit of a given sequence 234

TEUL, DTEUL--sum of a given function sequence 238

Interpolation, Approximation, and Smoothing

ALI, DALI--Aitken-Lagrange interpolation 241

AHI, DAHI--Aitken-Hermite interpolation 243

ACFI, DACFI--continued fraction interpolation 245

ATSG, DATSG--table selection out of a general table 248

ATSM, DATSM--table selection out of a monotonic table 250

ATSE, DATSE--table selection out of an equidistant table 251

*SG13, DSG13--local least-squares smoothing of tabulated functions 253

*SE13, DSE13
*SE15, DSE15
*SE35, DSE35--local least-squares smoothing of equidistantly tabulated functions 255

*APFS, DAPFS--solve normal equations for least-squares fit 260

*APCH, DAPCH--least-squares polynomial approximation 263

*ARAT, DARAT
*FRAT, DFRAT--rational least-squares approximation 265

*APLL, DAPLL--linear least-squares approximation 271

FORIF--Fourier analysis of a given function 274

FORIT--Fourier analysis of a tabulated function 275

HARM, DHARM--complex three-dimensional Fourier analysis 276

RHARM, DRHARM--real one-dimensional Fourier analysis 281

*APMM, DAPMM--linear Chebyshev approximation over a discrete range 283

Numerical Quadrature

QTFG, DQTFG--integration of monotonically tabulated function by trapezoidal rule 289

QTFE, DQTFE--integration of equidistantly tabulated function by trapezoidal rule 290

QSF, DQSF--integration of equidistantly tabulated function by Simpson's rule 291

QHFG, DQHFG--integration of monotonically tabulated function with first derivative by Hermitian formula of first order 293

QHFE, DQHFE--integration of equidistantly tabulated function with first derivative by Hermitian formula of first order 294

QHSG, DQHSG--integration of monotonically tabulated function with first and second derivatives by Hermitian formula of first order 295

QHSE, DQHSE--integration of equidistantly tabulated function with first and second derivatives by Hermitian formula of second order 296

QATR, DQATR--integration of a given function by trapezoidal rule together with Romberg's extrapolation method 297

QG2-QG10, DQG4-DQG32--integration of a given function by Gaussian quadrature formulas 299

QL2-QL10, DQL4-DQL32--integration of a given function by Gaussian-Laguerre quadrature formulas 303

QH2-QH10, DQH8-DQH64--integration of a given function by Gaussian-Hermite quadrature formulas 308

QA2-QA10, DQA4-DQA32--integration of a given function by associated Gaussian-Laguerre quadrature formulas 314

Numerical Differentiation

*DGT3, DDGT3--differentiation of a tabulated function by parabolic interpolation 319

*DET3, DDET3
*DET5, DDET5--differentiation of an equidistantly tabulated function 320

*DCAR, DDCAR--derivative of a function at the center of an interval 324

*DBAR, DDBAR--derivative of a function at the border of an interval 327

Ordinary Differential Equations

RK1--solution of first-order differential equation by Runge-Kutta method 331

RK2--tabulated solution of first-order differential equation by Runge-Kutta method 332

RKGS, DRKGS--solution of system of first-order ordinary differential equations with given initial values by the Runge-Kutta method 333

HPCG, DHPCG--solution of general system of first-order ordinary differential equations with given initial values by Hamming's modified predictor-corrector method 337

HPCL, DHPCL--solution of linear system of first-order ordinary differential equations with given initial values by Hamming's modified predictor-corrector method 343

LBVP, DLBVP--solution of system of linear first-order ordinary differential equations with linear boundary conditions by method of adjoint equations 350

Special Functions

GMMMA--gamma function 361

*DLGAM--log of gamma function 362

BESJ--J Bessel function 363

BESY--Y Bessel function 364

I0--I Bessel function, I_0 365

INUE--I_n Bessel function 366

BESK--K Bessel function 366

EXPI--exponential integral 368

SICI--sine cosine integral 370

CS--Fresnel integrals 372

CEL1, DCEL1--complete elliptic integral of the first kind 374

CEL2, DCEL2--complete elliptic integral of the second kind 376

ELI1, DELI1--generalized elliptic integral of the first kind 378

ELI2, DELI2--generalized elliptic integral of the second kind 380

JELF, DJELF--Jacobian elliptic functions 382

GUIDE TO SAMPLE PROGRAMS

Data Screening

DASCR--Sample Main Program 400

Illustrates use of:

SUBST--subset selection from observation matrix

TAB1--tabulation of data (one variable)

LOC--location in compressed-stored matrix

Special sample subroutines are:

BOOL--Boolean expression

HIST--histogram printing

MATIN--matrix input

Multiple Linear Regression

REGRE--Sample Main Program 404

Illustrates use of:

CORRE--means, standard deviations, and correlations

ORDER--rearrangement of intercorrelations

MINV--matrix inversion

MULTR--multiple regression

Special sample subroutine is:

DATA--sample data read

Polynomial Regression

POLRG--Sample Main Program 408

Illustrates use of:

GDATA--data generation

ORDER--rearrangement of intercorrelations

MINV--matrix inversion

MULTR--multiple regression

Special sample subroutine is:

PLOT--output plot

*Stepwise Multiple Regression

*STEPR--Sample Main Program 413

Illustrates use of:

CORRE--means, standard deviations, and correlations

MSTR--storage conversion

*STPRG--stepwise multiple regression

LOC--location in compressed-stored matrix

Special sample subroutines used are:

*STOUT--sample stepwise regression output subroutine

DATA--sample data read subroutine

Canonical Correlation

MCANO--Sample Main Program 418

Illustrates use of:

CORRE--means, standard deviations, and correlations

CANOR--canonical correlation

MINV--matrix inversion

NROOT--eigenvalues and eigenvectors of a special, nonsymmetric matrix

EIGEN--eigenvalues and eigenvectors of a symmetric matrix

Special sample subroutine is:

DATA--sample data read

Analysis of Variance

ANOVA--Sample Main Program 422

Illustrates use of:

AVDAT--data storage allocation

AVCAL--Σ and Δ operations

MEANQ--mean square operation

Discriminant Analysis

MDISC--Sample Main Program 425

Illustrates use of:

DMATX--means and dispersion matrix

MINV--matrix inversion

DISCR--discriminant functions

Factor Analysis

FACTO--Sample Main Program 429

Illustrates use of:

CORRE--means, standard deviations, and correlations

EIGEN--eigenvalues and eigenvectors of a real, symmetric matrix

TRACE--cumulative percentage of eigenvalues

LOAD--factor loading

VARMX--varimax rotation

Special sample subroutine is:

DATA--sample data read

*Kolmogorov-Smirnov Test:

*KOLM--Sample Main Program 433

Illustrates use of:

*KOLMO--one-sample test

*KOLM2--two-sample test

*SMIRN--Kolmogorov-Smirnov limiting distribution function

*NDTR--normal distribution function

Triple Exponential Smoothing

EXPON--Sample Main Program 439

Illustrates use of:

EXSMO--triple exponential smoothing

Matrix Addition

ADSAM--Sample Main Program 441

Illustrates use of:

MADD--matrix add

LOC--location in compressed-stored matrix

Special sample subroutines are:

MATIN--matrix input

MXOUT--matrix output

Numerical Quadrature Integration

QDINT--Sample Main Program 443

Illustrates use of:

QSF--numerical quadrature integration (Simpson's rule)

Runge-Kutta Integration

RKINT--Sample Main Program 445

Illustrates use of:

RK2--Runge-Kutta integration

Special sample function is:

FUN--definition of differential equation

Polynomial Roots

SMPRT--Sample Main Program 447

Illustrates use of:

POLRT--real and complex roots of polynomial

Solution of Simultaneous Equations

SOLN--Sample Main Program 449

Illustrates use of:

SIMQ--solution of simultaneous equations

LOC--location in compressed-stored matrix

Special sample subroutines are:

MATIN--matrix input

MXOUT--matrix output

Special Sample Subroutines

BOOL--Boolean expression 452

DATA--sample data read 452

FUN--definition of differential equation 452

HIST--histogram printing 452

PLOT--output plot 452

MATIN--matrix input 453

MXOUT--matrix output 454

*STOUT--stepwise regression output 454

ALPHABETIC GUIDE TO SUBROUTINES WITH STORAGE REQUIREMENTS

The following table lists the number of characters of storage required by each of the subroutines in the Scientific Subroutine Package. The figures given were obtained by using Basic Programming Support FORTRAN (Tape), Version 3, Level 0. The use of other FORTRAN compilers on System/360 may cause deviations from these figures.

The double-precision version of the subroutines in the Scientific Subroutine Package are listed immediately following their alphabetized single-precision counterparts. Main line routines are asterisked.

Name	(Label)	Storage Required (Bytes)	Page
ABSNT	(ABSN)	428	28
ACFI	(ACFI)	1214	245
DACFI	(DACF)	1286	245
*ADSAM	(ADSA)	13244	443
AHI	(AHI)	1084	243
DAHI	(DAHI)	1116	243
ALI	(ALI)	760	241
DALI	(DALI)	784	241
*ANOVA	(ANOV)	14624	425
APCH	(APCH)	1492	263
DAPCH	(DAPC)	1548	263
APFS	(APFS)	1602	260
DAPFS	(DAPF)	1626	260
APLL	(APLL)	836	271
DAPLL	(DAPL)	860	271
APMM	(APMM)	4694	283
DAPMM	(DAPM)	4728	283
ARAT	(ARAT)	3004	265
DARAT	(DARA)	3090	265
ARRAY	(ARRA)	670	98
ATEIG	(ATEI)	4538	167

Name	(Label)	Storage Required (Bytes)	Page
ATSE	(ATSE)	1022	251
DATSE	(DTSE)	1050	251
ATSG	(ATSG)	852	248
DATSG	(DTSG)	892	248
ATSM	(ATSM)	1124	250
DATSM	(DTSM)	1148	250
AUTO	(AUTO)	624	59
AVCAL	(AVCA)	802	50
AVDAT	(AVDA)	926	49
BDTR	(BDTR)	3354	78
BESJ	(BESJ)	1466	363
BESK	(BESK)	2022	366
BESY	(BESY)	1968	364
BISER	(BISE)	1182	87
BOOL	(BOOL)	164	452
BOUND	(BOUN)	784	27
CADD	(CADD)	550	107
CANOR	(CANO)	2836	47
CCPY	(CCPY)	500	95
CCUT	(CCUT)	716	114
CDTR	(CDTR)	3390	81
CEL1	(CEL1)	544	374
DCEL1	(DCE1)	584	374
CEL2	(CEL2)	718	376
DCEL2	(DCE2)	774	376
CHISQ	(CHIS)	1220	68

Name	(Label)	Storage Required (Bytes)	Page
CINT	(CINT)	364	109
CNP	(CNP)	340	198
DCNP	(DNP)	364	198
CNPS	(CNPS)	442	199
DCNPS	(DNPS)	482	199
CONVT	(CONV)	538	97
CORRE	(CORR)	2950	32
CROSS	(CROS)	750	60
| CS	(CS)	906	372
CSP	(CSP)	352	201
DCSP	(DSP)	376	201
CSPS	(CSPS)	458	202
DCSPS	(DSPS)	506	202
CSRT	(CSRT)	978	113
CSUM	(CSUM)	502	110
| CTAB	(CTAB)	826	112
CTIE	(CTIE)	746	115
*DASCR	(DASC)	6848	403
DATA	(DATA)	308	452
DBAR	(DBAR)	1252	327
DDBAR	(DDBA)	1354	327
DCAR	(DCAR)	1308	324
DDCAR	(DDCA)	1392	324
DCLA	(DCLA)	392	106
DCPY	(DCPY)	394	95
DET3	(DET3)	554	320
DDET3	(DDT3)	594	320

Name	(Label)	Storage Required (Bytes)	Page
DET5	(DET5)	752	322
DDET5	(DDT5)	832	322
DGT3	(DGT3)	738	319
DDGT3	(DDGT)	770	319
DISCR	(DISC)	2532	53
DLGAM	(DLGA)	826	362
DMATX	(DMAT)	1170	52
EIGEN	(EIGE)	2556	164
ELI1	(ELI1)	836	378
DELI1	(DEL1)	900	378
ELI2	(ELI2)	1254	380
DELI2	(DEL2)	1356	380
EXPI	(EXPI)	1032	368
*EXPON	(EXPO)	8952	440
EXSMO	(EXSM)	620	62
*FACTO	(FCTO)	11212	432
FACTR	(FCTR)	1586	126
FMCG	(FMCG)	2820	225
DFMCG	(DFMC)	2956	225
FMFP	(FMFP)	3580	221
DFMFP	(DFMF)	3702	221
FORIF	(FRIF)	990	274
FORIT	(FRIT)	946	275
FRAT	(FRAT)	912	269
DFRAT	(DFRA)	944	271
FUN	(FUN)	186	452
GAUSS	(GAUS)	430	77

Name	(Label)	Storage Required (Bytes)	Page
GDATA	(GDAT)	1764	39
GELB	(GELB)	2640	137
DGELB	(DELB)	2674	137
GELG	(GELG)	1882	121
DGELG	(DELG)	1920	121
GELS	(GELS)	1990	133
DGELS	(DELS)	2028	133
GMADD	(GMAD)	288	98
GMMMA	(GMMM)	826	361
GMPRD	(GMPR)	570	99
GMSUB	(GMSU)	288	99
GMTRA	(GMTR)	358	100
GTPRD	(GTPR)	566	100
HARM	(HARM)	5988	276
DHARM	(DHAR)	6070	276
HEP	(HEP)	398	205
DHEP	(DHP)	422	205
HEPS	(HEPS)	454	206
DHEPS	(DHPS)	486	206
HIST	(HIST)	1778	452
HPCG	(HPCG)	5560	337
DHPCG	(DHCG)	5744	337
HPCL	(HPCL)	5606	343
DHPCL	(DHCL)	5798	343
HSBG	(HSBG)	1440	169
I0	(I0)	546	365
INUE	(INUE)	768	366
JELF	(JELF)	1254	382
DJELF	(DJEL)	1416	382

Name	(Label)	Storage Required (Bytes)	Page	Name	(Label)	Storage Required (Bytes)	Page
*KOLM	(KOLM)	7952	438	MFGR	(MFGR)	2092	127
KOLMO	(KLMO)	1704	63	DMFGR	(DMGR)	2120	127
KOLM2	(KLM2)	1514	65	MFSD	(MFSD)	904	158
KRANK	(KRAN)	1598	69	DMFSD	(DMSD)	906	158
LAP	(LAP)	408	208	MFSS	(MFSS)	2688	152
DLAP	(DAP)	432	208	DMFSS	(DMSS)	2664	152
LAPS	(LAPS)	472	209	MFUN	(MFUN)	446	117
DLAPS	(DAPS)	512	209	MINV	(MINV)	1874	118
LBVP	(LBVP)	8000	350	MISR	(MISR)	2856	33
DLBVP	(DLBV)	8256	350	MLSS	(MLSS)	2020	145
LEP	(LEP)	400	212	DMLSS	(DMLS)	2004	145
DLEP	(DEP)	424	212	MOMEN	(MOME)	1288	85
LEPS	(LEPS)	464	213	MPAIR	(MPAI)	1380	70
DLEPS	(DEPS)	496	213	MPRC	(MPRC)	810	115
LLSQ	(LLSQ)	2938	160	DMPRC	(DMPR)	822	115
DLLSQ	(DLLS)	2986	160	MPRD	(MPRD)	944	102
LOAD	(LOAD)	490	56	MSTR	(MSTR)	590	96
LOC	(LOC)	492	97	MSUB	(MSUB)	942	101
MADD	(MADD)	942	101	MTDS	(MTDS)	1436	142
MATA	(MATA)	790	103	DMTDS	(DMTD)	1428	142
MATIN	(MATI)	1180	453	MTRA	(MTRA)	552	102
*MCANO	(MCAN)	8196	421	MULTR	(MULT)	1356	37
MCHB	(MCHB)	2286	148	MXOUT	(MXOU)	1440	454
DMCHB	(DMCH)	2270	148	NDTR	(NDTR)	488	78
MCPY	(MCPY)	410	94	NDTRI	(NTRI)	758	83
*MDISC	(MDIS)	15368	428	NROOT	(NROO)	1940	166
MEANQ	(MEAN)	1708	51	ORDER	(ORDE)	682	36

Name	(Label)	Storage Required (Bytes)	Page	Name	(Label)	Storage Required (Bytes)	Page
PADD	(PADD)	474	171	PROBT	(PROB)	2466	44
PADDM	(PDDM)	510	173	PRQD	(PRQD)	4572	183
PCLA	(PCLA)	282	173	DPRQD	(DPRQ)	4636	183
PCLD	(PCLD)	314	177	PSUB	(PSUB)	476	171
PDER	(PDER)	350	175	PVAL	(PVAL)	302	174
PDIV	(PDIV)	794	172	PVSUB	(PVSU)	714	174
PECN	(PECN)	848	178	QATR	(QATR)	1128	297
DPECN	(DPCN)	862	178	DQATR	(DQAT)	1190	297
PECS	(PECS)	746	180	QA2	(QA2)	330	314
DPECS	(DPCS)	766	180	QA3	(QA3)	376	314
PERM	(PERM)	1024	232	QA4	(QA4)	422	314
PGCD	(PGCD)	566	177	DQA4	(DQA4)	462	315
PHI	(PHI)	1390	88	QA5	(QA5)	468	315
PILD	(PILD)	402	175	QA6	(QA6)	514	315
PINT	(PINT)	344	176	QA7	(QA7)	560	315
PLOT	(PLOT)	2500	452	QA8	(QA8)	606	316
PMPY	(PMPY)	506	172	DQA8	(DQA8)	678	316
PNORM	(PNOR)	258	178	QA9	(QA9)	652	316
POINT	(POIN)	1144	89	QA10	(QA10)	698	316
*POLRG	(PLRG)	9282	411	DQA12	(DA12)	894	317
POLRT	(PLRT)	2086	181	DQA16	(DA16)	1110	317
PPRCN	(PPRC)	704	231	DQA24	(DA24)	1542	318
PQFB	(PQFB)	2252	193	DQA32	(DA32)	1974	318
DPQFB	(DPQF)	2384	193	*QDINT	(QDIN)	2916	445
PQSD	(PQSD)	382	176	QG2	(QG2)	442	299
PRBM	(PRBM)	2400	189	QG3	(QG3)	468	299
DPRBM	(DPRB)	2476	189	QG4	(QG4)	538	300

Name	(Label)	Storage Required (Bytes)	Page	Name	(Label)	Storage Required (Bytes)	Page
DQG4	(DQG4)	598	300	DQH8	(DQH8)	622	310
QG5	(QG5)	576	300	QH9	(QH9)	616	310
QG6	(QG6)	638	300	QH10	(QH10)	652	310
QG7	(QG7)	676	301	DQH16	(DH16)	982	311
QG8	(QG8)	738	301	DQH24	(DH24)	1342	311
DQG8	(DQG8)	814	301	DQH32	(DH32)	1702	312
QG9	(QG9)	776	301	DQH48	(DH48)	2422	312
QG10	(QG10)	838	302	DQH64	(DH64)	3142	313
DQG12	(DG12)	1030	302	QL2	(QL2)	330	303
DQG16	(DG16)	1246	302	QL3	(QL3)	376	304
DQG24	(DG24)	1678	302	QL4	(QL4)	422	304
DQG32	(DG32)	2110	303	DQL4	(DQL4)	462	304
QHFE	(QHFE)	434	294	QL5	(QL5)	468	304
DQHFE	(DQHE)	482	294	QL6	(QL6)	514	305
QHFG	(QHFG)	414	293	QL7	(QL7)	560	305
DQHFG	(DQHG)	454	293	QL8	(QL8)	606	305
QHSE	(QHSE)	478	296	DQL8	(DQL8)	678	305
DQHSE	(DQHS)	542	296	QL9	(QL9)	652	306
QHSG	(QHSG)	462	295	QL10	(QL10)	698	306
DQHSG	(DHSG)	514	295	DQL12	(DL12)	894	306
QH2	(QH2)	324	308	DQL16	(DL16)	1110	306
QH3	(QH3)	370	308	DQL24	(DL24)	1542	307
QH4	(QH4)	406	309	DQL32	(DL32)	1974	307
QH5	(QH5)	452	309	QSF	(QSF)	1286	291
QH6	(QH6)	488	309	DQSF	(DQSF)	1358	291
QH7	(QH7)	534	309	QTEST	(QTES)	710	71
QH8	(QH8)	570	310	QTFE	(QTFE)	382	290

Name	(Label)	Storage Required (Bytes)	Page	Name	(Label)	Storage Required (Bytes)	Page
DQTFE	(DTFE)	422	290	RTWI	(RTWI)	770	215
QTFG	(QTFG)	370	289	DRTWI	(DRTW)	804	215
DQTFG	(DTFG)	402	289	SADD	(SADD)	434	104
RADD	(RADD)	538	107	SCLA	(SCLA)	420	106
RANDU	(RAND)	310	77	SCMA	(SCMA)	430	108
RANK	(RANK)	672	71	SDIV	(SDIV)	470	105
RCPY	(RCPY)	500	94	SE13	(SE13)	460	255
RCUT	(RCUT)	716	113	DSE13	(DE13)	500	255
RECP	(RECP)	224	117	SE15	(SE15)	588	256
*REGRE	(REGR)	13736	407	DSE15	(DE15)	620	256
RHARM	(RHAR)	1660	281	SE35	(SE35)	564	258
DRHARM	(DRAR)	1762	281	DSE35	(DSE3)	604	258
RINT	(RINT)	376	109	SG13	(SG13)	648	253
RKGS	(RKGS)	2878	333	DSG13	(DSG1)	696	253
DRKGS	(DRKG)	3018	333	SICI	(SICI)	1192	370
*RKINT	(RKIN)	2704	446	SIGNT	(SIGN)	1170	72
| RK1	(RK1)	1420	331	SIMQ	(SIMQ)	1286	120
RK2	(RK2)	770	332	SINV	(SINV)	1092	119
RSLMC	(RSLM)	1830	124	DSINV	(DSIN)	1080	119
RSRT	(RSRT)	1034	112	SMIRN	(SMIR)	700	66
RSUM	(RSUM)	502	110	SMO	(SMO)	542	61
| RTAB	(RTAB)	826	111	*SMPRT	(SMPR)	1792	449
RTIE	(RTIE)	778	114	SMPY	(SMPY)	434	105
RTMI	(RTMI)	1510	217	*SOLN	(SOLN)	11754	451
DRTMI	(DRTM)	1582	217	SRANK	(SRAN)	1260	73
RTNI	(RTNI)	724	220	SRATE	(SRAT)	1534	92
DRTNI	(DRTN)	770	220	SRMA	(SRMA)	442	108

Name	(Label)	Storage Required (Bytes)	Page	Name	(Label)	Storage Required (Bytes)	Page
SSUB	(SSUB)	434	104	DTEUL	(DTEU)	924	238
*STEPR	(STEP)	11584	417	THEP	(THEP)	712	207
STOUT	(STOU)	1684	454	DTHEP	(DTHE)	768	207
STPRG	(STPR)	3200	41	TIE	(TIE)	640	74
SUBMX	(SUBM)	512	31	TLAP	(TLAP)	776	210
SUBST	(SUBS)	888	28	DTLAP	(DTLA)	820	210
TAB1	(TAB1)	1636	29	TLEP	(TLEP)	716	214
TAB2	(TAB2)	2806	30	DTLEP	(DTLE)	772	214
TALLY	(TALL)	1206	27	TPRD	(TPRD)	904	103
TCNP	(TCNP)	678	200	TRACE	(TRAC)	560	55
DTCNP	(DTCN)	726	200	TTEST	(TTES)	1794	86
TCSP	(TCSP)	706	203	TWOAV	(TWOA)	1108	74
DTCSP	(DTCS)	754	203	UTEST	(UTES)	1064	75
TEAS	(TEAS)	2024	234	VARMX	(VARM)	2914	56
DTEAS	(DTEA)	2096	234	WTEST	(WTES)	1488	76
TETRA	(TETR)	2038	90	XCPY	(XCPY)	670	96
TEUL	(TEUL)	844	238				

SUBROUTINE DESCRIPTIONS AND LISTINGS

Listed below are the subroutines. A brief description precedes those subroutines that require further clarification.

STATISTICS

Data Screening

Subroutine TALLY

```
C                                                                       TALL  10
C     ..................................................................TALL  20
C                                                                       TALL  30
C        SUBROUTINE TALLY                                               TALL  40
C                                                                       TALL  50
C        PURPOSE                                                        TALL  60
C           CALCULATE TOTAL, MEAN, STANDARD DEVIATION, MINIMUM, MAXIMUM TALL  70
C           FOR EACH VARIABLE IN A SET (OR A SUBSET) OF OBSERVATIONS    TALL  80
C                                                                       TALL  90
C        USAGE                                                          TALL 100
C           CALL TALLY(A,S,TCTAL,AVER,SD,VMIN,VMAX,NO,NV,IER)           TALL 110
C                                                                       TALL 120
C        DESCRIPTION OF PARAMETERS                                      TALL 130
C           A     - OBSERVATION MATRIX, NO BY NV                        TALL 140
C           S     - INPUT VECTOR INDICATING SUBSET OF A. ONLY THOSE     TALL 150
C                   OBSERVATIONS WITH A NON-ZERO S(J) ARE CONSIDERED.   TALL 160
C                   VECTOR LENGTH IS NO.                                TALL 170
C           TCTAL - OUTPUT VECTOR OF TOTALS OF EACH VARIABLE. VECTOR    TALL 180
C                   LENGTH IS NV.                                       TALL 190
C           AVER  - OUTPUT VECTOR OF AVERAGES OF EACH VARIABLE. VECTOR  TALL 200
C                   LENGTH IS NV.                                       TALL 210
C           SD    - OUTPUT VECTOR OF STANDARD DEVIATIONS OF EACH        TALL 220
C                   VARIABLE. VECTOR LENGTH IS NV.                      TALL 230
C           VMIN  - OUTPUT VECTOR OF MINIMA OF EACH VARIABLE. VECTOR    TALL 240
C                   LENGTH IS NV.                                       TALL 250
C           VMAX  - OUTPUT VECTOR OF MAXIMA OF EACH VARIABLE. VECTOR    TALL 260
C                   LENGTH IS NV.                                       TALL 270
C           NO    - NUMBER OF OBSERVATIONS                              TALL 280
C           NV    - NUMBER OF VARIABLES FOR EACH OBSERVATION            TALL 290
C           IER   - ZERO, IF NO ERROR.                                  TALL 291
C                 - 1, IF S IS NULL.  VMIN=-1.E75, VMAX=SD=AVER=1.E75.  TALL 292
C                 - 2, IF S HAS ONLY ONE NON-ZERO ELEMENT. VMIN=VMAX.   TALL 293
C                   SD=0.0                                              TALL 294
C                                                                       TALL 300
C        REMARKS                                                        TALL 310
C           NONE                                                        TALL 320
C                                                                       TALL 330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TALL 340
C           NONE                                                        TALL 350
C                                                                       TALL 360
C        METHOD                                                         TALL 370
C           ALL OBSERVATIONS CORRESPONDING TO A NON-ZERO ELEMENT IN S   TALL 380
C           VECTOR ARE ANALYZED FOR EACH VARIABLE IN MATRIX A.          TALL 390
C           TOTALS ARE ACCUMULATED AND MINIMUM AND MAXIMUM VALUES ARE   TALL 400
C           FOUND. FOLLOWING THIS, MEANS AND STANDARD DEVIATIONS ARE    TALL 410
C           CALCULATED.  THE DIVISOR FOR STANDARD DEVIATION IS ONE LESS TALL 420
C           THAN THE NUMBER OF OBSERVATIONS USED.                       TALL 430
C                                                                       TALL 440
C     ..................................................................TALL 450
C                                                                       TALL 460
      SUBROUTINE TALLY(A,S,TCTAL,AVER,SD,VMIN,VMAX,NO,NV)               TALL 470
      DIMENSION A(1),S(1),TOTAL(1),AVER(1),SD(1),VMIN(1),VMAX(1)        TALL 480
C                                                                       TALL 490
C        CLEAR OUTPUT VECTORS AND INITIALIZE VMIN,VMAX                  TALL 500
C                                                                       TALL 510
      IER=0                                                             TALL 511
      DO 1 K=1,NV                                                       TALL 520
      TOTAL(K)=0.0                                                      TALL 530
      AVER(K)=1.0E75                                                    TALL 540
      SD(K)=1.0E75                                                      TALL 550
      VMIN(K)=-1.0E75                                                   TALL 560
    1 VMAX(K)=1.0E75                                                    TALL 570
C                                                                       TALL 580
C        TEST SUBSET VECTOR                                             TALL 590
C                                                                       TALL 600
      SCNT=0.0                                                          TALL 610
      DO 7 J=1,NO                                                       TALL 620
      IJ=J-NO                                                           TALL 630
      IF(S(J)) 2,7,2                                                    TALL 640
    2 SCNT=SCNT+1.0                                                     TALL 650
C                                                                       TALL 660
C        CALCULATE TOTAL, MINIMA, MAXIMA                                TALL 670
C                                                                       TALL 680
      DO 6 I=1,NV                                                       TALL 690
      IJ=IJ+NO                                                          TALL 700
      TOTAL(I)=TOTAL(I)+A(IJ)                                           TALL 710
      IF(A(IJ)-VMIN(I)) 3,4,4                                           TALL 720
    3 VMIN(I)=A(IJ)                                                     TALL 730
    4 IF(A(IJ)-VMAX(I)) 6,6,5                                           TALL 740
    5 VMAX(I)=A(IJ)                                                     TALL 750
    6 SD(I)=SD(I)+A(IJ)*A(IJ)                                           TALL 760
    7 CONTINUE                                                          TALL 770
C                                                                       TALL 780
C        CALCULATE MEANS AND STANDARD DEVIATIONS                        TALL 790
C                                                                       TALL 800
      IF (SCNT)8,8,9                                                    TALL 801
    8 IER=1                                                             TALL 802
      GO TO 15                                                          TALL 803
    9 DO 10 I=1,NV                                                      TALL 810
   10 AVER(I)=TOTAL(I)/SCNT                                             TALL 820
      IF (SCNT-1.0) 13,11,13                                            TALL 821
   11 IER=2                                                             TALL 822
      DO 12 I=1,NV                                                      TALL 823
   12 SD(I)=0.0                                                         TALL 824
      GO TO 15                                                          TALL 825
   13 DO 14 I=1,NV                                                      TALL 826
   14 SD(I)=SQRT(ABS((SD(I)-TOTAL(I)*TOTAL(I)/SCNT)/(SCNT-1.0)))        TALL 830
   15 RETURN                                                            TALL 840
      END                                                               TALL 850
```

Subroutine BOUND

```
C                                                                       BOUN  10
C     ..................................................................BOUN  20
C                                                                       BOUN  30
C        SUBROUTINE BOUND                                               BOUN  40
C                                                                       BOUN  50
C        PURPOSE                                                        BOUN  60
C           SELECT FROM A SET (OR A SUBSET) OF OBSERVATIONS THE NUMBER  BOUN  70
C           OF OBSERVATIONS UNDER, BETWEEN AND OVER TWO GIVEN BOUNDS    BOUN  80
C           FOR EACH VARIABLE                                           BOUN  90
C                                                                       BOUN 100
C        USAGE                                                          BOUN 110
C           CALL BOUND (A,S,BLO,BHI,UNDER,BETW,OVER,NO,NV,IER)          BOUN 120
C                                                                       BOUN 130
C        DESCRIPTION OF PARAMETERS                                      BOUN 140
C           A     - OBSERVATION MATRIX, NO BY NV                        BOUN 150
C           S     - VECTOR INDICATING SUBSET OF A. ONLY THOSE           BOUN 160
C                   OBSERVATIONS WITH A NON-ZERO S(J) ARE CONSIDERED.   BOUN 170
C                   VECTOR LENGTH IS NO.                                BOUN 180
C           BLO   - INPUT VECTOR OF LOWER BOUNDS ON ALL VARIABLES.      BOUN 190
C                   VECTOR LENGTH IS NV.                                BOUN 200
C           BHI   - INPUT VECTOR OF UPPER BOUNDS ON ALL VARIABLES.      BOUN 210
C                   VECTOR LENGTH IS NV.                                BOUN 220
C           UNDER - OUTPUT VECTOR INDICATING, FOR EACH VARIABLE, NUMBER BOUN 230
C                   OF OBSERVATIONS UNDER LOWER BOUNDS. VECTOR LENGTH   BOUN 240
C                   IS NV.                                              BOUN 250
C           BETW  - OUTPUT VECTOR INDICATING, FOR EACH VARIABLE, NUMBER BOUN 260
C                   OF OBSERVATIONS EQUAL TO OR BETWEEN LOWER AND UPPER BOUN 270
C                   BOUNDS. VECTOR LENGTH IS NV.                        BOUN 280
C           OVER  - OUTPUT VECTOR INDICATING, FOR EACH VARIABLE, NUMBER BOUN 290
C                   OF OBSERVATIONS OVER UPPER BOUNDS. VECTOR LENGTH    BOUN 300
C                   IS NV.                                              BOUN 310
C           NO    - NUMBER OF OBSERVATIONS                              BOUN 320
C           NV    - NUMBER OF VARIABLES FOR EACH OBSERVATION            BOUN 330
C           IER   - ZERO, IF NO ERROR.                                  BOUN 331
C                 - 1, IF LOWER BOUND IS GREATER THAN THE UPPER BOUND   BOUN 332
C                   FOR SOME VARIABLE                                   BOUN 333
C                                                                       BOUN 340
C        REMARKS                                                        BOUN 350
C           NONE                                                        BOUN 360
C                                                                       BOUN 370
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  BOUN 380
C           NONE                                                        BOUN 390
C                                                                       BOUN 400
C        METHOD                                                         BOUN 410
C           EACH ROW (OBSERVATION) OF MATRIX A WITH CORRESPONDING       BOUN 420
C           NON-ZERO ELEMENT IN S VECTOR IS TESTED. OBSERVATIONS ARE    BOUN 430
C           COMPARED WITH SPECIFIED LOWER AND UPPER VARIABLE BOUNDS AND BOUN 440
C           A COUNT IS KEPT IN VECTORS UNDER, BETWEEN, AND OVER.        BOUN 450
C                                                                       BOUN 460
C     ..................................................................BOUN 470
C                                                                       BOUN 480
      SUBROUTINE BOUND(A,S,BLO,BHI,UNDER,BETW,OVER,NO,NV)               BOUN 490
      DIMENSION A(1),S(1),BLO(1),BHI(1),UNDER(1),BETW(1),OVER(1)        BOUN 500
C                                                                       BOUN 510
C        CLEAR OUTPUT VECTORS.                                          BOUN 520
C                                                                       BOUN 530
      IER=0                                                             BOUN 531
      DO 10 I=1,NV                                                      BOUN 532
      IF (BLO(I)-BHI(I)) 10,10,11                                       BOUN 533
   11 IER=1                                                             BOUN 534
      GO TO 12                                                          BOUN 535
   10 CONTINUE                                                          BOUN 536
      DO 1 K=1,NV                                                       BOUN 540
      UNDER(K)=0.0                                                      BOUN 550
      BETW(K)=0.0                                                       BOUN 560
    1 OVER(K)=0.0                                                       BOUN 570
C                                                                       BOUN 580
C        TEST SUBSET VECTOR                                             BOUN 590
C                                                                       BOUN 600
      DO 8 J=1,NO                                                       BOUN 610
      IJ=J-NO                                                           BOUN 620
      IF(S(J)) 2,8,2                                                    BOUN 630
C                                                                       BOUN 640
C        COMPARE OBSERVATIONS WITH BOUNDS                               BOUN 650
C                                                                       BOUN 660
    2 DO 7 I=1,NV                                                       BOUN 670
      IJ=IJ+NO                                                          BOUN 680
      IF(A(IJ)-BLO(I)) 5,3,3                                            BOUN 690
    3 IF(A(IJ)-BHI(I)) 4,4,6                                            BOUN 700
C                                                                       BOUN 710
C        COUNT                                                          BOUN 720
C                                                                       BOUN 730
    4 BETW(I)=BETW(I)+1.0                                               BOUN 740
      GO TO 7                                                           BOUN 750
    5 UNDER(I)=UNDER(I)+1.0                                             BOUN 760
      GO TO 7                                                           BOUN 770
    6 OVER(I)=OVER(I)+1.0                                               BOUN 780
    7 CONTINUE                                                          BOUN 790
    8 CONTINUE                                                          BOUN 800
   12 RETURN                                                            BOUN 810
      END                                                               BOUN 820
```

Subroutine SUBST

```
C                                                                       SUBS  10
C     ..................................................................SUBS  20
C                                                                       SUBS  30
C        SUBROUTINE SUBST                                               SUBS  40
C                                                                       SUBS  50
C        PURPOSE                                                        SUBS  60
C           DERIVE A SUBSET VECTOR INDICATING WHICH OBSERVATIONS IN A   SUBS  70
C           SET HAVE SATISFIED CERTAIN CONDITIONS ON THE VARIABLES.     SUBS  80
C                                                                       SUBS  90
C        USAGE                                                          SUBS 100
C           CALL SUBST (A,C,R,B,S,NC,NV,NC)                             SUBS 110
C           PARAMETER B MUST BE DEFINED BY AN EXTERNAL STATEMENT IN THE SUBS 120
C           CALLING PROGRAM                                             SUBS 130
C                                                                       SUBS 140
C        DESCRIPTION OF PARAMETERS                                      SUBS 150
C           A  - OBSERVATION MATRIX, NC BY NV                           SUBS 160
C           C  - INPUT MATRIX, 3 BY NC, OF CONDITIONS TO BE CONSIDERED. SUBS 170
C                THE FIRST ELEMENT OF EACH COLUMN OF C REPRESENTS THE   SUBS 180
C                NUMBER OF THE VARIABLE (COLUMN OF THE MATRIX A) TO BE  SUBS 190
C                TESTED, THE SECOND ELEMENT OF EACH COLUMN IS A         SUBS 200
C                RELATIONAL CODE AS FOLLOWS                             SUBS 210
C                      1. FOR LT (LESS THAN)                            SUBS 220
C                      2. FOR LE (LESS THAN OR EQUAL TO)                SUBS 230
C                      3. FOR EQ (EQUAL TO)                             SUBS 240
C                      4. FOR NE (NOT EQUAL TO)                         SUBS 250
C                      5. FOR GE (GREATER THAN OR EQUAL TO)             SUBS 260
C                      6. FOR GT (GREATER THAN)                         SUBS 270
C                THE THIRD ELEMENT OF EACH COLUMN IS A QUANTITY TO BE   SUBS 280
C                USED FOR COMPARISON WITH THE OBSERVATION VALUES. FOR   SUBS 290
C                EXAMPLE, THE FOLLOWING COLUMN IN C                     SUBS 300
C                              2.                                       SUBS 310
C                              5.                                       SUBS 320
C                             92.5                                      SUBS 330
C                CAUSES THE SECOND VARIABLE TO BE TESTED FOR GREATER    SUBS 340
C                THAN OR EQUAL TO 92.5                                  SUBS 350
C           R  - WORKING VECTOR USED TO STORE INTERMEDIATE RESULTS OF   SUBS 360
C                ABOVE TESTS ON A SINGLE OBSERVATION. IF CONDITION IS   SUBS 370
C                SATISFIED, R(I) IS SET TO 1. IF IT IS NOT, R(I) IS SET SUBS 380
C                TO 0. VECTOR LENGTH IS NC.                             SUBS 390
C           E  - NAME OF SUBROUTINE TO BE SUPPLIED BY THE USER. IT      SUBS 400
C                CONSISTS OF A BOOLEAN EXPRESSION LINKING THE           SUBS 410
C                INTERMEDIATE VALUES STORED IN VECTOR R. THE BOOLEAN    SUBS 420
C                OPERATORS ARE '*' FOR 'AND', '+' FOR 'OR'. EXAMPLE     SUBS 430
C                      SUBROUTINE BOOL(R,T)                             SUBS 440
C                      DIMENSION R(3)                                   SUBS 450
C                      T=R(1)*(R(2)+R(3))                               SUBS 460
C                      RETURN                                           SUBS 470
C                      END                                              SUBS 480
C                THE ABOVE EXPRESSION IS TESTED FOR                     SUBS 490
C                      R(1).AND.(R(2).OR.R(3))                          SUBS 500
C           S  - OUTPUT VECTOR INDICATING, FOR EACH OBSERVATION,        SUBS 510
C                WHETHER OR NOT PROPOSITION B IS SATISFIED. IF IT IS,   SUBS 520
C                S(I) IS NON-ZERO. IF IT IS NOT, S(I) IS ZERO. VECTOR   SUBS 530
C                LENGTH IS NO.                                          SUBS 540
C           NC - NUMBER OF OBSERVATIONS. NO MUST BE > OR = TO 1.        SUBS 550
C           NV - NUMBER OF VARIABLES. NV MUST BE > OR = TO 1.           SUBS 560
C           NC - NUMBER OF BASIC CONDITIONS TO BE SATISFIED. NC MUST BE SUBS 570
C                GREATER THAN OR EQUAL TO 1.                            SUBS 571
C                                                                       SUBS 580
C        REMARKS                                                        SUBS 590
C           NONE                                                        SUBS 600
C                                                                       SUBS 610
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SUBS 620
C           B  THE NAME OF ACTUAL SUBROUTINE SUPPLIED BY THE USER MAY   SUBS 630
C              BE DIFFERENT (E.G., BOOL), BUT SUBROUTINE SUBST ALWAYS   SUBS 640
C              CALLS IT AS B.  IN ORDER FOR SUBROUTINE SUBST TO DO THIS,SUBS 650
C              THE NAME OF THE USER-SUPPLIED SUBROUTINE MUST BE         SUBS 660
C              DEFINED BY AN EXTERNAL STATEMENT IN THE CALLING PROGRAM. SUBS 670
C              THE NAME MUST ALSO BE LISTED IN THE ''CALL SUBST''       SUBS 680
C              STATEMENT.  (SEE USAGE ABOVE)                            SUBS 690
C                                                                       SUBS 700
C        METHOD                                                         SUBS 710
C           THE FOLLOWING IS DONE FOR EACH OBSERVATION.                 SUBS 720
C           CONDITION MATRIX IS ANALYZED TO DETERMINE WHICH VARIABLES   SUBS 730
C           ARE TO BE EXAMINED. INTERMEDIATE VECTOR R IS FORMED. THE    SUBS 740
C           BOOLEAN EXPRESSION (IN SUBROUTINE B) IS THEN EVALUATED TO   SUBS 750
C           DERIVE THE ELEMENT IN SUBSET VECTOR S CORRESPONDING TO THE  SUBS 760
C           OBSERVATION.                                                SUBS 770
C                                                                       SUBS 780
C     ..................................................................SUBS 790
      SUBROUTINE SUBST(A,C,R,B,S,NC,NV,NC)                              SUBS 800
      DIMENSION A(1),C(1),R(1),S(1)                                     SUBS 810
C                                                                       SUBS 820
      DO 5 I=1,NC                                                       SUBS 830
      IC=1-NC                                                           SUBS 840
      K=-2                                                              SUBS 850
      DO 8 J=1,NC                                                       SUBS 860
C                                                                       SUBS 870
C        CLEAR R VECTOR                                                 SUBS 880
C                                                                       SUBS 890
      R(J)=0.0                                                          SUBS 900
C                                                                       SUBS 910
C        LOCATE ELEMENT IN OBSERVATION MATRIX AND RELATIONAL CODE       SUBS 920
C                                                                       SUBS 930
      K=K+3                                                             SUBS 940
      IZ=C(K)                                                           SUBS 950
      IA=IC+IZ*NC                                                       SUBS 960
      IGO=C(K+1)                                                        SUBS 970
C                                                                       SUBS 980
C        FORM R VECTOR                                                  SUBS 990
C                                                                       SUBS1000
      G=A(IA)-C(K+2)                                                    SUBS1010
      GO TO(1,2,3,4,5,6),IGO                                            SUBS1020
    1 IF(G) 7,8,8                                                       SUBS1030
    2 IF(G) 7,7,8                                                       SUBS1040
    3 IF(G) 8,7,8                                                       SUBS1050
    4 IF(G) 7,8,7                                                       SUBS1060
    5 IF(G) 8,7,7                                                       SUBS1070
    6 IF(G) 8,8,7                                                       SUBS1080
    7 R(J)=1.0                                                          SUBS1090
    8 CONTINUE                                                          SUBS1100
C                                                                       SUBS1110
C        CALCULATE S VECTOR                                             SUBS1120
C                                                                       SUBS1130
    5 CALL B(R,S(I))                                                    SUBS1140
      RETURN                                                            SUBS1150
      END                                                               SUBS1160
                                                                        SUBS1170
```

Subroutine ABSNT

```
C                                                                       ABSN  10
C     ..................................................................ABSN  20
C                                                                       ABSN  30
C        SUBROUTINE ABSNT                                               ABSN  40
C                                                                       ABSN  50
C        PURPOSE                                                        ABSN  60
C           TEST MISSING OR ZERO VALUES FOR EACH OBSERVATION IN         ABSN  70
C           MATRIX A.                                                   ABSN  80
C                                                                       ABSN  90
C        USAGE                                                          ABSN 100
C           CALL ABSNT (A,S,NC,NV)                                      ABSN 110
C                                                                       ABSN 120
C        DESCRIPTION OF PARAMETERS                                      ABSN 130
C           A  - OBSERVATION MATRIX, NC BY NV                           ABSN 140
C           S  - OUTPUT VECTOR OF LENGTH NO INDICATING THE FOLLOWING    ABSN 150
C                CODES FOR EACH OBSERVATION.                            ABSN 160
C                1  THERE IS NOT A MISSING OR ZERO VALUE.               ABSN 170
C                0  AT LEAST ONE VALUE IS MISSING OR ZERO.              ABSN 180
C           NC - NUMBER OF OBSERVATIONS. NO MUST BE > OR = TO 1.        ABSN 190
C           NV - NUMBER OF VARIABLES FOR EACH OBSERVATION. NV MUST BE   ABSN 200
C                GREATER THAN OR EQUAL TO 1.                            ABSN 201
C                                                                       ABSN 210
C        REMARKS                                                        ABSN 220
C           NONE                                                        ABSN 230
C                                                                       ABSN 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ABSN 250
C           NONE                                                        ABSN 260
C                                                                       ABSN 270
C        METHOD                                                         ABSN 280
C           A TEST IS MADE FOR EACH ROW (OBSERVATION) OF THE MATRIX A.  ABSN 290
C           IF THERE IS NOT A MISSING OR ZERO VALUE, 1 IS PLACED IN     ABSN 300
C           S(J). IF AT LEAST ONE VALUE IS MISSING OR ZERO, 0 IS PLACED ABSN 310
C           IN S(J).                                                    ABSN 320
C                                                                       ABSN 330
C     ..................................................................ABSN 340
      SUBROUTINE ABSNT(A,S,NC,NV)                                       ABSN 350
      DIMENSION A(1),S(1)                                               ABSN 360
C                                                                       ABSN 370
      DO 20 J=1,NC                                                      ABSN 380
      IJ=J-NC                                                           ABSN 390
      S(J)=1.0                                                          ABSN 400
      DO 10 I=1,NV                                                      ABSN 410
      IJ=IJ+NC                                                          ABSN 420
      IF(A(IJ)) 10,5,10                                                 ABSN 430
    5 S(J)=0                                                            ABSN 440
      GO TO 20                                                          ABSN 450
   10 CONTINUE                                                          ABSN 460
   20 CONTINUE                                                          ABSN 470
      RETURN                                                            ABSN 480
      END                                                               ABSN 490
                                                                        ABSN 500
```

Subroutine TAB1

This subroutine tabulates, for a selected variable in an observation matrix, the frequencies and percent frequencies over class intervals. Interval size is computed as follows:

$$k = \frac{UBO_3 - UBO_1}{UBO_2 - 2} \qquad (1)$$

where UBO_1 = given lower bound

UBO_2 = given number of intervals

UBO_3 = given upper bound

If $UBO_1 = UBO_3$, the subroutine finds and uses the minimum and maximum values of the variable.

A table lookup is used to obtain the frequency of the ith class interval for the variable, where i = 1, 2, ..., UBO_2. Then each frequency is divided by the number of observations, n, to obtain the percent frequency:

$$P_i = \frac{100F_i}{n} \qquad (2)$$

In addition, the following statistics are calculated for the variable:

$$\text{Total: } T = \sum_{i=1}^{n} X_{ij} \qquad (3)$$

where j = selected variable

$$\text{Mean: } \overline{X} = \frac{T}{n} \qquad (4)$$

Standard deviation:

$$s = \sqrt{\frac{\sum_{i=1}^{n} X_{ij}^2 - \left(\sum_{i=1}^{n} X_{ij}\right)^2 \Big/ n}{n-1}} \qquad (5)$$

```
C                                                                       TAB1  10
C     ..................................................................TAB1  20
C                                                                       TAB1  30
C        SUBROUTINE TAB1                                                TAB1  40
C                                                                       TAB1  50
C        PURPOSE                                                        TAB1  60
C           TABULATE FOR ONE VARIABLE IN AN OBSERVATION MATRIX (OR A    TAB1  70
C           MATRIX SUBSET), THE FREQUENCY AND PERCENT FREQUENCY OVER    TAB1  80
C           GIVEN CLASS INTERVALS. IN ADDITION, CALCULATE FOR THE SAME  TAB1  90
C           VARIABLE THE TOTAL, AVERAGE, STANDARD DEVIATION, MINIMUM,   TAB1 100
C           AND MAXIMUM.                                                TAB1 110
C                                                                       TAB1 120
C        USAGE                                                          TAB1 130
C           CALL TAB1(A,S,NOVAR,UBC,FREQ,PCT,STATS,NO,NV)               TAB1 140
C                                                                       TAB1 150
C        DESCRIPTION OF PARAMETERS                                      TAB1 160
C           A     - OBSERVATION MATRIX, NO BY NV                        TAB1 170
C           S     - INPUT VECTOR GIVING SUBSET OF A. ONLY THOSE         TAB1 180
C                   OBSERVATIONS WITH A CORRESPONDING NON-ZERO S(J) ARE TAB1 190
C                   CONSIDERED. VECTOR LENGTH IS NO.                    TAB1 200
C           NOVAR - THE VARIABLE TO BE TABULATED. NOVAR MUST BE GREATER TAB1 210
C                   THAN OR EQUAL TO 1 AND LESS THAN OR EQUAL TO NV.    TAB1 220
C                   AND UPPER LIMIT OF VARIABLE TO BE TABULATED         TAB1 230
C                   IN UBC(1), UBC(2) AND UBC(3) RESPECTIVELY. IF       TAB1 240
C                   LOWER LIMIT IS EQUAL TO UPPER LIMIT, THE PROGRAM    TAB1 250
C                   USES THE MINIMUM AND MAXIMUM VALUES OF THE VARIABLE.TAB1 260
C                   NUMBER OF INTERVALS, UBC(2), MUST INCLUDE TWO CELLS TAB1 270
C                   FOR VALUES UNDER AND ABOVE LIMITS. VECTOR LENGTH    TAB1 280
C                   IS 3.                                               TAB1 290
C           FREQ  - OUTPUT VECTOR OF FREQUENCIES. VECTOR LENGTH IS      TAB1 300
C                   UBC(2).                                             TAB1 310
C           PCT   - OUTPUT VECTOR OF RELATIVE FREQUENCIES. VECTOR       TAB1 320
C                   LENGTH IS UBC(2).                                   TAB1 330
C           STATS - OUTPUT VECTOR OF SUMMARY STATISTICS, I.E., TOTAL,   TAB1 340
C                   AVERAGE, STANDARD DEVIATION, MINIMUM AND MAXIMUM.   TAB1 350
C                   VECTOR LENGTH IS 5. IF S IS NULL, THEN TOTAL,AVERAGETAB1 360
C                   AND STANDARD DEVIATION = 0, MIN=1.E75 AND MAX=-1.E75TAB1 361
C           NO    - NUMBER OF OBSERVATIONS. NO MUST BE > OR = TO 1      TAB1 370
C           NV    - NUMBER OF VARIABLES FOR EACH OBSERVATION. NV MUST   TAB1 380
C                   BE GREATER THAN OR EQUAL TO 1.                      TAB1 381
C                                                                       TAB1 390
C        REMARKS                                                        TAB1 400
C           NONE                                                        TAB1 410
C                                                                       TAB1 420
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TAB1 430
C           NONE                                                        TAB1 440
C                                                                       TAB1 450
C        METHOD                                                         TAB1 460
C           THE INTERVAL SIZE IS CALCULATED FROM THE GIVEN INFORMATION  TAB1 470
C           OR OPTIONALLY FROM THE MINIMUM AND MAXIMUM VALUES FOR       TAB1 480
C           VARIABLE NOVAR. THE FREQUENCIES AND PERCENT FREQUENCIES ARE TAB1 490
C           THEN CALCULATED ALONG WITH SUMMARY STATISTICS.              TAB1 500
C           THE DIVISOR FOR STANDARD DEVIATION IS ONE LESS THAN THE     TAB1 510
C           NUMBER OF OBSERVATIONS USED.                                TAB1 520
C                                                                       TAB1 530
C     ..................................................................TAB1 540
C                                                                       TAB1 550
      SUBROUTINE TAB1(A,S,NOVAR,UBC,FREQ,PCT,STATS,NO,NV)               TAB1 560
      DIMENSION A(1),S(1),UBC(1),FREQ(1),PCT(1),STATS(1)                TAB1 570
      DIMENSION NBC(3)                                                  TAB1 580
      DO 5 I=1,3                                                        TAB1 590
    5 NBC(I)=UBC(I)                                                     TAB1 600
C                                                                       TAB1 610
C        CALCULATE MIN AND MAX                                          TAB1 620
C                                                                       TAB1 630
      VMIN=1.0E75                                                       TAB1 640
      VMAX=-1.0E75                                                      TAB1 650
      IJ=NO*(NOVAR-1)                                                   TAB1 660
      DO 30 J=1,NO                                                      TAB1 670
      IJ=IJ+1                                                           TAB1 680
      IF(S(J)) 10,30,10                                                 TAB1 690
   10 IF(A(IJ)-VMIN) 15,20,20                                           TAB1 700
   15 VMIN=A(IJ)                                                        TAB1 710
   20 IF(A(IJ)-VMAX) 30,30,25                                           TAB1 720
   25 VMAX=A(IJ)                                                        TAB1 730
   30 CONTINUE                                                          TAB1 740
      STATS(4)=VMIN                                                     TAB1 750
      STATS(5)=VMAX                                                     TAB1 760
C                                                                       TAB1 770
C        DETERMINE LIMITS                                               TAB1 780
C                                                                       TAB1 790
      IF(UBC(1)-UBC(3)) 40,35,40                                        TAB1 800
   35 LBC(1)=VMIN                                                       TAB1 810
      LBC(3)=VMAX                                                       TAB1 820
   40 INA=LBC(2)                                                        TAB1 830
C                                                                       TAB1 840
C        CLEAR OUTPUT AREAS                                             TAB1 850
C                                                                       TAB1 860
      DO 45 I=1,INA                                                     TAB1 870
      FREQ(I)=0.0                                                       TAB1 880
   45 PCT(I)=0.0                                                        TAB1 890
      DO 50 I=1,3                                                       TAB1 900
   50 STATS(I)=0.0                                                      TAB1 910
C                                                                       TAB1 920
C        CALCULATE INTERVAL SIZE                                        TAB1 930
C                                                                       TAB1 940
      SINT=ABS((UBC(3)-UBC(1))/(UBC(2)-2.0))                            TAB1 950
C                                                                       TAB1 960
C        TEST SUBSET VECTOR                                             TAB1 970
C                                                                       TAB1 980
      SCNT=0.0                                                          TAB1 990
      IJ=NO*(NOVAR-1)                                                   TAB11000
      DO 75 J=1,NO                                                      TAB11010
      IJ=IJ+1                                                           TAB11020
      IF(S(J)) 55,75,55                                                 TAB11030
   55 SCNT=SCNT+1.0                                                     TAB11040
C                                                                       TAB11050
C        DEVELOP TOTAL AND FREQUENCIES                                  TAB11060
C                                                                       TAB11070
      STATS(1)=STATS(1)+A(IJ)                                           TAB11080
      STATS(3)=STATS(3)+A(IJ)*A(IJ)                                     TAB11090
      TEMP=UBC(1)-SINT                                                  TAB11100
      INTX=INA-1                                                        TAB11110
      DO 60 I=1,INTX                                                    TAB11120
      TEMP=TEMP+SINT                                                    TAB11130
      IF(A(IJ)-TEMP) 70,60,60                                           TAB11140
   60 CONTINUE                                                          TAB11150
      IF(A(IJ)-TEMP) 75,65,65                                           TAB11160
   65 FREQ(INA)=FREQ(INA)+1.0                                           TAB11170
      GO TO 75                                                          TAB11180
   70 FREQ(I)=FREQ(I)+1.0                                               TAB11190
   75 CONTINUE                                                          TAB11200
      IF (SCNT)79,100,79                                                TAB11201
C                                                                       TAB11210
C        CALCULATE RELATIVE FREQUENCIES                                 TAB11220
C                                                                       TAB11230
   79 DO 80 I=1,INA                                                     TAB11240
   80 PCT(I)=FREQ(I)*100.0/SCNT                                         TAB11250
C                                                                       TAB11260
C        CALCULATE MEAN AND STANDARD DEVIATION                          TAB11270
C                                                                       TAB11280
      IF(SCNT-1.0) 85,85,90                                             TAB11290
   85 STATS(2)=STATS(1)                                                 TAB11300
      STATS(3)=0.0                                                      TAB11310
      GO TO 95                                                          TAB11320
   90 STATS(2)=STATS(1)/SCNT                                            TAB11330
      STATS(3)=SQRT(ABS((STATS(3)-STATS(1)*STATS(1)/SCNT)/(SCNT-1.0)))  TAB11340
   95 DO 100 I=1,3                                                      TAB11350
  100 UBC(I)=NBC(I)                                                     TAB11360
  105 RETURN                                                            TAB11370
      END                                                               TAB11380
```

ubroutine TAB2

This subroutine performs a two-way classification f the frequency, percent frequency, and other tatistics over given class intervals, for two se-ected variables in an observation matrix.

Interval size for each variable is computed as ollows:

$$k_j = \frac{UBO_{3j} - UBO_{1j}}{UBO_{2j} - 2} \qquad (1)$$

where UBO_{1j} = given lower bound

UBO_{2j} = given number of intervals

UBO_{3j} = given upper bound

$j = 1, 2$

f $UBO_{1j} = UBO_{3j}$, the subroutine finds and uses the ninimum and maximum values of the jth variable.

A frequency tabulation is then made for each pair f observations in a two-way table as shown in igure 6.

Symbols ≥ and < in Figure 6 indicate that a count s classified into a particular interval if the data oint is greater than or equal to the lower limit of hat interval but less than the upper limit of the same nterval.

Then, each entry in the frequency matrix, F_{ij}, is ivided by the number of observations, N, to obtain he percent frequency:

$$P_{ij} = \frac{100F_{ij}}{N} \qquad (2)$$

where $i = 1, 2, \ldots, UBO_{21}$

$j = 1, 2, \ldots, UBO_{22}$

As data are classified into the frequency matrix, the ollowing intermediate results are accumulated for ach class interval of both variables:

1. Number of data points, n
2. Sum of data points, $\sum_{i=1}^{n} X_i$
3. Sum of data points squared, $\sum_{i=1}^{n} X_i^2$

From these, the following statistics are calculated for each class interval:

Mean: $$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n} \qquad (3)$$

Standard deviation:

$$s = \sqrt{\frac{\sum_{i=1}^{n} X_i^2 - \left(\sum_{i=1}^{n} X_i\right)^2 / n}{n-1}} \qquad (4)$$

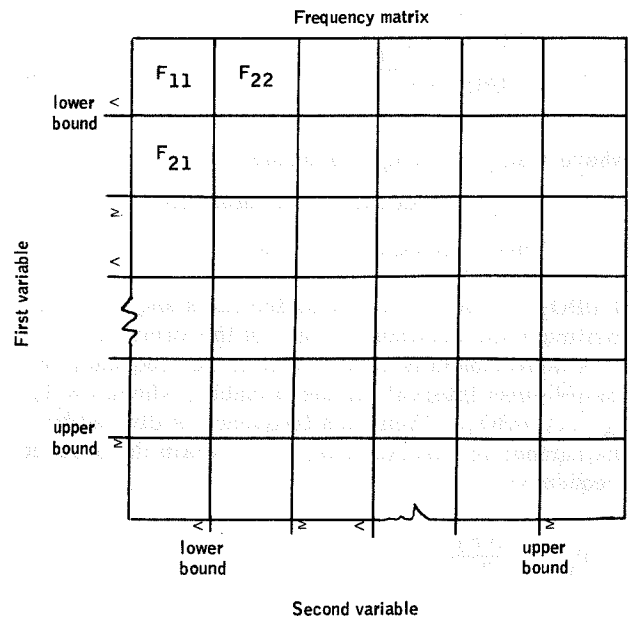


Figure 6. Frequency matrix

```
C                                                                       TAB2  10
C     ..............................................................TAB2  20
C                                                                       TAB2  30
C        SUBROUTINE TAB2                                                TAB2  40
C                                                                       TAB2  50
C        PURPOSE                                                        TAB2  60
C           PERFORMS A TWO-WAY CLASSIFICATION FOR TWO VARIABLES IN AN   TAB2  70
C           OBSERVATION MATRIX (OR A MATRIX SUBSET) OF THE FREQUENCY,   TAB2  80
C           PERCENT FREQUENCY, AND OTHER STATISTICS OVER GIVEN CLASS    TAB2  90
C           INTERVALS.                                                  TAB2 100
C                                                                       TAB2 110
C        USAGE                                                          TAB2 120
C           CALL TAB2(A,S,NOV,UBC,FREQ,PCT,STAT1,STAT2,NC,NV)           TAB2 130
C                                                                       TAB2 140
C        DESCRIPTION OF PARAMETERS                                      TAB2 150
C           A      - OBSERVATION MATRIX, NC BY NV                       TAB2 160
C           S      - INPUT VECTOR GIVING SUBSET OF A. ONLY THOSE        TAB2 170
C                    OBSERVATIONS WITH A CORRESPONDING NON-ZERO S(J) ARE TAB2 180
C                    CONSIDERED. VECTOR LENGTH IS NC.                   TAB2 190
C           NOV    - VARIABLES TO BE CROSS-TABULATED. NOV(1) IS VARIABLE TAB2 200
C                       1 AND NOV(2) IS VARIABLE 2. VECTOR LENGTH IS 2. NOVTAB2 210
C                       MUST BE GREATER THAN OR EQUAL TO 1 AND LESS THANTAB2 211
C                       OR EQUAL TO NV.                                 TAB2 212
C           UBC    - 3 BY 2 MATRIX GIVING LOWER LIMIT, NUMBER OF        TAB2 220
C                    INTERVALS, AND UPPER LIMIT OF BOTH VARIABLES TO BE TAB2 230
C                    TABULATED (FIRST COLUMN FOR VARIABLE 1, SECOND     TAB2 240
C                    COLUMN FOR VARIABLE 2). IF LOWER LIMIT IS EQUAL TO TAB2 250
C                    UPPER LIMIT FOR VARIABLE 1, THE PROGRAM USES THE   TAB2 260
C                    MINIMUM AND MAXIMUM VALUES ON EACH VARIABLE. NUMBER TAB2 270
C                    OF INTERVALS MUST INCLUDE TWO CELLS FOR UNDER AND  TAB2 280
C                    ABOVE LIMITS.                                      TAB2 290
C           FREQ   - OUTPUT MATRIX OF FREQUENCIES IN THE TWO-WAY        TAB2 300
C                    CLASSIFICATION. ORDER OF MATRIX IS INT1 BY INT2,   TAB2 310
C                    WHERE INT1 IS THE NUMBER OF INTERVALS OF VARIABLE 1 TAB2 320
C                    AND INT2 IS THE NUMBER OF INTERVALS OF VARIABLE 2. TAB2 330
C                    INT1 AND INT2 MUST BE SPECIFIED IN THE SECOND      TAB2 340
C                    POSITION OF RESPECTIVE COLUMN OF UBC MATRIX.       TAB2 350
C           PCT    - OUTPUT MATRIX OF PERCENT FREQUENCIES, SAME ORDER  TAB2 360
C                    AS FREQ.                                           TAB2 370
C           STAT1  - OUTPUT MATRIX SUMMARIZING TOTALS, MEANS, AND      TAB2 380
C                    STANDARD DEVIATIONS FOR EACH CLASS INTERVAL OF     TAB2 390
C                    VARIABLE 1. ORDER OF MATRIX IS 3 BY INT1.          TAB2 400
C           STAT2  - SAME AS STAT1 BUT OVER VARIABLE 2. ORDER OF MATRIX TAB2 410
C                    IS 3 BY INT2.                                      TAB2 420
C           NC     - NUMBER OF OBSERVATIONS. NC MUST BE GREATER THAN   TAB2 430
C                    OR EQUAL TO 1.                                     TAB2 431
C           NV     - NUMBER OF VARIABLES FOR EACH OBSERVATION. NV      TAB2 440
C                    MUST BE GREATER THAN OR EQUAL TO 1.                TAB2 441
C                                                                       TAB2 450
C        REMARKS                                                        TAB2 460
C           IF S IS NULL, OUTPUT AREAS ARE SET TO ZERO                  TAB2 470
C                                                                       TAB2 480
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TAB2 490
C           NONE                                                        TAB2 500
C                                                                       TAB2 510
C        METHOD                                                         TAB2 520
C           INTERVAL SIZES FOR BOTH VARIABLES ARE CALCULATED FROM THE   TAB2 530
C           GIVEN INFORMATION OR OPTIONALLY FROM THE MINIMUM AND MAXIMUMTAB2 540
```

```
C          VALUES. THE FREQUENCY AND PERCENT FREQUENCY MATRICES ARE
C          DEVELOPED. MATRICES STAT1 AND STAT2 SUMMARIZING TOTALS,
C          MEANS, AND STANDARD DEVIATIONS ARE THEN CALCULATED.
C          THE DIVISOR FOR STANDARD DEVIATION IS ONE LESS THAN THE
C          NUMBER OF OBSERVATIONS USED IN EACH CLASS INTERVAL.
C
C
C     ..................................................................
C
      SUBROUTINE TAB2(A,S,NOV,LBC,FREQ,PCT,STAT1,STAT2,NC,NV)
      DIMENSION A(1),S(1),NOV(2),UBC(3,2),FREQ(1),PCT(1),STAT1(1),
     1STAT2(2),SINT(2)
      DIMENSION NBC(2,2)
      DO 5 I=1,3
      DO 5 J=1,2
    5 NBC(I,J)=LBC(I,J)
C
C        DETERMINE LIMITS
C
      DO 40 I=1,2
      IF(UBC(1,I)-UBC(3,I)) 40, 10, 40
   10 VMIN=1.0E75
      VMAX=-1.0E75
      IJ=NO*(NOV(I)-1)
      DO 35 J=1,NO
      IJ=IJ+1
      IF(S(IJ)) 15,35,15
   15 IF(A(IJ)-VMIN) 20,25,25
   20 VMIN=A(IJ)
   25 IF(A(IJ)-VMAX) 35,35,30
   30 VMAX=A(IJ)
   35 CONTINUE
      UBC(1,I)=VMIN
      UBC(3,I)=VMAX
   40 CONTINUE
C
C        CALCULATE INTERVAL SIZE
C
   45 DO 50 I=1,2
   50 SINT(I)=ABS((UBC(3,I)-UBC(1,I))/(LBC(2,I)-2.0))
C
C        CLEAR OUTPUT AREAS
C
      INT1=LBC(2,1)
      INT2=LBC(2,2)
      INTT=INT1*INT2
      DO 55 I=1,INTT
      FREQ(I)=0.0
   55 PCT(I)=0.0
      INTY=3*INT1
      DO 60 I=1,INTY
   60 STAT1(I)=0.0
      INTZ=3*INT2
      DO 65 I=1,INTZ
   65 STAT2(I)=0.0
C
C        TEST SUBSET VECTOR
C
      SCNT=0.0
      INTY=INT1-1
      INTX=INT2-1
      IJ=NO*(NOV(1)-1)
      IJX=NO*(NOV(2)-1)
      DO 55 J=1,NO
      IJ=IJ+1
      IJX=IJX+1
      IF(S(IJ)) 70,55,70
   70 SCNT=SCNT+1.0
C
C        CALCULATE FREQUENCIES
C
      TEMP1=LBC(1,1)-SINT(1)
      DO 75 IY=1,INTY
      TEMP1=TEMP1+SINT(1)
      IF(A(IJ)-TEMP1) 80,75,75
   75 CONTINUE
      IY=INT1
   80 IYY=3*(IY-1)+1
      STAT1(IYY)=STAT1(IYY)+A(IJ)
      IYY=IYY+1
      STAT1(IYY)=STAT1(IYY)+1.0
      IYY=IYY+1
      STAT1(IYY)=STAT1(IYY)+A(IJ)*A(IJ)
      TEMP2=LBC(1,2)-SINT(2)
      DO 85 IX=1,INTX
      TEMP2=TEMP2+SINT(2)
      IF(A(IJX)-TEMP2) 90,85,85
   85 CONTINUE
      IX=INT2
   90 IJF=INT1*(IX-1)+IY
      FREQ(IJF)=FREQ(IJF)+1.0
      IX=3*(IX-1)+1
      STAT2(IX)=STAT2(IX)+A(IJX)
      IX=IX+1
      STAT2(IX)=STAT2(IX)+1.0
      IX=IX+1
      STAT2(IX)=STAT2(IX)+A(IJX)*A(IJX)
   95 CONTINUE
      IF (SCNT)98,151,98
C
C        CALCULATE PERCENT FREQUENCIES
C
   98 DO 100 I=1,INTT
  100 PCT(I)=FREQ(I)*100.0/SCNT
C
C        CALCULATE TOTALS, MEANS, STANDARD DEVIATIONS
C
      IXY=-1
      DO 120 I=1,INT1
      IXY=IXY+2
      ISC=IXY+1
      TEMP1=STAT1(IXY)
      SUM=STAT1(IXY-1)
      IF(TEMP1-1.0) 120,105,110
  105 STAT1(ISC)=0.0
      GO TO 115
  110 STAT1(ISC)=SQRT(ABS((STAT1(ISC)-SUM*SUM/TEMP1)/(TEMP1-1.0)))
  115 STAT1(IXY)=SUM/TEMP1
  120 CONTINUE
      IXX=-1
      DO 140 I=1,INT2
      IXX=IXX+3
      ISC=IXX+1
      TEMP2=STAT2(IXX)
      SUM=STAT2(IXX-1)
      IF(TEMP2-1.0) 140,125,130
  125 STAT2(ISC)=0.0
      GO TO 135
  130 STAT2(ISC)=SQRT(ABS((STAT2(ISC)-SUM*SUM/TEMP2)/(TEMP2-1.0)))
  135 STAT2(IXX)=SUM/TEMP2
  140 CONTINUE
      DO 150 I=1,3
      DO 150 J=1,2
  150 UBC(I,J)=NBC(I,J)
  151 RETURN
      END
```

Subroutine SUBMX

```
C
C     ..................................................................
C
C        SUBROUTINE SUBMX
C
C        PURPOSE
C           BASED ON VECTOR S DERIVED FROM SUBROUTINE SUBST OR ABSNT,
C           THIS SUBROUTINE COPIES FROM A LARGER MATRIX OF OBSERVATION
C           DATA A SUBSET MATRIX OF THOSE OBSERVATIONS WHICH HAVE
C           SATISFIED CERTAIN CONDITIONS.  THIS SUBROUTINE IS NORMALLY
C           USED PRIOR TO STATISTICAL ANALYSES (E.G., MULTIPLE REGRES-
C           SION, FACTOR ANALYSIS).
C
C        USAGE
C           CALL SUBMX (A,C,S,NC,NV,N)
C
C        DESCRIPTION OF PARAMETERS
C           A  - INPUT MATRIX OF OBSERVATIONS, NC BY NV.
C           C  - OUTPUT MATRIX OF OBSERVATIONS, N BY NV.
C           S  -  INPUT VECTOR OF LENGTH NO CONTAINING THE CODES DERIVED
C                 FROM SUBROUTINE SUBST OR ABSNT.
C           NC - NUMBER OF OBSERVATIONS. NC MUST BE > OR = TO 1.
C           NV - NUMBER OF VARIABLES. NV MUST BE > OR = TO 1.
C           N  - OUTPUT VARIABLE CONTAINING THE NUMBER OF NON-ZERO CODES
C                IN VECTOR S.
C
C        REMARKS
C           MATRIX C CAN BE IN THE SAME LOCATION AS MATRIX A.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           IF S(I) CONTAINS A NON-ZERO CODE, I-TH OBSERVATION IS
C           COPIED FROM THE INPUT MATRIX TO THE OUTPUT MATRIX.
C
C     ..................................................................
C
      SUBROUTINE SUBMX (A,C,S,NO,NV,N)
      DIMENSION A(1),C(1),S(1)
C
      L=0
      LL=0
      DO 20 J=1,NV
      DO 15 I=1,NO
      L=L+1
      IF(S(I)) 15, 15, 10
   10 LL=LL+1
      C(LL)=A(L)
   15 CONTINUE
   20 CONTINUE
C
C        COUNT NON-ZERO CODES IN VECTOR S
C
      N=0
      DO 30 I=1,NO
      IF(S(I)) 30, 30, 25
   25 N=N+1
   30 CONTINUE
C
      RETURN
      END
```

Correlation and Regression (see Smoothing, Factorization)

Subroutine CORRE

This subroutine calculates means, standard deviations, sums of cross-products of deviations from means, and product moment correlation coefficients from input data X_{ij}, where i = 1, 2, ..., n implies observations and j = 1, 2, ..., m implies variables.

The following equations are used to calculate these statistics.

Sums of cross-products of deviations:

$$S_{jk} = \sum_{i=1}^{n} \left(X_{ij} - T_j\right)\left(X_{ik} - T_k\right) - \frac{\sum_{i=1}^{n}\left(X_{ij} - T_j\right)\sum_{i=1}^{n}\left(X_{ik} - T_k\right)}{n} \quad (1)$$

where j = 1, 2, ..., m; k = 1, 2, ..., m

$$T_j = \frac{\sum_{i=1}^{m} X_{ij}}{m} \quad (2)$$

(These temporary means T_j are subtracted from the data in equation (1) to obtain computational accuracy.)

Means: $$\overline{X}_j = \frac{\sum_{i=1}^{n} X_{ij}}{n} \quad (3)$$

where j = 1, 2, ..., m

Correlation coefficients:

$$r_{jk} = \frac{S_{jk}}{\sqrt{S_{jj}}\ \sqrt{S_{kk}}} \quad (4)$$

where j = 1, 2, ..., m; k = 1, 2, ..., m

Standard deviations:

$$s_j = \frac{\sqrt{S_{jj}}}{\sqrt{n-1}} \quad (5)$$

where j = 1, 2, ..., m

```
C     ...............................................................
C
C        SUBROUTINE CORRE
C
C        PURPOSE
C           COMPUTE MEANS, STANDARD DEVIATIONS, SUMS OF CROSS-PRODUCTS
C           OF DEVIATIONS, AND CORRELATION COEFFICIENTS.
C
C        USAGE
C           CALL CORRE (N,M,IC,X,XBAR,STD,RX,R,B,D,T)
C
C        DESCRIPTION OF PARAMETERS
C           N      - NUMBER OF OBSERVATIONS. N MUST BE > OR = TO 2.
C           M      - NUMBER OF VARIABLES. M MUST BE > OR = TO 1.
C           IC     - OPTION CODE FOR INPUT DATA
C                    0 IF DATA ARE TO BE READ IN FROM INPUT DEVICE IN THE
C                      SPECIAL SUBROUTINE NAMED DATA.  (SEE SUBROUTINES
C                      USED BY THIS SUBROUTINE BELOW.)
C                    1 IF ALL DATA ARE ALREADY IN CORE.
C           X      - IF IC=0, THE VALUE OF X IS 0.0.
C                    IF IC=1, X IS THE INPUT MATRIX (N BY M) CONTAINING
C                             DATA.
C           XBAR   - OUTPUT VECTOR OF LENGTH M CONTAINING MEANS.
C           STD    - OUTPUT VECTOR OF LENGTH M CONTAINING STANDARD
C                    DEVIATIONS.
C           RX     - OUTPUT MATRIX (M X M) CONTAINING SUMS OF CROSS-
C                    PRODUCTS OF DEVIATIONS FROM MEANS.
C           R      - OUTPUT MATRIX (ONLY UPPER TRIANGULAR PORTION OF THE
C                    SYMMETRIC MATRIX OF M BY M) CONTAINING CORRELATION
C                    COEFFICIENTS.  (STORAGE MODE OF 1)
C           B      - OUTPUT VECTOR OF LENGTH M CONTAINING THE DIAGONAL
C                    OF THE MATRIX OF SUMS OF CROSS-PRODUCTS OF
C                    DEVIATIONS FROM MEANS.
C           D      - WORKING VECTOR OF LENGTH M.
C           T      - WORKING VECTOR OF LENGTH M.
C
C        REMARKS
C           CORRE WILL NOT ACCEPT A CONSTANT VECTOR.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           DATA(M,D) - THIS SUBROUTINE MUST BE PROVIDED BY THE USER.
C                       (1) IF IC=0, THIS SUBROUTINE IS EXPECTED TO
C                           FURNISH AN OBSERVATION IN VECTOR D FROM AN
C                           EXTERNAL INPUT DEVICE.
C                       (2) IF IC=1, THIS SUBROUTINE IS NOT USED BY
C                           CORRE BUT MUST EXIST IN JOB DECK. IF USER
C                           HAS NOT SUPPLIED A SUBROUTINE NAMED DATA,
C                           THE FOLLOWING IS SUGGESTED.
C                                SUBROUTINE DATA
C                                RETURN
C                                END
C
C        METHOD
C           PRODUCT-MOMENT CORRELATION COEFFICIENTS ARE COMPUTED.
C
C     ...............................................................
C
      SUBROUTINE CORRE (N,M,IC,X,XBAR,STD,RX,R,B,D,T)
      DIMENSION X(1),XBAR(1),STD(1),RX(1),R(1),B(1),D(1),T(1)
C
C        ............................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
C        DOUBLE PRECISION XBAR,STD,RX,R,B,T
C
C           THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C           APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C           ROUTINE.
C
C           THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C           CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT AND ABS IN
C           STATEMENT 220 MUST BE CHANGED TO DSQRT AND DABS.
C
C        ............................................................
C
C        INITIALIZATION
C
      DO 100 J=1,M
      B(J)=0.0
  100 T(J)=0.0
      K=(M*M+M)/2
      DO 102 I=1,K
  102 R(I)=0.0
      FN=N
      L=0
C
      IF(IC) 105, 127, 105
C
C     DATA ARE ALREADY IN CORE
C
  105 DO 108 J=1,M
      DO 107 I=1,N
      L=L+1
  107 T(J)=T(J)+X(L)
      XBAR(J)=T(J)
  108 T(J)=T(J)/FN
C
      DO 115 I=1,N
      JK=0
      L=I-N
      DO 110 J=1,M
      L=L+N
      D(J)=X(L)-T(J)
  110 B(J)=B(J)+D(J)
      DO 115 J=1,M
      DO 115 K=1,J
      JK=JK+1
  115 R(JK)=R(JK)+D(J)*D(K)
      GO TO 205
C
C     READ OBSERVATIONS AND CALCULATE TEMPORARY
C     MEANS FROM THESE DATA IN T(J)
C
  127 IF(N-M) 130, 130, 135
  130 NN=N
      GO TO 137
  135 NN=M
  137 DO 140 I=1,NN
      CALL DATA (M,D)
      DO 140 J=1,M
      T(J)=T(J)+D(J)
      L=L+1
  140 RX(L)=D(J)
      FNN=NN
      DO 150 J=1,M
      XBAR(J)=T(J)
  150 T(J)=T(J)/FNN
```

```
C
C     CALCULATE SUMS OF CROSS-PRODUCTS OF DEVIATIONS
C     FROM TEMPORARY MEANS FOR M OBSERVATIONS
C
      L=0
      DO 180 I=1,KK
      JK=0
      DO 170 J=1,M
      L=L+1
  170 D(J)=RX(L)-T(J)
      DO 180 J=1,M
      E(J)=E(J)+D(J)
      DO 180 K=1,J
      JK=JK+1
  180 R(JK)=R(JK)+D(J)*D(K)
C
      IF(N-KK) 205, 205, 185
C
C     READ THE REST OF OBSERVATIONS ONE AT A TIME, SUM
C     THE OBSERVATION, AND CALCULATE SUMS OF CROSS-
C     PRODUCTS OF DEVIATIONS FROM TEMPORARY MEANS
C
  185 KK=N-KK
      DO 200 I=1,KK
      JK=0
      CALL DATA (M,D)
      DO 190 J=1,M
      XBAR(J)=XBAR(J)+D(J)
      D(J)=D(J)-T(J)
  190 E(J)=E(J)+D(J)
      DO 200 J=1,M
      DO 200 K=1,J
      JK=JK+1
  200 R(JK)=R(JK)+D(J)*D(K)
C
C     CALCULATE MEANS
C
  205 JK=0
      DO 210 J=1,M
      XBAR(J)=XBAR(J)/FN
C
C     ADJUST SUMS OF CROSS-PRODUCTS OF DEVIATIONS
C     FROM TEMPORARY MEANS
C
      DO 210 K=1,J
      JK=JK+1
  210 R(JK)=R(JK)-E(J)*E(K)/FN
C
C     CALCULATE CORRELATION COEFFICIENTS
C
      JK=0
      DO 220 J=1,M
      JK=JK+J
  220 STD(J)= SQRT( ABS(R(JK)))
      DO 230 J=1,M
      DO 230 K=J,M
      JK=J*(K*K-K)/2
      L=M*(J-1)+K
      RX(L)=R(JK)
      L=M*(K-1)+J
      RX(L)=R(JK)
      IF(STD(J)*STD(K)) 225, 222, 225
  222 R(JK)=0.0
      GO TO 230
  225 R(JK)=R(JK)/(STD(J)*STD(K))
  230 CONTINUE
C
C     CALCULATE STANDARD DEVIATIONS
C
      FN=SQRT(FN-1.0)
      DO 240 J=1,M
  240 STD(J)=STD(J)/FN
C
C     COPY THE DIAGONAL OF THE MATRIX OF SUMS OF CROSS-PRODUCTS OF
C     DEVIATIONS FROM MEANS.
C
      L=-M
      DO 250 I=1,M
      L=L+M+1
  250 E(I)=RX(L)
      RETURN
      END
```

Subroutine MISR

This subroutine computes means, standard deviations, third and fourth moments, correlation coefficients, regression coefficients, and standard errors of regression coefficients when there is missing data. Effective sample sizes are also provided. Missing observations or certain values of the data can be skipped at the user's option. The computational steps are as follows:

1. Compute means:

$$\bar{x}_j = \frac{\sum_{\alpha=1}^{n} x_{\alpha j}}{n} \qquad (1)$$

where $j = 1, 2, \ldots, m$ implies variables
n = number of nonmissing values for the jth variable

2. Compute sums of cross-products of deviations from means for complete sets of ith and jth variables:

$$S_{ij} = \sum_{\alpha=1}^{n'} (x_{\alpha i} - \bar{x}_i)(x_{\alpha j} - \bar{x}_j) - \frac{\sum_{\alpha=1}^{n'} (x_{\alpha i} - \bar{x}_i) \sum_{\alpha=1}^{n'} (x_{\alpha j} - \bar{x}_j)}{n'} \qquad (2)$$

where $\bar{x}_i$, $\bar{x}_j$ = means of ith and jth variables computed as above
n' = number of sets where the ith and jth variables are both present

3. Compute product-moment correlations:

$$r_{ij} = \frac{S_{ij}}{\sqrt{S_{ii}}\sqrt{S_{jj}}} \qquad (3)$$

4. Compute regression coefficients, intercepts, and standard errors of regression coefficients:

a. ith variable as independent and jth variable as dependent:

Regression coefficient: $b_{ij} = \frac{S_{ij}}{S_{ii}}$ (4)

Intercept: $a_{ij} = \bar{x}'_j - b_{ij}\bar{x}'_i$ (5)

Standard error regression coefficient:

$$s_{b_{ij}} = \frac{S_{jj} - r_{ij}^2 \, S_{jj}}{(n' - 2) \, S_{ii}} \tag{6}$$

b. jth variable as independent and ith variable as dependent:

Regression coefficient: $$b_{ji} = \frac{S_{ij}}{S_{jj}} \tag{7}$$

Intercept: $$a_{ji} = \bar{x}_i' - b_{ji} \bar{x}_j' \tag{8}$$

Standard error of regression coefficient:

$$s_{b_{ji}} = \frac{S_{ii} - r_{ij}^2 \, S_{ii}}{(n' - 2) \, S_{jj}} \tag{9}$$

5. Compute standard deviations:

$$s_j = \sqrt{\frac{S_{jj}}{n - 1}} \tag{10}$$

where n = number of nonmissing values for jth variable

6. Compute skewness and kurtosis, s and k:

Let $$\mu_{i_j} = \sum_{\alpha = 1}^{n} (x_{\alpha_j} - \bar{x}_j)^{\,i}/n$$

Then $$s_j = \mu_{3j} / (\mu_{2j})^{3/2} \tag{11}$$

and $$k_j = (\mu_{4_j}/\mu_{2j}^2) - 3 \tag{12}$$

Note: This subroutine cannot distinguish between a blank and a zero. Therefore, if a blank is specified as a missing data code in input cards, it will be treated as 0 (zero).

For reference see B. Ostle, Statistics in Research. The Iowa State College Press, 1954, chapter 6.

```
C                                                                       MISR  10
C     ..................................................................MISR  20
C                                                                       MISR  30
C        SUBROUTINE MISR                                                MISR  40
C                                                                       MISR  50
C        PURPOSE                                                        MISR  60
C           COMPUTE MEANS, STANDARD DEVIATIONS, SKEWNESS AND KURTOSIS,  MISR  70
C           CORRELATION COEFFICIENTS, REGRESSION COEFFICIENTS, AND      MISR  80
C           STANDARD ERRORS OF REGRESSION COEFFICIENTS WHEN THERE ARE   MISR  90
C           MISSING DATA POINTS.  THE USER IDENTIFIES THE MISSING DATA  MISR 100
C           BY MEANS OF A NUMERIC CODE.  THOSE VALUES HAVING THIS CODE  MISR 110
C           ARE SKIPPED IN COMPUTING THE STATISTICS.  IN THE CASE OF THEMISR 120
C           CORRELATION COEFFICIENTS, ANY PAIR OF VALUES ARE SKIPPED IF MISR 130
C           EITHER ONE OF THEM ARE MISSING.                             MISR 140
C                                                                       MISR 150
C        USAGE                                                          MISR 160
C           CALL MISR (NO,M,X,CODE,XBAR,STD,SKEW,CURT,R,N,A,B,S,IER)    MISR 170
C                                                                       MISR 180
C        DESCRIPTION OF PARAMETERS                                      MISR 190
C           NO   - NUMBER OF OBSERVATIONS                               MISR 200
C           M    - NUMBER OF VARIABLES                                  MISR 210
C           X    - INPUT DATA MATRIX OF SIZE NO X M.                    MISR 220
C           CODE - INPUT VECTOR OF LENGTH M, WHICH CONTAINS A NUMERIC   MISR 230
C                  MISSING DATA CODE FOR EACH VARIABLE. ANY OBSERVATION MISR 240
C                  FOR A GIVEN VARIABLE HAVING A VALUE EQUAL TO THE CODEMISR 250
C                  WILL BE DROPPED FOR THE COMPUTATIONS.                MISR 260
C           XBAR - OUTPUT VECTOR OF LENGTH M CONTAINING MEANS           MISR 270
C           STD  - OUTPUT VECTOR OF LENGTH M CONTAINING STANDARD DEVI-  MISR 280
C                  ATIONS                                               MISR 290
C           SKEW - OUTPUT VECTOR OF LENGTH M CONTAINING SKEWNESS        MISR 300
C           CURT - OUTPUT VECTOR OF LENGTH M CONTAINING KURTOSIS        MISR 310
C           R    - OUTPUT MATRIX OF PRODUCT-MOMENT CORRELATION          MISR 320
C                  COEFFICIENTS.  THIS WILL BE THE UPPER TRIANGULAR     MISR 330
C                  MATRIX ONLY, SINCE THE M X M MATRIX OF COEFFICIENTS  MISR 340
C                  IS SYMMETRIC. (STORAGE MODE 1)                       MISR 350
C           N    - OUTPUT MATRIX OF NUMBER OF PAIRS OF OBSERVATIONS USEDMISR 360
C                  IN COMPUTING THE CORRELATION COEFFICIENTS.  ONLY THE MISR 370
C                  UPPER TRIANGULAR PORTION OF THE MATRIX IS GIVEN.     MISR 380
C                  (STORAGE MODE 1)                                     MISR 390
C           A    - OUTPUT MATRIX (M BY M)  CONTAINING INTERCEPTS OF     MISR 400
C                  REGRESSION LINES (A) OF THE FORM Y=A+BX.  THE FIRST  MISR 410
C                  SUBSCRIPT OF THIS MATRIX REFERS TO THE INDEPENDENT   MISR 420
C                  VARIABLE AND THE SECOND TO THE DEPENDENT VARIABLE.   MISR 430
C                  FOR EXAMPLE, A(1,3) CONTAINS THE INTERCEPT OF THE    MISR 440
C                  REGRESSION LINE FOR TWO VARIABLES WHERE VARIABLE 1   MISR 450
C                  IS INDEPENDENT AND VARIABLE 3 IS DEPENDENT.  NOTE    MISR 460
C                  THAT MATRIX A IS STORED IN A VECTOR FORM.            MISR 470
C           B    - OUTPUT MATRIX (M BY M)  CONTAINING REGRESSION        MISR 480
C                  COEFFICIENTS (B) CORRESPONDING TO THE VALUES OF      MISR 490
C                  INTERCEPTS CONTAINED IN THE OUTPUT MATRIX A.         MISR 500
C           S    - OUTPUT MATRIX (M BY M)  CONTAINING STANDARD ERRORS   MISR 510
C                  OF REGRESSION COEFFICIENTS CORRESPONDING TO THE      MISR 520
C                  COEFFICIENTS CONTAINED IN THE OUTPUT MATRIX B.       MISR 530
C           IER  - 0, NO ERROR.                                         MISR 540
C                  1, IF NUMBER OF NON-MISSING DATA ELEMENTS FOR J-TH   MISR 550
C                     VARIABLE IS TWO OR LESS.  IN THIS CASE, STD(J),   MISR 560
C                     SKEW(J), AND CURT(J) ARE SET TO 10**75.  ALL      MISR 570
C                     VALUES OF R, A, B, AND S RELATED TO THIS VARIABLE MISR 580
C                     ARE ALSO SET TO 10**75.                           MISR 590
C                  2, IF VARIANCE OF J-TH VARIABLE IS LESS THAN         MISR 600
C                     10**(-20).  IN THIS CASE, STD(J), SKEW(J), AND    MISR 610
C                     CURT(J) ARE SET TO 10**75.  ALL VALUES OF R, A,   MISR 620
C                     B, AND S RELATED TO THIS VARIABLE ARE ALSO SET TO MISR 630
C                     10**75.                                           MISR 640
C                                                                       MISR 650
C        REMARKS                                                        MISR 660
C           THIS SUBROUTINE CANNOT DISTINGUISH A BLANK AND A ZERO.      MISR 670
C           THEREFORE, IF A BLANK IS SPECIFIED AS A MISSING DATA CODE INMISR 680
C           INPUT CARDS, IT WILL BE TREATED AS 0 (ZERO).                MISR 690
C                                                                       MISR 700
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MISR 710
C           NONE                                                        MISR 720
C                                                                       MISR 730
C        METHOD                                                         MISR 740
C           LEAST SQUARES REGRESSION LINES AND PRODUCT-MOMENT CORRE-    MISR 750
C           LATION COEFFICIENTS ARE COMPUTED.                           MISR 760
C                                                                       MISR 770
C     ..................................................................MISR 780
C                                                                       MISR 790
      SUBROUTINE MISR (NO,M,X,CODE,XBAR,STD,SKEW,CURT,R,N,A,B,S,IER)    MISR 800
C                                                                       MISR 810
      DIMENSION X(1),CODE(1),XBAR(1),STD(1),SKEW(1),CURT(1),R(1),N(1)   MISR 820
      DIMENSION A(1),B(1),S(1)                                          MISR 830
C                                                                       MISR 840
C        COMPUTE MEANS                                                  MISR 850
C                                                                       MISR 860
      IER=0                                                             MISR 870
      L=0                                                               MISR 880
      DO 20 J=1,M                                                       MISR 890
      FN=0.0                                                            MISR 900
      XBAR(J)=0.0                                                       MISR 910
      DO 15 I=1,NO                                                      MISR 920
      L=L+1                                                             MISR 930
      IF(X(L)-CODE(J)) 12, 15, 12                                       MISR 940
   12 FN=FN+1.0                                                         MISR 950
      XBAR(J)=XBAR(J)+X(L)                                              MISR 960
   15 CONTINUE                                                          MISR 970
      IF(FN) 16, 16, 17                                                 MISR 980
   16 XBAR(J)=0.0                                                       MISR 990
      GO TO 20                                                          MISR1000
   17 XBAR(J)=XBAR(J)/FN                                                MISR1010
   20 CONTINUE                                                          MISR1020
C                                                                       MISR1030
C        SET-UP WORK AREAS AND TEST WHETHER DATA IS MISSING             MISR1040
C                                                                       MISR1050
      L=0                                                               MISR1060
      DO 55 J=1,M                                                       MISR1070
      LJJ=NO*(J-1)                                                      MISR1080
      SKEW(J)=0.0                                                       MISR1090
      CURT(J)=0.0                                                       MISR1100
      KI=M*(J-1)                                                        MISR1110
      KJ=J-M                                                            MISR1120
      DO 54 I=1,J                                                       MISR1130
      KI=KI+1                                                           MISR1140
      KJ=KJ+M                                                           MISR1150
      SUMX=0.0                                                          MISR1160
      SUMY=0.0                                                          MISR1170
      TI=0.0                                                            MISR1180
      TJ=0.0                                                            MISR1190
      TII=0.0                                                           MISR1200
      TJJ=0.0                                                           MISR1210
      TIJ=0.0                                                           MISR1220
      NIJ=0                                                             MISR1230
      LI=NO*(I-1)                                                       MISR1240
      LJ=LJJ                                                            MISR1250
      L=L+1                                                             MISR1260
      DO 38 K=1,NO                                                      MISR1270
      LI=LI+1                                                           MISR1280
      LJ=LJ+1                                                           MISR1290
      IF(X(LI)-CODE(I)) 30, 38, 30                                      MISR1300
   30 IF(X(LJ)-CODE(J)) 35, 38, 35                                      MISR1310
```

```
C
C       BOTH DATA ARE PRESENT
C
   35 XX=X(LI)-XBAR(I)
      YY=X(LJ)-XBAR(J)
      TI=TI+XX
      TII=TII+XX**2
      TJ=TJ+YY
      TJJ=TJJ+YY**2
      TIJ=TIJ+XX*YY
      NIJ=NIJ+1
      SUMX=SUMX+X(LI)
      SUMY=SUMY+X(LJ)
      IF(I-J) 38, 37, 37
   37 SKEW(J)=SKEW(J)+YY**3
      CURT(J)=CURT(J)+YY**4
   38 CONTINUE
C
C       COMPUTE SUM OF CROSS-PRODUCTS OF DEVIATIONS
C
      IF(NIJ) 40, 40, 39
   39 FN=NIJ
      R(L)=TIJ-TI*TJ/FN
      N(L)=NIJ
      TII=TII-TI*TI/FN
      TJJ=TJJ-TJ*TJ/FN
C
C       COMPUTE STANDARD DEVIATION, SKEWNESS, AND KURTOSIS
C
   40 IF(I-J) 47, 41, 47
   41 IF(NIJ-2) 42,42,43
   42 IER=1
      R(L)=1.0E75
      A(KI)=1.0E75
      B(KI)=1.0E75
      S(KI)=1.0E75
      GO TO 45
C
   43 STD(J)=R(L)
      R(L)=1.0
      A(KI)=0.0
      B(KI)=1.0
      S(KI)=0.0
C
      IF(STD(J)-(1.0E-20)) 44,44,46
   44 IER=2
   45 STD(J)=1.0E75
      SKEW(J)=1.0E75
      CURT(J)=1.0E75
      GO TO 55
C
   46 WORK=STD(J)/FN
      SKEW(J)=(SKEW(J)/FN)/(WORK*SQRT(WORK))
      CURT(J)=((CURT(J)/FN)/WORK**2)-3.0
      STD(J)=SQRT(STD(J)/(FN-1.0))
      GO TO 55
C
C       COMPUTE REGRESSION COEFFICIENTS
C
   47 IF(NIJ-2) 48,48,50
   48 IER=1
   49 R(L)=1.0E75
      A(KI)=1.0E75
      B(KI)=1.0E75
      S(KI)=1.0E75
      A(KJ)=1.0E75
      B(KJ)=1.0E75
      S(KJ)=1.0E75
      GO TO 54
C
   50 IF(TII-(1.0E-20)) 52,52,51
   51 IF(TJJ-(1.0E-20)) 52,52,53
   52 IER=2
      GO TO 49
C
   53 SUMX=SUMX/FN
      SUMY=SUMY/FN
      B(KI)=R(L)/TII
      A(KI)=SUMY-B(KI)*SUMX
      B(KJ)=R(L)/TJJ
      A(KJ)=SUMX-B(KJ)*SUMY
C
C       COMPUTE CORRELATION COEFFICIENTS
C
      R(L)=R(L)/(SQRT(TII)*SQRT(TJJ))
C
C       COMPUTE STANDARD ERRORS OF REGRESSION COEFFICIENTS
C
      RR=R(L)**2
      SUMX=(TJJ-TJJ*RR)/(FN-2)
      S(KI)=SQRT(SUMX/TII)
      SUMY=(TII-TII*RR)/(FN-2)
      S(KJ)=SQRT(SUMY/TJJ)
C
   54 CONTINUE
   55 CONTINUE
C
      RETURN
      END
```

Multiple Linear Regression

In the Scientific Subroutine Package, multiple linear regression is normally performed by calling four subroutines in sequence.

1. CORRE - to find means, standard deviations, and correlation matrix
2. ORDER - to choose a dependent variable and a subset of independent variables from a larger set of variables
3. MINV - to invert the correlation matrix of the subset selected by ORDER
4. MULTR - to compute the regression coefficients b_0, b_1, b_2, ..., b_m, and various confidence measures

The subroutine CORRE works in either of two ways: (1) it expects all observations in core, or (2) it triggers a user-provided input subroutine, DATA, to read one observation at a time into a work area. In either case, the user must provide a subroutine named DATA (see "Subroutines Required" in the comment cards description of subroutine CORRE).

Subroutine ORDER

```
C                                                                       ORDE  10
C     ...................................................................ORDE  20
C                                                                       ORDE  30
C        SUBROUTINE ORDER                                               ORDE  40
C                                                                       ORDE  50
C        PURPOSE                                                        ORDE  60
C           CONSTRUCT FROM A LARGER MATRIX OF CORRELATION COEFFICIENTS  ORDE  70
C           A SUBSET MATRIX OF INTERCORRELATIONS AMONG INDEPENDENT      ORDE  80
C           VARIABLES AND A VECTOR OF INTERCORRELATIONS OF INDEPENDENT  ORDE  90
C           VARIABLES WITH DEPENDENT VARIABLE.  THIS SUBROUTINE IS      ORDE 100
C           NORMALLY USED IN THE PERFORMANCE OF MULTIPLE AND POLYNOMIAL ORDE 110
C           REGRESSION ANALYSES.                                        ORDE 120
C                                                                       ORDE 130
C        USAGE                                                          ORDE 140
C           CALL ORDER (M,R,NDEP,K,ISAVE,RX,RY)                         ORDE 150
C                                                                       ORDE 160
C        DESCRIPTION OF PARAMETERS                                      ORDE 170
C           M     - NUMBER OF VARIABLES AND ORDER OF MATRIX R.          ORDE 180
C           R     - INPUT MATRIX CONTAINING CORRELATION COEFFICIENTS.   ORDE 190
C                   THIS SUBROUTINE EXPECTS ONLY UPPER TRIANGULAR       ORDE 200
C                   PORTION OF THE SYMMETRIC MATRIX TO BE STORED (BY    ORDE 210
C                   COLUMN) IN R.  (STORAGE MODE OF 1)                  ORDE 220
C           NDEP  - THE SUBSCRIPT NUMBER OF THE DEPENDENT VARIABLE.     ORDE 230
C           K     - NUMBER OF INDEPENDENT VARIABLES TO BE INCLUDED      ORDE 240
C                   IN THE FORTHCOMING REGRESSION. K MUST BE GREATER    ORDE 250
C                   THAN OR EQUAL TO 1.                                 ORDE 251
C           ISAVE - INPUT VECTOR OF LENGTH K+1 CONTAINING, IN ASCENDING ORDE 260
C                   ORDER, THE SUBSCRIPT NUMBERS OF K INDEPENDENT       ORDE 270
C                   VARIABLES TO BE INCLUDED IN THE FORTHCOMING REGRES- ORDE 280
C                   SION.                                               ORDE 290
C                   UPON RETURNING TO THE CALLING ROUTINE, THIS VECTOR  ORDE 300
C                   CONTAINS, IN ADDITION, THE SUBSCRIPT NUMBER OF      ORDE 310
C                   THE DEPENDENT VARIABLE IN K+1 POSITION.             ORDE 320
C           RX    - OUTPUT MATRIX (K X K) CONTAINING INTERCORRELATIONS  ORDE 330
C                   AMONG INDEPENDENT VARIABLES TO BE USED IN FORTH-    ORDE 340
C                   COMING REGRESSION.                                  ORDE 350
C           RY    - OUTPUT VECTOR OF LENGTH K CONTAINING INTERCORRELA-  ORDE 360
C                   TIONS OF INDEPENDENT VARIABLES WITH DEPENDENT       ORDE 370
C                   VARIABLES.                                          ORDE 380
C                                                                       ORDE 390
C        REMARKS                                                        ORDE 400
C           NONE                                                        ORDE 410
C                                                                       ORDE 420
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ORDE 430
C           NONE                                                        ORDE 440
C                                                                       ORDE 450
C        METHOD                                                         ORDE 460
C           FROM THE SUBSCRIPT NUMBERS OF THE VARIABLES TO BE INCLUDED  ORDE 470
C           IN THE FORTHCOMING REGRESSION, THE SUBROUTINE CONSTRUCTS THEORDE 480
C           MATRIX RX AND THE VECTOR RY.                                ORDE 490
C                                                                       ORDE 500
C     ...................................................................ORDE 510
C                                                                       ORDE 520
      SUBROUTINE ORDER (M,R,NDEP,K,ISAVE,RX,RY)                         ORDE 530
      DIMENSION R(1),ISAVE(1),RX(1),RY(1)                               ORDE 540
C                                                                       ORDE 550
C        ...................................................................ORDE 560
C                                                                       ORDE 570
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  ORDE 580
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      ORDE 590
C        STATEMENT WHICH FOLLOWS.                                       ORDE 600
C                                                                       ORDE 610
C     DOUBLE PRECISION R,RX,RY                                          ORDE 620
C                                                                       ORDE 630
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    ORDE 640
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      ORDE 650
C        ROUTINE.                                                       ORDE 660
C                                                                       ORDE 670
C        ...................................................................ORDE 680
C                                                                       ORDE 690
C     COPY INTERCORRELATIONS OF INDEPENDENT VARIABLES                   ORDE 700
C     WITH DEPENDENT VARIABLE                                           ORDE 710
C                                                                       ORDE 720
      MM=0                                                              ORDE 730
      DO 130 J=1,K                                                      ORDE 740
      L2=ISAVE(J)                                                       ORDE 750
      IF(NDEP-L2) 122, 123, 123                                         ORDE 760
  122 L=NDEP+(L2*L2-L2)/2                                               ORDE 770
      GO TO 125                                                         ORDE 780
  123 L=L2+(NDEP*NDEP-NDEP)/2                                           ORDE 790
  125 RY(J)=R(L)                                                        ORDE 800
C                                                                       ORDE 810
C     COPY A SUBSET MATRIX OF INTERCORRELATIONS AMONG                   ORDE 820
C     INDEPENDENT VARIABLES                                             ORDE 830
C                                                                       ORDE 840
      DO 130 I=1,K                                                      ORDE 850
      L1=ISAVE(I)                                                       ORDE 860
      IF(L1-L2) 127, 128, 128                                           ORDE 870
  127 L=L1+(L2*L2-L2)/2                                                 ORDE 880
      GO TO 129                                                         ORDE 890
  128 L=L2+(L1*L1-L1)/2                                                 ORDE 900
  129 MM=MM+1                                                           ORDE 910
  130 RX(MM)=R(L)                                                       ORDE 920
C                                                                       ORDE 930
C     PLACE THE SUBSCRIPT NUMBER OF THE DEPENDENT                       ORDE 940
C     VARIABLE IN ISAVE(K+1)                                            ORDE 950
C                                                                       ORDE 960
      ISAVE(K+1)=NDEP                                                   ORDE 970
      RETURN                                                            ORDE 980
      END                                                               ORDE 990
```

Subroutine MULTR

This subroutine performs a multiple regression analysis for a dependent variable and a set of independent variables.

Beta weights are calculated using the following equation:

$$\beta_j = \sum_{i=1}^{k} r_{iy} \cdot r_{ij}^{-1} \qquad (1)$$

where r_{iy} = intercorrelation of ith independent variable with dependent variable

r_{ij}^{-1} = the inverse of intercorrelation r_{ij}

$i, j = 1, 2, \ldots, k$ imply independent variables

r_{iy} and r_{ij}^{-1} are input to this subroutine.

Then, the regression coefficients are calculated as follows:

$$b_j = \beta_j \cdot \frac{s_y}{s_j} \qquad (2)$$

where s_y = standard deviation of dependent variable

s_j = standard deviation of jth independent variable

$j = 1, 2, \ldots, k$

s_y and s_j are input to this subroutine.

The intercept is found by the following equation:

$$b_0 = \overline{Y} - \sum_{j=1}^{k} b_j \cdot \overline{X}_j \qquad (3)$$

where $\overline{Y}$ = mean of dependent variable

$\overline{X}_j$ = mean of jth independent variable

$\overline{Y}$ and $\overline{X}_j$ are input to this subroutine.

Multiple correlation coefficient, R, is found first by calculating the coefficient of determination by the following equation:

$$R^2 = \sum_{i=1}^{k} \beta_i r_{iy} \qquad (4)$$

and taking the square root of R^2:

$$R = \sqrt{R^2} \qquad (5)$$

The sum of squares attributable to the regression is found by:

$$SSAR = R^2 \cdot D_{yy} \qquad (6)$$

where D_{yy} = sum of squares of deviations from mean for dependent variable

D_{yy} is input to this subroutine.

The sum of squares of deviations from the regression is obtained by:

$$SSDR = D_{yy} - SSAR \qquad (7)$$

Then, the F-value for the analysis of variance is calculated as follows:

$$F = \frac{SSAR/k}{SSDR/(n-k-1)} = \frac{SSAR(n-k-1)}{SSDR(k)} \qquad (8)$$

Certain other statistics are calculated as follows:

Variance and standard error of estimate:

$$S^2_{y.12\ldots k} = \frac{SSDR}{n-k-1} \qquad (9)$$

where n = number of observations

$$S_{y.12\ldots k} = \sqrt{S^2_{y.12\ldots k}} \qquad (10)$$

Standard deviations of regression coefficients:

$$S_{b_j} = \sqrt{\frac{r_{jj}^{-1}}{D_{jj}} \cdot S^2_{y.12\ldots k}} \qquad (11)$$

where D_{jj} = sum of squares of deviations from mean for j^{th} independent variable. D_{jj} is input to this subroutine.

$j = 1, 2, \ldots, k$

Computed t:

$$t_j = \frac{b_j}{S_{b_j}} \qquad (12)$$

$j = 1, 2, \ldots, k$

```
C
C ......................................................................
C
C     SUBROUTINE MULTR
C
C     PURPOSE
C        PERFORM A MULTIPLE LINEAR REGRESSION ANALYSIS FOR A
C        DEPENDENT VARIABLE AND A SET OF INDEPENDENT VARIABLES.  THIS
C        SUBROUTINE IS NORMALLY USED IN THE PERFORMANCE OF MULTIPLE
C        AND POLYNOMIAL REGRESSION ANALYSES.
C
C     USAGE
C        CALL MULTR (N,K,XBAR,STD,D,RX,RY,ISAVE,B,SB,T,ANS)
C
C     DESCRIPTION OF PARAMETERS
C        N     - NUMBER OF OBSERVATIONS.
C        K     - NUMBER OF INDEPENDENT VARIABLES IN THIS REGRESSION.
C        XBAR  - INPUT VECTOR OF LENGTH M CONTAINING MEANS OF ALL
C                VARIABLES. M IS NUMBER OF VARIABLES IN OBSERVATIONS.
C        STD   - INPUT VECTOR OF LENGTH M CONTAINING STANDARD DEVI-
C                ATIONS OF ALL VARIABLES.
C        D     - INPUT VECTOR OF LENGTH M CONTAINING THE DIAGONAL OF
C                THE MATRIX OF SUMS OF CROSS-PRODUCTS OF DEVIATIONS
C                FROM MEANS FOR ALL VARIABLES.
C        RX    - INPUT MATRIX (K X K) CONTAINING THE INVERSE OF
C                INTERCORRELATIONS AMONG INDEPENDENT VARIABLES.
C        RY    - INPUT VECTOR OF LENGTH K CONTAINING INTERCORRELA-
C                TIONS OF INDEPENDENT VARIABLES WITH DEPENDENT
C                VARIABLE.
C        ISAVE - INPUT VECTOR OF LENGTH K+1 CONTAINING SUBSCRIPTS OF
C                INDEPENDENT VARIABLES IN ASCENDING ORDER.  THE
C                SUBSCRIPT OF THE DEPENDENT VARIABLE IS STORED IN
C                THE LAST, K+1, POSITION.
C        B     - OUTPUT VECTOR OF LENGTH K CONTAINING REGRESSION
C                COEFFICIENTS.
C        SB    - OUTPUT VECTOR OF LENGTH K CONTAINING STANDARD
C                DEVIATIONS OF REGRESSION COEFFICIENTS.
C        T     - OUTPUT VECTOR OF LENGTH K CONTAINING T-VALUES.
C        ANS   - OUTPUT VECTOR OF LENGTH 10 CONTAINING THE FOLLOWING
C                INFORMATION..
C                ANS(1)  INTERCEPT
C                ANS(2)  MULTIPLE CORRELATION COEFFICIENT
C                ANS(3)  STANDARD ERROR OF ESTIMATE
C                ANS(4)  SUM OF SQUARES ATTRIBUTABLE TO REGRES-
C                        SION (SSAR)
C                ANS(5)  DEGREES OF FREEDOM ASSOCIATED WITH SSAR
C                ANS(6)  MEAN SQUARE OF SSAR
C                ANS(7)  SUM OF SQUARES OF DEVIATIONS FROM REGRES-
C                        SION (SSDR)
C                ANS(8)  DEGREES OF FREEDOM ASSOCIATED WITH SSDR
C                ANS(9)  MEAN SQUARE OF SSDR
C                ANS(10) F-VALUE
C
C     REMARKS
C        N MUST BE GREATER THAN K+1.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        NONE
C
C     METHOD
C        THE GAUSS-JORDAN METHOD IS USED IN THE SOLUTION OF THE
C        NORMAL EQUATIONS.  REFER TO W. W. COOLEY AND P. R. LOHNES,
C        'MULTIVARIATE PROCEDURES FOR THE BEHAVIORAL SCIENCES',
C        JOHN WILEY AND SONS, 1962, CHAPTER 3, AND B. OSTLE,
C        'STATISTICS IN RESEARCH', THE IOWA STATE COLLEGE PRESS,
C        1954, CHAPTER 8.
C
C ......................................................................
C
      SUBROUTINE MULTR (N,K,XBAR,STD,D,RX,RY,ISAVE,B,SB,T,ANS)
      DIMENSION XBAR(1),STD(1),D(1),RX(1),RY(1),ISAVE(1),B(1),SB(1),
     1          T(1),ANS(1)
C
C     ......................................................................
C
C     IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C     C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C     STATEMENT WHICH FOLLOWS.
C
C     DOUBLE PRECISION XBAR,STD,D,RX,RY,B,SB,T,ANS,RM,BO,SSAR,SSDR,SY,
C    1                 FN,FK,SSARM,SSDRM,F
C
C     THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C     APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C     ROUTINE.
C
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT AND ABS IN
C        STATEMENTS 122, 125, AND 135 MUST BE CHANGED TO DSQRT AND DABS.
C
C     ......................................................................
C
      RM=K+1
C
C        BETA WEIGHTS
C
      DO 100 J=1,K
  100 B(J)=0.0
      DO 110 J=1,K
      L1=K*(J-1)
      DO 110 I=1,K
      L=L1+I
  110 B(J)=B(J)+RY(I)*RX(L)
      RM=0.0
      BO=0.0
      L1=ISAVE(MM)
C
C        COEFFICIENT OF DETERMINATION
C
      DO 120 I=1,K
      RM=RM+B(I)*RY(I)
C
C        REGRESSION COEFFICIENTS
C
      L=ISAVE(I)
      B(I)=B(I)*(STD(L1)/STD(L))
C
C        INTERCEPT
C
  120 BO=BO+B(I)*XBAR(L)
      BO=XBAR(L1)-BO
C
C        SUM OF SQUARES ATTRIBUTABLE TO REGRESSION
C
      SSAR=RM*D(L1)
C
C        MULTIPLE CORRELATION COEFFICIENT
C
  122 RM= SQRT( ABS(RM))
C
C        SUM OF SQUARES OF DEVIATIONS FROM REGRESSION
C
      SSDR=D(L1)-SSAR
C
C        VARIANCE OF ESTIMATE
C
      FN=N-K-1
      SY=SSDR/FN
C
C        STANDARD DEVIATIONS OF REGRESSION COEFFICIENTS
C
      DO 130 J=1,K
      L1=K*(J-1)+J
      L=ISAVE(J)
  125 SB(J)= SQRT( ABS((RX(L1)/D(L))*SY))
C
C        COMPUTED T-VALUES
C
  130 T(J)=B(J)/SB(J)
C
C        STANDARD ERROR OF ESTIMATE
C
  135 SY= SQRT( ABS(SY))
C
C        F VALUE
C
      FK=K
      SSARM=SSAR/FK
      SSDRM=SSDR/FN
      F=SSARM/SSDRM
C
      ANS(1)=BO
      ANS(2)=RM
      ANS(3)=SY
      ANS(4)=SSAR
      ANS(5)=FK
      ANS(6)=SSARM
      ANS(7)=SSDR
      ANS(8)=FN
      ANS(9)=SSDRM
      ANS(10)=F
      RETURN
      END
```

Polynomial Regression

In the Scientific Subroutine Package, polynomial regression is normally performed by calling the following four subroutines in sequence:

1. GDATA - to generate the powers of the independent variable and find means, standard deviations, and correlation matrix
2. ORDER - to choose a dependent variable and subset of independent variables from a larger set of variables
3. MINV - to invert the correlation coefficient matrix
4. MULTR - to compute the regression coefficients $b_0, b_1, b_2, \ldots, b_m$, and various confidence measures

Subroutine GDATA

This subroutine generates independent variables up to the mth power (the highest degree polynomial specified) and calculates means, standard deviations, sums of cross-products of deviations from means, and product moment correlation coefficients.

X_{i1} denotes the ith case of the independent variable;

X_{ip} denotes the ith case of the dependent variable,

where $i = 1, 2, \ldots, n$

n = number of cases (observations)

$p = m + 1$

m = highest degree polynomial specified

The subroutine GDATA generates powers of the independent variable as follows:

$$
\begin{aligned}
X_{i2} &= X_{i1} \cdot X_{i1} \\
X_{i3} &= X_{i2} \cdot X_{i1} \\
X_{i4} &= X_{i3} \cdot X_{i1} \\
&\cdot \\
&\cdot \\
&\cdot \\
X_{im} &= X_{i,m-1} \cdot X_{i1}
\end{aligned}
\tag{1}
$$

where i and m are as defined as above.

Then, the following are calculated.

Means:

$$\overline{X}_j = \frac{\sum_{i=1}^{n} X_{ij}}{n} \tag{2}$$

where $j = 1, 2, \ldots, p$

Sums of cross-products of deviations from means:

$$D_{jk} = \sum_{i=1}^{n}\left(X_{ij} - \overline{X}_j\right)\left(X_{ik} - \overline{X}_k\right) - \frac{\sum_{i=1}^{n}\left(X_{ij} - \overline{X}_j\right)\sum_{i=1}^{n}\left(X_{ik} - \overline{X}_k\right)}{n} \tag{3}$$

where $j = 1, 2, \ldots, p$; $k = 1, 2, \ldots, p$.

Correlation coefficients:

$$r_{ij} = \frac{D_{ij}}{\sqrt{D_{ii}}\ \sqrt{D_{jj}}} \tag{4}$$

where $i = 1, 2, \ldots, p$; $j = 1, 2, \ldots, p$.

Standard deviations:

$$s_j = \frac{\sqrt{D_{jj}}}{\sqrt{n-1}} \tag{5}$$

where $j = 1, 2, \ldots, p$

```
C                                                                       GDAT  10
C     ..................................................................GDAT  20
C                                                                       GDAT  30
C        SUBROUTINE GDATA                                               GDAT  40
C                                                                       GDAT  50
C        PURPOSE                                                        GDAT  60
C           GENERATE INDEPENDENT VARIABLES UP TO THE M-TH POWER (THE    GDAT  70
C           HIGHEST DEGREE POLYNOMIAL SPECIFIED) AND COMPUTE MEANS,     GDAT  80
C           STANDARD DEVIATIONS, AND CORRELATION COEFFICIENTS.  THIS    GDAT  90
C           SUBROUTINE IS NORMALLY CALLED BEFORE SUBROUTINES ORDER,     GDAT 100
C           MINV AND MULTR IN THE PERFORMANCE OF A POLYNOMIAL           GDAT 110
C           REGRESSION.                                                 GDAT 120
C                                                                       GDAT 130
C        USAGE                                                          GDAT 140
C           CALL GDATA (N,M,X,XBAR,STD,D,SUMSQ)                         GDAT 150
C                                                                       GDAT 160
C        DESCRIPTION OF PARAMETERS                                      GDAT 170
C           N      - NUMBER OF OBSERVATIONS.                            GDAT 180
C           M      - THE HIGHEST DEGREE POLYNOMIAL TO BE FITTED.        GDAT 190
C           X      - INPUT MATRIX (N BY M+1) .  WHEN THE SUBROUTINE IS  GDAT 200
C                    CALLED, DATA FOR THE INDEPENDENT VARIABLE ARE      GDAT 210
C                    STORED IN THE FIRST COLUMN OF MATRIX X, AND DATA FORGDAT 220
C                    THE DEPENDENT VARIABLE ARE STORED IN THE LAST      GDAT 230
C                    COLUMN OF THE MATRIX.  UPON RETURNING TO THE       GDAT 240
C                    CALLING ROUTINE, GENERATED POWERS OF THE INDEPENDENTGDAT 250
C                    VARIABLE ARE STORED IN COLUMNS 2 THROUGH M.        GDAT 260
C           XBAR   - OUTPUT VECTOR OF LENGTH M+1 CONTAINING MEANS OF    GDAT 270
C                    INDEPENDENT AND DEPENDENT VARIABLES.               GDAT 280
C           STD    - OUTPUT VECTOR OF LENGTH M+1 CONTAINING STANDARD    GDAT 290
C                    DEVIATIONS OF INDEPENDENT AND DEPENDENT VARIABLES. GDAT 300
C           D      - OUTPUT MATRIX (ONLY UPPER TRIANGULAR PORTION OF THE GDAT 310
C                    SYMMETRIC MATRIX OF M+1 BY M+1) CONTAINING CORRELA- GDAT 320
C                    TION COEFFICIENTS.  (STORAGE MODE OF 1)            GDAT 330
C           SUMSQ  - OUTPUT VECTOR OF LENGTH M+1 CONTAINING SUMS OF     GDAT 340
C                    PRODUCTS OF DEVIATIONS FROM MEANS  OF INDEPENDENT  GDAT 350
C                    AND DEPENDENT VARIABLES.                           GDAT 360
C                                                                       GDAT 370
C        REMARKS                                                        GDAT 380
C           N MUST BE GREATER THAN M+1.                                 GDAT 390
C           IF M IS EQUAL TO 5 OR GREATER, SINGLE PRECISION MAY NOT BE  GDAT 400
C           SUFFICIENT TO GIVE SATISFACTORY COMPUTATIONAL RESULTS.      GDAT 410
```

```
C                                                                       GDAT 420
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   GDAT 430
C          NONE                                                         GDAT 440
C                                                                       GDAT 450
C       METHOD                                                          GDAT 460
C          REFER TO B. OSTLE, 'STATISTICS IN RESEARCH', THE IOWA STATE  GDAT 470
C          COLLEGE PRESS, 1954, CHAPTER 6.                              GDAT 480
C                                                                       GDAT 490
C     ..................................................................GDAT 500
C                                                                       GDAT 510
      SUBROUTINE GDATA (N,M,X,XBAR,STD,D,SUMSQ)                         GDAT 520
      DIMENSION X(1),XBAR(1),STD(1),D(1),SUMSQ(1)                       GDAT 530
C                                                                       GDAT 540
C        ................................................................GDAT 550
C                                                                       GDAT 560
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  GDAT 570
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      GDAT 580
C        STATEMENT WHICH FOLLOWS.                                       GDAT 590
C                                                                       GDAT 600
C     DOUBLE PRECISION X,XBAR,STD,D,SUMSQ,T1,T2                         GDAT 610
C                                                                       GDAT 620
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    GDAT 630
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      GDAT 640
C        ROUTINE.                                                       GDAT 650
C                                                                       GDAT 660
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      GDAT 670
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT AND ABS IN   GDAT 680
C        STATEMENT 180 MUST BE CHANGED TO DSQRT AND DABS.               GDAT 690
C                                                                       GDAT 700
C        ................................................................GDAT 710
C                                                                       GDAT 720
C     GENERATE INDEPENDENT VARIABLES                                    GDAT 730
C                                                                       GDAT 740
      IF(M-1) 105, 105, 90                                              GDAT 750
   90 L1=0                                                              GDAT 760
      DO 100 I=2,M                                                      GDAT 770
      L1=L1+N                                                           GDAT 780
      DO 100 J=1,N                                                      GDAT 790
      L=L1+J                                                            GDAT 800
      K=L-N                                                             GDAT 810
  100 X(L)=X(K)*X(J)                                                    GDAT 820
C                                                                       GDAT 830
C     CALCULATE MEANS                                                   GDAT 840
C                                                                       GDAT 850
  105 MM=M+1                                                            GDAT 860
      DF=N                                                              GDAT 870
      L=0                                                               GDAT 880
      DO 115 I=1,MM                                                     GDAT 890
      XBAR(I)=0.0                                                       GDAT 900
      DO 110 J=1,N                                                      GDAT 910
      L=L+1                                                             GDAT 920
  110 XBAR(I)=XBAR(I)+X(L)                                              GDAT 930
  115 XBAR(I)=XBAR(I)/DF                                                GDAT 940
C                                                                       GDAT 950
      DO 130 I=1,MM                                                     GDAT 960
  130 STD(I)=0.0                                                        GDAT 970
C                                                                       GDAT 980
C     CALCULATE SUMS OF CROSS-PRODUCTS OF DEVIATIONS                    GDAT 990
C                                                                       GDAT1000
      L=((MM+1)*MM)/2                                                   GDAT1010
      DO 150 I=1,L                                                      GDAT1020
  150 D(I)=0.0                                                          GDAT1030
      DO 170 K=1,N                                                      GDAT1040
      L=0                                                               GDAT1050
      DO 170 J=1,MM                                                     GDAT1060
      L2=N*(J-1)+K                                                      GDAT1070
      T2=X(L2)-XBAR(J)                                                  GDAT1080
      STD(J)=STD(J)+T2                                                  GDAT1090
      DO 170 I=1,J                                                      GDAT1100
      L1=N*(I-1)+K                                                      GDAT1110
      T1=X(L1)-XBAR(I)                                                  GDAT1120
      L=L+1                                                             GDAT1130
  170 D(L)=D(L)+T1*T2                                                   GDAT1140
      L=0                                                               GDAT1150
      DO 175 J=1,MM                                                     GDAT1160
      DO 175 I=1,J                                                      GDAT1170
      L=L+1                                                             GDAT1180
  175 D(L)=D(L)-STD(I)*STD(J)/DF                                        GDAT1190
      L=0                                                               GDAT1200
      DO 180 I=1,MM                                                     GDAT1210
      L=L+I                                                             GDAT1220
      SUMSQ(I)=D(L)                                                     GDAT1230
  180 STD(I)= SQRT( ABS(D(L)))                                          GDAT1240
C                                                                       GDAT1250
C     CALCULATE CORRELATION COEFFICIENTS                                GDAT1260
C                                                                       GDAT1270
      L=0                                                               GDAT1280
      DO 190 J=1,MM                                                     GDAT1290
      DO 190 I=1,J                                                      GDAT1300
      L=L+1                                                             GDAT1310
  190 D(L)=D(L)/(STD(I)*STD(J))                                         GDAT1320
C                                                                       GDAT1330
C     CALCULATE STANDARD DEVIATIONS                                     GDAT1340
C                                                                       GDAT1350
      DF=SQRT(DF-1.0)                                                   GDAT1360
      DO 200 I=1,MM                                                     GDAT1370
  200 STD(I)=STD(I)/DF                                                  GDAT1380
      RETURN                                                            GDAT1390
      END                                                               GDAT1400
```

Stepwise Multiple Regression

In the Scientific Subroutine Package, stepwise multiple regression is normally performed by calling the following subroutines:

1. CORRE - to find means, standard deviations, sums of cross-products of deviation matrix, and correlation matrix
2. MSTR - to save the matrix of sums of cross-products of deviations and to copy it to a working matrix for the purpose of using the same data repeatedly
3. LOC - to compute a vector subscript for an element in general and symmetric matrices. This subroutine is called by the subroutine MSTR
4. STPRG - to compute regression coefficients and other statistics for each step of regression

The subroutine CORRE works in either of two ways: (1) it expects all observations in core, or (2) it triggers a user-provided input subroutine, DATA, to read one observation at a time into a wor area. In either case, the user must provide a subroutine named DATA (see "Subroutines and Functioı Subprograms Required" in the comment cards description of the subroutine CORRE).

The subroutine STPRG calls an output subroutine STOUT, in order to print various statistics computed for each step of regression. The subroutine STOUT must also be provided by the user.

Subroutine STPRG

This subroutine performs a stepwise multiple regression analysis for a dependent variable and a set of independent variables. In each step of the regression $i=1,2,\ldots,q$, where q is the number of independent variables, the abbreviated Doolittle method is used to calculate the following statistics:

The independent variable entering in the regression is selected, first, by computing the amount of reduction of sum of squares for each variable:

$$C_j = \frac{a_{jy}^2}{a_{jj}} \tag{1}$$

where a_{jj} is initially an element in the sums of cross-products of deviations matrix which will be modified in successive steps.

$j =$ 1, 2, . . . , q are independent variables ($j \neq$ variables deleted and variables entered before the ith step)

$y =$ dependent variable

and, second, by finding the maximum (over j) of C.

Set $S_i = C_j$ to indicate the sum of squares that will be reduced in the ith step.

The proportion of S_i to the total is obtained by:

$$p = \frac{S_i}{D} \tag{2}$$

$$\text{where } D = \sum_{j=1}^{n} (y_j - \bar{y})^2$$

n = number of observations

If p is less than the constant specified by the user to limit independent variables, the analysis will be terminated without entering the last variable selected; otherwise, the following calculations are continued:

The cumulative sum of squares reduced is obtained by

$$S_{cum} = S_{cum} + S_i \tag{3}$$

and the cumulative proportion reduced by

$$P_{cum} = P_{cum} + p \tag{4}$$

The multiple correlation coefficient is computed by

$$R = \sqrt{P_{cum}} \tag{5}$$

and adjusted for degrees of freedom by

$$R_c = \sqrt{1 - (1-R^2) \quad (n-1) / (n-k)}$$

where there are k independent variables in the regression.

The F-value for analysis of variance is given by

$$F = \frac{S_{cum} / k}{(D-S_{cum}) / (n-k-1)} \tag{6}$$

The standard error of the estimated y is obtained by the use of the formula

$$s_{y.12\ldots i} = \sqrt{\frac{D - S_{cum}}{n - k - 1}} \tag{7}$$

and adjusted by

$$s_c = s \sqrt{(n-1) / (n-k)}$$

Then, the following is computed:

$$a_{jj} = a_{jj} + \frac{a_{ij}^2}{a_{ii}} \tag{8}$$

where i = variable entered in the ith step

j = $v_1, v_2, \ldots, v_{i-1}$ are the variables entered in the regression before the ith step, and

$$g_{ik} = \frac{a_{ik}}{a_{ii}} \tag{9}$$

where k = 1, 2, . . . , m are variables including y ($k \neq$ variables deleted and the variable entered in the ith step).

Regression coefficients are computed by:

$$\begin{aligned} b_i &= g_{iy} \\ b_{i-1} &= g_{(i-1)y} - b_i g_{(i-1)i} \\ b_{i-2} &= g_{(i-2)y} - b_i g_{(i-2)i} - b_{i-1} g_{(i-2)(i-1)} \\ &\text{etc.,} \end{aligned} \tag{10}$$

and the value of the intercept as

$$b_0 = \bar{y} - \sum_{j=1}^{k} b_j \bar{x}_j \tag{11}$$

where k = number of independent variables in the regression.

Standard errors of regression coefficients are given by

$$s_{b_j} = \sqrt{a_{jj}}\, s_{y.12\ldots i} \tag{12}$$

where j = $v_1, v_2, \ldots, v_i$ are variables in the regression, and t-values as

$$t_j = \frac{b_j}{s_{b_j}} \tag{13}$$

Perform the reduction to eliminate the variable entered in the ith step:

$$a_{jk} = a_{jk} - a_{ji} g_{ik} \tag{14}$$

where i = variable entered in the ith step

j = 1,2,...,m (j ≠ variables deleted or variables in the regression)

k = 1,2,...,m (k ≠ variables deleted and the variable entered in the ith step)

$$a_{ji} = \frac{a_{ji}}{-a_{ii}} \tag{15}$$

$$a_{ii} = \frac{1}{a_{ii}} \tag{16}$$

For reference see C. A. Bennett and N. L. Franklin, Statistical Analysis in Chemistry and the Chemical Industry. John Wiley and Sons, 1954, Appendix 6A.

```
C                                                                       STPR  10
C     ..................................................................STPR  20
C                                                                       STPR  30
C        SUBROUTINE STPRG                                               STPR  40
C                                                                       STPR  50
C        PURPOSE                                                        STPR  60
C           TO PERFORM A STEPWISE MULTIPLE REGRESSION ANALYSIS FOR A    STPR  70
C           DEPENDENT VARIABLE AND A SET OF INDEPENDENT VARIABLES.  AT  STPR  80
C           EACH STEP, THE VARIABLE ENTERED INTO THE REGRESSION EQUATIONSTPR  90
C           IS THAT WHICH EXPLAINS THE GREATEST AMOUNT OF VARIANCE      STPR 100
C           BETWEEN IT AND THE DEPENDENT VARIABLE (I.E. THE VARIABLE    STPR 110
C           WITH THE HIGHEST PARTIAL CORRELATION WITH THE DEPENDENT     STPR 120
C           VARIABLE). ANY VARIABLE CAN BE DESIGNATED AS THE DEPENDENT  STPR 130
C           VARIABLE.  ANY INDEPENDENT VARIABLE CAN BE FORCED INTO OR   STPR 140
C           DELETED FROM THE REGRESSION EQUATION, IRRESPECTIVE OF ITS   STPR 150
C           CONTRIBUTION TO THE EQUATION.                               STPR 160
C                                                                       STPR 170
C        USAGE                                                          STPR 180
C           CALL STPRG (M,N,C,XBAR,ICX,PCT,NSTEP,ANS,L,B,S,T,LL,IER)    STPR 190
C                                                                       STPR 200
C        DESCRIPTION OF PARAMETERS                                      STPR 210
C           M    - TOTAL NUMBER OF VARIABLES IN DATA MATRIX             STPR 220
C           N    - NUMBER OF OBSERVATIONS                               STPR 230
C           C    - INPUT MATRIX (M X M) OF SUMS OF CROSS-PRODUCTS OF    STPR 240
C                  DEVIATIONS FROM MEAN.  THIS MATRIX WILL BE DESTROYED.STPR 250
C           XBAR - INPUT VECTOR OF LENGTH M OF MEANS                    STPR 260
C           ICX  - INPUT VECTOR OF LENGTH M HAVING ONE OF THE FOLLOWING STPR 270
C                  CODES FOR EACH VARIABLE.                             STPR 280
C                    0 - INDEPENDENT VARIABLE AVAILABLE FOR SELECTION   STPR 290
C                    1 - INDEPENDENT VARIABLE TO BE FORCED INTO THE     STPR 300
C                        REGRESSION EQUATION                            STPR 310
C                    2 - VARIABLE NOT TO BE CONSIDERED IN THE EQUATION  STPR 320
C                    3 - DEPENDENT VARIABLE                             STPR 330
C                  THIS VECTOR WILL BE DESTROYED                        STPR 340
C           PCT  - A CONSTANT VALUE INDICATING THE PROPORTION OF THE    STPR 350
C                  TOTAL VARIANCE TO BE EXPLAINED BY ANY INDEPENDENT    STPR 360
C                  VARIABLE.  THOSE INDEPENDENT VARIABLES WHICH FALL    STPR 370
C                  BELOW THIS PROPORTION WILL NOT ENTER THE REGRESSION  STPR 380
C                  EQUATION.  TO ENSURE THAT ALL VARIABLES ENTER THE    STPR 390
C                  EQUATION, SET PCT = 0.0.                             STPR 400
C           NSTEP- OUTPUT VECTOR OF LENGTH 5 CONTAINING THE FOLLOWING   STPR 410
C                  INFORMATION                                          STPR 420
C                     NSTEP(1)- THE NUMBER OF THE DEPENDENT VARIABLE    STPR 430
C                     NSTEP(2)- NUMBER OF VARIABLES FORCED INTO THE     STPR 440
C                               REGRESSION EQUATION                     STPR 450
C                     NSTEP(3)- NUMBER OF VARIABLE DELETED FROM THE     STPR 460
C                               EQUATION                                STPR 470
C                     NSTEP(4)- THE NUMBER OF THE LAST STEP             STPR 480
C                     NSTEP(5)- THE NUMBER OF THE LAST VARIABLE ENTERED STPR 490
C           ANS  - OUTPUT VECTOR OF LENGTH 11 CONTAINING THE FOLLOWING  STPR 500
C                  INFORMATION FOR THE LAST STEP                        STPR 510
C                     ANS(1)- SUM OF SQUARES REDUCED BY THIS STEP       STPR 520
C                     ANS(2)- PROPORTION OF TOTAL SUM OF SQUARES REDUCEDSTPR 530
C                     ANS(3)- CUMULATIVE SUM OF SQUARES REDUCED UP TO   STPR 540
C                             THIS STEP                                 STPR 550
C                     ANS(4)- CUMULATIVE PROPORTION OF TOTAL SUM OF     STPR 560
C                             SQUARES REDUCED                           STPR 570
C                     ANS(5)- SUM OF SQUARES OF THE DEPENDENT VARIABLE  STPR 580
C                     ANS(6)- MULTIPLE CORRELATION COEFFICIENT          STPR 590
C                     ANS(7)- F RATIO FOR SUM OF SQUARES DUE TO         STPR 600
C                             REGRESSION                                STPR 610
C                     ANS(8)- STANDARD ERROR OF THE ESTIMATE (RESIDUAL  STPR 620
C                             MEAN SQUARE)                              STPR 630
C                     ANS(9)- INTERCEPT CONSTANT                        STPR 640
C                     ANS(10)-MULTIPLE CORRELATION COEFFICIENT ADJUSTED STPR 650
C                             FOR DEGREES OF FREEDOM.                   STPR 660
C                     ANS(11)-STANDARD ERROR OF THE ESTIMATE ADJUSTED   STPR 670
C                             FOR DEGREES OF FREEDOM.                   STPR 680
C           L    - OUTPUT VECTOR OF LENGTH K, WHERE K IS THE NUMBER OF  STPR 690
C                  INDEPENDENT VARIABLES IN THE REGRESSION EQUATION.    STPR 700
C                  THIS VECTOR CONTAINS THE NUMBERS OF THE INDEPENDENT  STPR 710
C                  VARIABLES IN THE EQUATION.                           STPR 720
C           B    - OUTPUT VECTOR OF LENGTH K, CONTAINING THE PARTIAL    STPR 730
C                  REGRESSION COEFFICIENTS CORRESPONDING TO THE         STPR 740
C                  VARIABLES IN VECTOR L.                               STPR 750
C           S    - OUTPUT VECTOR OF LENGTH K, CONTAINING THE STANDARD   STPR 760
C                  ERRORS OF THE PARTIAL REGRESSION COEFFICIENTS,       STPR 770
C                  CORRESPONDING TO THE VARIABLES IN VECTOR L.          STPR 780
C           T    - OUTPUT VECTOR OF LENGTH K, CONTAINING THE COMPUTED   STPR 790
C                  T-VALUES CORRESPONDING TO THE VARIABLES IN VECTOR L. STPR 800
C           LL   - WORKING VECTOR OF LENGTH M                           STPR 810
C           IER  - 0, IF THERE IS NO ERROR.                             STPR 820
C                  1, IF RESIDUAL SUM OF SQUARES IS NEGATIVE OR IF THE  STPR 830
C                  PIVOTAL ELEMENT IN THE STEPWISE INVERSION PROCESS IS STPR 840
C                  ZERO.  IN THIS CASE, THE VARIABLE WHICH CAUSES THIS  STPR 850
C                  ERROR IS NOT ENTERED IN THE REGRESSION, THE RESULT   STPR 860
C                  PRIOR TO THIS STEP IS RETAINED, AND THE CURRENT      STPR 870
C                  SELECTION IS TERMINATED.                             STPR 880
C                                                                       STPR 890
C        REMARKS                                                        STPR 900
C           THE NUMBER OF DATA POINTS MUST BE AT LEAST GREATER THAN THE STPR 910
C           NUMBER OF INDEPENDENT VARIABLES PLUS ONE.  FORCED VARIABLES STPR 920
C           ARE ENTERED INTO THE REGRESSION EQUATION BEFORE ALL OTHER   STPR 930
C           INDEPENDENT VARIABLES.  WITHIN THE SET OF FORCED VARIABLES, STPR 940
C           THE ONE TO BE CHOSEN FIRST WILL BE THAT ONE WHICH EXPLAINS  STPR 950
C           THE GREATEST AMOUNT OF VARIANCE.                            STPR 960
C           INSTEAD OF USING, AS A STOPPING CRITERION, A PROPORTION OF  STPR 970
C           THE TOTAL VARIANCE, SOME OTHER CRITERION MAY BE ADDED TO    STPR 980
C           SUBROUTINE STOLT.                                           STPR 990
C                                                                       STPR1000
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  STPR1010
C           STOLT(NSTEP,ANS,L,B,S,T,NSTOP)                              STPR1020
C           THIS SUBROUTINE MUST BE PROVIDED BY THE USER.  IT IS AN     STPR1030
C           OUTPUT ROUTINE WHICH WILL PRINT THE RESULTS OF EACH STEP OF STPR1040
C           THE REGRESSION ANALYSIS.  NSTOP IS AN OPTION CODE WHICH IS  STPR1050
C           ONE IF THE STEPWISE REGRESSION IS TO BE TERMINATED, AND IS  STPR1060
C           ZERO IF IT IS TO CONTINUE.  THE USER MUST CONSIDER THIS IF  STPR1070
C           SOME OTHER STOPPING CRITERION THAN VARIANCE PROPORTION IS TOSTPR1080
C           BE USED.                                                    STPR1090
C                                                                       STPR1100
C        METHOD                                                         STPR1110
C           THE ABBREVIATED DOOLITTLE METHOD IS USED TO (1) DECIDE VARI-STPR1120
C           ABLES ENTERING IN THE REGRESSION AND (2) COMPUTE REGRESSION STPR1130
C           COEFFICIENTS.  REFER TO C. A. BENNETT AND N. L. FRANKLIN,   STPR1140
C           'STATISTICAL ANALYSIS IN CHEMISTRY AND THE CHEMICAL INDUS-  STPR1150
C           TRY', JOHN WILEY AND SONS, 1954, APPENDIX 6A.               STPR1160
C                                                                       STPR1170
C     ..................................................................STPR1180
C                                                                       STPR1190
      SUBROUTINE STPRG (M,N,C,XBAR,ICX,PCT,NSTEP,ANS,L,B,S,T,LL,IER)    STPR1200
C                                                                       STPR1210
      DIMENSION C(1),XBAR(1),ICX(1),NSTEP(1),ANS(1),L(1),B(1),S(1),T(1),STPR1220
     1LL(1)                                                             STPR1230
C                                                                       STPR1240
C     ..................................................................STPR1250
C                                                                       STPR1260
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  STPR1270
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      STPR1280
C        STATEMENT WHICH FOLLOWS.                                       STPR1290
C                                                                       STPR1300
C     DOUBLE PRECISION D,XBAR,ANS,B,S,T,RD,RE                           STPR1310
C                                                                       STPR1320
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    STPR1330
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      STPR1340
C        ROUTINE.                                                       STPR1350
C                                                                       STPR1360
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      STPR1370
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENTSSTPR1380
C        85,90,114,132,AND 134, MUST BE CHANGED TO DSQRT.               STPR1390
C                                                                       STPR1400
C     ..................................................................STPR1410
C                                                                       STPR1420
C        INITIALIZATION                                                 STPR1430
C                                                                       STPR1440
      IER=0                                                             STPR1450
      KNP=N-1                                                           STPR1460
      NFC=0                                                             STPR1470
      NSTEP(3)=0                                                        STPR1480
      ANS(3)=0.0                                                        STPR1490
      ANS(4)=0.0                                                        STPR1500
      NSTOP=0                                                           STPR1510
C                                                                       STPR1520
C        FIND DEPENDENT VARIABLE, NUMBER OF VARIABLES TO BE FORCED TO   STPR1530
C        ENTER IN THE REGRESSION, AND NUMBER OF VARIABLES TO BE DELETED STPR1540
C                                                                       STPR1550
      DO 30 I=1,M                                                       STPR1560
      LL(I)=1                                                           STPR1570
      IF(ICX(I)) 30, 30, 10                                             STPR1580
   10 IF(ICX(I)-2) 15, 20, 25                                           STPR1590
   15 NFC=NFC+1                                                         STPR1600
      ICX(NFC)=I                                                        STPR1610
      GO TO 30                                                          STPR1620
   20 NSTEP(3)=NSTEP(3)+1                                               STPR1630
      LL(I)=-1                                                          STPR1640
      GO TO 30                                                          STPR1650
   25 MY=I                                                              STPR1660
```

```
      NSTEP(1)=PY                                                       STPR1670
      LY=P*(PY-1)                                                       STPR1680
      LYP=LY+PY                                                         STPR1690
      ANS(5)=C(LYP)                                                     STPR1700
   3C CONTINUE                                                          STPR1710
      NSTEP(2)=NFC                                                      STPR1720
C                                                                       STPR1730
C        FIND THE MAXIMUM NUMBER OF STEPS                               STPR1740
C                                                                       STPR1750
      MX=P-NSTEP(3)-1                                                   STPR1760
C                                                                       STPR1770
C        START SELECTION OF VARIABLES                                   STPR1780
C                                                                       STPR1790
      DO 14C NL=1,MX                                                    STPR1800
      RC=C                                                              STPR1810
      IF(NL-NFC) 35, 35, 55                                             STPR1820
C                                                                       STPR1830
C        SELECT NEXT VARIABLE TO ENTER AMONG FORCED VARIABLES           STPR1840
C                                                                       STPR1850
   35 DO 5C I=1,NFC                                                     STPR1860
      K=IDX(I)                                                          STPR1870
      IF(LL(K)) 5C, 5C, 4C                                              STPR1880
   4C LYP=LY+K                                                          STPR1890
      IP=P*(K-1)+K                                                      STPR1900
      RE=C(LYP)*C(LYP)/C(IP)                                            STPR1910
      IF(RC-RE) 45, 5C, 5C                                              STPR1920
   45 RC=RE                                                             STPR1930
      NEW=K                                                             STPR1940
   5C CONTINUE                                                          STPR1950
      GO TO 75                                                          STPR1960
C                                                                       STPR1970
C        SELECT NEXT VARIABLE TO ENTER AMONG NON-FORCED VARIABLES       STPR1980
C                                                                       STPR1990
   55 DO 7C I=1,P                                                       STPR2000
      IF(I-PY) 6C, 7C, 6C                                               STPR2010
   6C IF(LL(I)) 7C, 7C, 62                                              STPR2020
   62 LYP=LY+I                                                          STPR2030
      IP=P*(I-1)+I                                                      STPR2040
      RE=C(LYP)*C(LYP)/C(IP)                                            STPR2050
      IF(RC-RE) 64, 7C, 7C                                              STPR2060
   64 RC=RE                                                             STPR2070
      NEW=I                                                             STPR2080
   7C CONTINUE                                                          STPR2090
C                                                                       STPR2100
C        TEST WHETHER THE PROPORTION OF THE SUM OF SQUARES REDUCED BY   STPR2110
C        THE LAST VARIABLE ENTERED IS GREATER THAN OR EQUAL TO THE      STPR2120
C        SPECIFIED PROPORTION                                           STPR2130
C                                                                       STPR2140
   75 IF(RC) 77,77,76                                                   STPR2150
   76 IF(ANS(5)-(ANS(3)+RC))77,77,78                                    STPR2160
   77 IER=1                                                             STPR2170
      GO TO 15C                                                         STPR2180
   78 RE=RC/ANS(5)                                                      STPR2190
      IF(RE-FCT) 15C, 8C, 8C                                            STPR2200
C                                                                       STPR2210
C        IT IS GREATER THAN OR EQUAL                                    STPR2220
C                                                                       STPR2230
   8C LL(NEW)=C                                                         STPR2240
      L(NL)=NEW                                                         STPR2250
      ANS(1)=RC                                                         STPR2260
      ANS(2)=RE                                                         STPR2270
      ANS(3)=ANS(3)+RC                                                  STPR2280
      ANS(4)=ANS(4)+RE                                                  STPR2290
      NSTEP(4)=NL                                                       STPR2300
      NSTEP(5)=NEW                                                      STPR2310
C                                                                       STPR2320
C        COMPUTE MULTIPLE CORRELATION, F-VALUE FOR ANALYSIS OF          STPR2330
C        VARIANCE, AND STANDARD ERROR OF ESTIMATE                       STPR2340
C                                                                       STPR2350
   85 ANS(6)= SQRT(ANS(4))                                              STPR2360
      RC=NL                                                             STPR2370
      RE=CAP-RC                                                         STPR2380
      RE=(ANS(5)-ANS(3))/RE                                             STPR2390
      ANS(7)=(ANS(3)/RC)/RE                                             STPR2400
   9C ANS(8)= SQRT(RE)                                                  STPR2410
C                                                                       STPR2420
C        DIVIDE BY THE PIVOTAL ELEMENT                                  STPR2430
C                                                                       STPR2440
      IP=P*(NEW-1)+NEW                                                  STPR2450
      RC=C(IP)                                                          STPR2460
      LYP=NEW-P                                                         STPR2470
      DO 1CC J=1,P                                                      STPR2480
      LYP=LYP+P                                                         STPR2490
      IF(LL(J)) 1CC, 94, 97                                             STPR2500
   94 IF(J-NEW) 96, 98, 96                                              STPR2510
   96 IJ=P*(J-1)+J                                                      STPR2520
      C(IJ)=C(IJ)+C(LYP)*C(LYP)/RC                                      STPR2530
   97 C(LYP)=C(LYP)/RC                                                  STPR2540
      GO TO 1CC                                                         STPR2550
   98 C(IP)=1.C/RC                                                      STPR2560
  1CC CONTINUE                                                          STPR2570
C                                                                       STPR2580
C        COMPUTE REGRESSION COEFFICIENTS                                STPR2590
C                                                                       STPR2600
      LYP=LY+NEW                                                        STPR2610
      E(NL)=C(LYP)                                                      STPR2620
      IF(NL-1) 112, 112, 1C5                                            STPR2630
  1C5 ID=NL-1                                                           STPR2640
      DO 11C J=1,ID                                                     STPR2650
      IJ=NL-J                                                           STPR2660
      KK=L(IJ)                                                          STPR2670
      LYP=LY+KK                                                         STPR2680
      E(IJ)=C(LYP)                                                      STPR2690
      DO 11C K=1,J                                                      STPR2700
      IK=NL-K+1                                                         STPR2710
      KK=L(IK)                                                          STPR2720
      LYP=P*(KK-1)+KK                                                   STPR2730
  11C E(IJ)=E(IJ)-C(LYP)*E(IK)                                          STPR2740
C                                                                       STPR2750
C        COMPUTE INTERCEPT                                              STPR2760
C                                                                       STPR2770
  112 ANS(9)=XBAR(PY)                                                   STPR2780
      DO 115 I=1,NL                                                     STPR2790
      KK=L(I)                                                           STPR2800
      ANS(9)=ANS(9)-E(I)*XBAR(KK)                                       STPR2810
      IJ=P*(KK-1)+KK                                                    STPR2820
  114 S(I)=ANS(8)* SQRT(C(IJ))                                          STPR2830
  115 T(I)=E(I)/S(I)                                                    STPR2840
C                                                                       STPR2850
C        PERFORM A REDUCTION TO ELIMINATE THE LAST VARIABLE ENTERED     STPR2860
C                                                                       STPR2870
      IP=P*(NEW-1)                                                      STPR2880
      DO 13C I=1,P                                                      STPR2890
      IJ=I-P                                                            STPR2900
      IK=NEW-P                                                          STPR2910
      IP=IP+1                                                           STPR2920
      IF(LL(I)) 13C, 13C, 12C                                           STPR2930
  12C DO 126 J=1,P                                                      STPR2940
      IJ=IJ+P                                                           STPR2950
      IK=IK+P                                                           STPR2960
      IF(LL(J)) 126, 122, 122                                           STPR2970
  122 IF(J-NEW) 124, 126, 124                                           STPR2980
  124 C(IJ)=C(IJ)-C(IP)*C(IK)                                           STPR2990
  126 CONTINUE                                                          STPR3000
      C(IP)=C(IP)/(-RC)                                                 STPR3010
  13C CONTINUE                                                          STPR3020
C                                                                       STPR3030
C        ADJUST STANDARD ERROR OF THE ESTIMATE AND MULTIPLE CORRELATION STPR3040
C        COEFFICIENT                                                    STPR3050
C                                                                       STPR3060
      RC=N-NSTEP(4)                                                     STPR3070
      RC=CAP/RC                                                         STPR3080
  132 ANS(10)=SQRT(1.0-(1.0-ANS(6)*ANS(6))*RC)                          STPR3090
  134 ANS(11)=ANS(8)*SQRT(RC)                                           STPR3100
C                                                                       STPR3110
C        CALL THE OUTPUT SUBROUTINE                                     STPR3120
      CALL STOUT (NSTEP,ANS,L,B,S,T,NSTOP)                              STPR3130
C                                                                       STPR3140
C        TEST WHETHER THE STEP-WISE REGRESSION WAS TERMINATED IN        STPR3150
C        SUBROUTINE STOUT                                               STPR3160
C                                                                       STPR3170
      IF(NSTOP) 14C, 14C, 15C                                           STPR3180
C                                                                       STPR3190
  14C CONTINUE                                                          STPR3200
C                                                                       STPR3210
  15C RETURN                                                            STPR3220
      END                                                               STPR3230
```

ubroutine PROBT

This subroutine obtains maximum likelihood estimates of a and b for the probit equation $Y = a+bX$. An iterative scheme is used. The input to the subroutine consists of k different levels $X_1, X_2, \ldots, X_k$ of a stimulus applied to $N_1, N_2, \ldots, N_k$ objects with $R_1, R_2, \ldots, R_k$ responses respectively. The procedures described here assume that the distribution of critical values is normal. The computational steps are as follows:

1. Compute the proportions of objects responding to various levels of a stimulus:

$$p_i = \frac{R_i}{N_i} \tag{1}$$

where $i = 1, 2, \ldots, k$ are levels of a stimulus.

2. Compute the initial estimates of a and b for the probit equation, $Y = a+bX$, as follows:

$$b = \frac{\sum_{i=1}^{m} X_i Z_i - \frac{\left(\sum_{i=1}^{m} X_i\right)\left(\sum_{i=1}^{m} Z_i\right)}{m}}{\sum_{i=1}^{m} X^2_i - \frac{\left(\sum_{i=1}^{m} X_i\right)^2}{m}} \tag{2}$$

$$a = \overline{Z} - b\overline{X} \tag{3}$$

where X_i = ith level of a stimulus

Z_i = 5 plus the standard normal variable corresponding to p_i

m = number of p values which are not equal to 0.0 and 1.0

3. Compute values of expected probit using the initial estimates of a and b:

$$Y_i = a + bX_i \tag{4}$$

where $i = 1, 2, \ldots, k$

4. Corresponding to each value of Y, compute the following:

a. Weighting coefficient for probit analysis:

$$W_i = \frac{D^2_i}{P_i Q_i} \tag{5}$$

where D_i = density of the normal curve corresponding to the standard normal variable $Y_i - 5$

P_i = cumulative normal distribution value corresponding to $Y_i - 5$

$Q_i = 1 - P_i$

b. Working probit:

For $Y_i \leq 5.0$

$$y_i = Y_i - \frac{P_i}{D_i} + p_i \frac{1}{D_i} \tag{6}$$

For $Y_i > 5.0$

$$y_i = Y_i + \frac{Q_i}{D_i} - (1-p_i)\frac{1}{D_i} \tag{7}$$

5. Compute:

$$S_1 = \sum_{i=1}^{k} N_i W_i X^2_i - \frac{\left(\sum_{i=1}^{k} N_i W_i X_i\right)^2}{\sum_{i=1}^{k} N_i W_i} \tag{8}$$

$$S_2 = \sum_{i=1}^{k} N_i W_i X_i y_i - \frac{\left(\sum_{i=1}^{k} N_i W_i X_i\right)\left(\sum_{i=1}^{k} N_i W_i y_i\right)}{\sum_{i=1}^{k} N_i W_i} \tag{9}$$

$$S_3 = \sum_{i=1}^{k} N_i W_i y^2_i - \frac{\left(\sum_{i=1}^{k} N_i W_i y_i\right)^2}{\sum_{i=1}^{k} N_i W_i} \tag{10}$$

6. Compute new estimates of a and b for the probit equation:

$$b = \frac{S_2}{S_1} \tag{11}$$

$$a = \frac{\sum_{i=1}^{k} N_i W_i y_i}{\sum_{i=1}^{k} N_i W_i} - b \cdot \frac{\sum_{i=1}^{k} N_i W_i X_i}{\sum_{i=1}^{k} N_i W_i} \tag{12}$$

7. Compute new values of expected probit and compare them against the values computed in the previous iteration:

$$Y'_i = a + bX_i \tag{13}$$

$$DSQ = \sum_{i=1}^{k} (Y_i - Y')^2 \tag{14}$$

If the difference between two consecutive values of DSQ is less than or equal to 10^{-7}, go to step 8 and compute X^2. Otherwise, set $Y_i = Y_i'$, where $i = 1, 2, \ldots, k$, and go back to step 4 for another iteration.

8. Compute X^2 for the test of the final probit equation:

$$X^2 = \sum_{i=1}^{k} \frac{(R_i - N_i P_i)^2}{N_i P_i (1-P_i)} \tag{15}$$

where P_i = cumulative normal distribution value corresponding to z_i

$z_i = (a+bX_i) - 5$

a and b are computed for the final probit equation.

9. Degrees of freedom for X^2:

$$d.f. = k - 2 \tag{16}$$

For reference see:

(1) M. G. Natrella, Experimental Statistics - NBS Handbook 91. National Bureau of Standards, 1963, chapter 10.

(2) D. J. Finney, Probit Analysis (2nd ed.). Cambridge University Press, 1952.

```
C                                                                       PROB  10
C .......................................................................PROB  20
C                                                                       PROB  30
C     SUBROUTINE PROBT                                                  PROB  40
C                                                                       PROB  50
C     PURPOSE                                                           PROB  60
C        TO OBTAIN MAXIMUM LIKELIHOOD ESTIMATES FOR THE PARAMETERS A PROB  70
C        AND B IN THE PROBIT EQUATION  Y = A + BX.  AN ITERATIVE        PROB  80
C        SCHEME IS USED.  THE INPUT TO THE SUBROUTINE CONSISTS OF K    PROB  90
C        DIFFERENT DOSAGE LEVELS APPLIED TO K GROUPS OF SUBJECTS, ANDPROB 100
C        THE NUMBER OF SUBJECTS IN EACH GROUP RESPONDING TO THE        PROB 110
C        RESPECTIVE DOSAGE OF THE DRUG.                                PROB 120
C                                                                       PROB 130
C     USAGE                                                             PROB 140
C        CALL PROBT (K,X,S,R,LOG,ANS,W1,W2,IEF)                         PROB 150
C                                                                       PROB 160
C     DESCRIPTION OF PARAMETERS                                         PROB 170
C        K    - NUMBER OF DIFFERENT DOSE LEVELS OF THE DRUG.  K SHOULDPROB 180
C               BE GREATER THAN 2.                                      PROB 190
C        X    - INPUT VECTOR OF LENGTH K CONTAINING THE DOSE LEVEL OF PROB 200
C               THE DRUG TESTED.  X MUST BE NON-NEGATIVE.               PROB 210
C        S    - INPUT VECTOR OF LENGTH K CONTAINING THE NUMBER OF      PROB 220
C               SUBJECTS TESTED AT EACH DOSE LEVEL                      PROB 230
C        R    - INPUT VECTOR OF LENGTH K CONTAINING THE NUMBER OF      PROB 240
C               SUBJECTS AT EACH LEVEL RESPONDING TO THE DRUG           PROB 250
C                                                                       PROB 260
C        LOG  - INPUT OPTION CODE                                       PROB 270
C               1- IF IT IS DESIRED TO CONVERT THE DOSE LEVELS TO       PROB 280
C                  COMMON LOGARITHMS.  THE DOSAGE LEVELS SHOULD BE      PROB 290
C                  NON-NULL IN THIS CASE.                               PROB 300
C               0- IF NO CONVERSION IS DESIRED                          PROB 310
C        ANS  - OUTPUT VECTOR OF LENGTH 4 CONTAINING THE FOLLOWING      PROB 320
C               RESULTS                                                 PROB 330
C               ANS(1)- ESTIMATE OF THE INTERCEPT CONSTANT A            PROB 340
C               ANS(2)- ESTIMATE OF THE PROBIT REGRESSION COEFFICIENT   PROB 350
C                       B                                               PROB 360
C               ANS(3)- CHI-SQUARED VALUE FOR A TEST OF SIGNIFICANCE    PROB 370
C                       OF THE FINAL PROBIT EQUATION                    PROB 380
C               ANS(4)- DEGREES OF FREEDOM FOR THE CHI-SQUARE           PROB 390
C                       STATISTIC                                       PROB 400
C        W1   - OUTPUT VECTOR OF LENGTH K CONTAINING THE PROPORTIONS    PROB 410
C               OF SUBJECTS RESPONDING TO THE VARIOUS DOSE LEVELS OF    PROB 420
C               THE DRUG                                                PROB 430
C        W2   - OUTPUT VECTOR OF LENGTH K CONTAINING THE VALUES OF THEPROB 440
C               EXPECTED PROBIT FOR THE VARIOUS LEVELS OF A DRUG        PROB 450
C        IER  - 1 IF K IS NOT GREATER THAN 2.                           PROB 460
C               2 IF SOME DOSAGE LEVEL IS NEGATIVE, OR IF THE INPUT     PROB 470
C                 OPTION CODE LOG IS 1 AND SOME DOSAGE LEVEL IS ZERO.   PROB 480
C               3 IF SOME ELEMENT OF S IS NOT POSITIVE.                 PROB 490
C               4 IF NUMBER OF SUBJECTS RESPONDING IS GREATER THAN      PROB 500
C               NUMBER OF SUBJECTS TESTED.                              PROB 510
C               ONLY IF IER IS ZERO IS A PROBIT ANALYSIS PERFORMED.     PROB 520
C               OTHERWISE, ANS, W1, AND W2 ARE SET TO ZERO.             PROB 530
C                                                                       PROB 540
C     REMARKS                                                           PROB 550
C        THE PROGRAM WILL ITERATE ON THE PROBIT EQUATION UNTIL TWO      PROB 560
C        SUCCESSIVE SOLUTIONS PRODUCE CHANGES OF LESS THAN 10**(-7).    PROB 570
C                                                                       PROB 580
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     PROB 590
C        NDTF                                                           PROB 600
C        NDTRI                                                          PROB 610
C                                                                       PROB 620
C     METHOD                                                            PROB 630
C        REFER TO D. J. FINNEY, PROBIT ANALYSIS. 2ND ED. (CAMBRIDGE,    PROB 640
C        1952)                                                          PROB 650
C .......................................................................PROB 660
      SUBROUTINE PROBT (K,X,S,R,LOG,ANS,W1,W2,IER)                      PROB 670
C                                                                       PROB 680
      DIMENSION X(1),S(1),R(1),ANS(1),W1(1),W2(1)                       PROB 690
C                                                                       PROB 700
C        TEST WHETHER LOG CONVERSION IS NEEDED                          PROB 710
C                                                                       PROB 720
      IEF=0                                                             PROB 730
      IF(K-2)5,5,7                                                      PROB 740
    5 IER = 1                                                           PROB 750
      GO TO 90                                                          PROB 760
    7 DO 8 I=1,K                                                        PROB 770
      IF(X(I))12,8,8                                                    PROB 780
    8 CONTINUE                                                          PROB 790
      IF(LOG-1) 16,10,16                                                PROB 800
   10 DO 15 I=1,K                                                       PROB 810
      IF(X(I))12,12,14                                                  PROB 820
   12 IER=2                                                             PROB 830
      GO TO 90                                                          PROB 840
   14 X(I)= ALOG10(X(I))                                                PROB 850
   15 CONTINUE                                                          PROB 860
                                                                        PROB 870
```

```
C
C        COMPUTE PROPORTIONS OF OBJECTS RESPONDING
C
   16 DO 18 I=1,K
      IF(S(I)-R(I)) 17,18,18
   17 IER=4
      GO TO 90
   18 CONTINUE
   20 DO 23 I=1,K
      IF(S(I))21,21,22
   21 IER=3
      GO TO 90
   22 W1(I)=R(I)/S(I)
   23 CONTINUE
C
C        COMPUTE INITIAL ESTIMATES OF INTERCEPT AND PROBIT REGRESSION
C        COEFFICIENT
C
      WN=0.0
      XBAR=0.0
      SNWY=0.0
      SXX=0.0
      SXY=0.0
C
      DO 30 I=1,K
      P=W1(I)
      IF(P) 30, 30, 24
   24 IF(P-1.0) 25, 30, 30
   25 WN=WN+1.0
C
      CALL NDTRI (P,Z,D,IER)
C
      Z=Z+5.0
      XBAR=XBAR+X(I)
      SNWY=SNWY+Z
      SXX=SXX+X(I)**2
      SXY=SXY+X(I)*Z
   30 CONTINUE
C
      B=(SXY-(XBAR*SNWY)/WN)/(SXX-(XBAR*XBAR)/WN)
      XBAR=XBAR/WN
      SNWY=SNWY/WN
      A=SNWY-B*XBAR
      DD=0.0
C
C        COMPUTE EXPECTED PROBIT
C
      DO 31 I=1,K
   31 W2(I)=A+B*X(I)
C
   33 SNW=0.0
      SNWX=0.0
      SNWY=0.0
      SNWXX=0.0
      SNWXY=0.0
      DO 50 I=1,K
      Y=W2(I)
C
C        FIND A WEIGHTING COEFFICIENT FOR PROBIT ANALYSIS
C
      D=Y-5.0
C
      CALL NDTF (D,P,Z)
C
      Q=1.0-P
      W=(Z*Z)/(P*Q)
C
C        COMPUTE WORKING PROBIT
C
      IF(Y-5.0) 35, 35, 40
   35 WP=(Y-P/Z)+W1(I)/Z
      GO TO 45
   40 WP=(Y+Q/Z)-(1.0-W1(I))/Z
C
C        SUM INTERMEDIATE RESULTS
C
   45 WN=W*S(I)
      SNW=SNW+WN
      SNWX=SNWX+WN*X(I)
      SNWY=SNWY+WN*WP
      SNWXX=SNWXX+WN*X(I)**2
   50 SNWXY=SNWXY+WN*X(I)*WP
C
C        COMPUTE NEW ESTIMATES OF INTERCEPT AND COEFFICIENT
C
      XBAR=SNWX/SNW
C
      SXX=SNWXX-(SNWX)*(SNWX)/SNW
      SXY=SNWXY-(SNWX)*(SNWY)/SNW
      B=SXY/SXX
C
      A=SNWY/SNW-B*XBAR
C
C        EXAMINE THE CHANGES IN Y
C
      SXX=0.0
      DO 60 I=1,K
      Y=A+B*X(I)
      D=W2(I)-Y
      SXX=SXX+D*D
   60 W2(I)=Y
      IF(( ABS(DD-SXX))-(1.0E-7)) 65, 65, 63
   63 DD=SXX
      GO TO 33
C
C        STORE INTERCEPT AND COEFFICIENT
C
   65 ANS(1)=A
      ANS(2)=B
C
C        COMPUTE CHI-SQUARE
C
      ANS(3)=0.0
      DO 70 I=1,K
      Y=W2(I)-5.0
C
      CALL NDTR (Y,P,D)
C
      AA=P(I)-S(I)*P
      DD=S(I)*P*(1.0-P)
   70 ANS(3)=ANS(3)+AA*AA/DD
C
C        DEGREES OF FREEDOM FOR CHI-SQUARE
C
      ANS(4)=K-2
C
   80 RETURN
   90 DO 100 I=1,K
      W1(I)=0.0
  100 W2(I)=0.0
      DO 110 I=1,4
  110 ANS(I)=0.0
      GO TO 80
      END
```

Canonical Correlation

In the Scientific Subroutine Package, canonical correlation analysis is normally performed by calling the following five subroutines:

1. CORRE - to compute means, standard deviations, and correlation matrix
2. MINV - to invert a part of the correlation matrix
3. EIGEN - to compute eigenvalues and eigenvectors
4. NROOT - to compute eigenvalues and eigenvectors of real nonsymmetric matrix of the form $B^{-1} A$
5. CANOR - to compute canonical correlations and coefficients

The subroutine CORRE works in either of two ways: (1) it expects all observations in core, or (2) it triggers a user-provided input subroutine, DATA, to read one observation at a time into a work area. In either case, the user must provide a subroutine named DATA (see "Subroutines Required" in the comment cards description of subroutine CORRE).

Subroutine CANOR

This subroutine performs a canonical correlation analysis between two sets of variables.

The matrix of intercorrelations, R, is partitioned into four submatrices:

$$R = \left[\begin{array}{c|c} R_{11} & R_{12} \\ \hline R_{21} & R_{22} \end{array}\right] \qquad (1)$$

R_{11} = intercorrelations among p variables in the first set (that is, left-hand variables)

R_{12} = intercorrelations between the variables in the first and second sets

R_{21} = the transpose of R_{12}

R_{22} = intercorrelations among q variables in the second set (that is, right-hand variables)

The equation:

$$\left| R_{22}^{-1} R_{21} R_{11}^{-1} R_{12} - \lambda I \right| = 0 \qquad (2)$$

is then solved for all values of λ, eigenvalues, in the following matrix operation:

$$T = R_{11}^{-1} R_{12} \qquad (3)$$

$$A = R_{21} T \qquad (4)$$

The subroutine NROOT calculates eigenvalues (λ_i), with associated eigenvectors, of $R_{22}^{-1} A$, where $i = 1, 2, \ldots, q$.

For each subscript $i = 1, 2, \ldots, q$, the following statistics are calculated:

Canonical correlation:

$$CANR = \sqrt{\lambda_i} \qquad (5)$$

where λ_i = i^{th} eigenvalue

Chi-square:

$$\chi^2 = -\left[n - 0.5\,(p + q + 1)\right] \log_e \Lambda \qquad (6)$$

where n = number of observations

$$\Lambda = \prod_{j=i}^{q} (1 - \lambda_j);$$

Degrees of freedom for χ^2:

$$DF = \left[p - (i-1)\right]\left[q - (i-1)\right] \qquad (7)$$

i^{th} set of right-hand coefficients:

$$b_k = v_{ki} \qquad (8)$$

where v_{ki} = eigenvector associated with λ_i

$k = 1, 2, \ldots, q;$

i^{th} set of left-hand coefficients:

$$a_j = \frac{\sum_{k=1}^{q} t_{jk}\, b_k}{CANR} \qquad (9)$$

where $\left\{ t_{jk} \right\} = T = R_{11}^{-1} R_{12}$

$j = 1, 2, \ldots, p$

```
C                                                                       CANO  10
C     ..................................................................CANO  20
C                                                                       CANO  30
C        SUBROUTINE CANOR                                               CANO  40
C                                                                       CANO  50
C        PURPOSE                                                        CANO  60
C           COMPUTE THE CANONICAL CORRELATIONS BETWEEN TWO SETS OF      CANO  70
C           VARIABLES.  CANOR IS NORMALLY PRECEDED BY A CALL TO SUBROU- CANO  80
C           TINE CORRE.                                                 CANO  90
C                                                                       CANO 100
C        USAGE                                                          CANO 110
C           CALL CANOR (N,MP,MQ,RR,ROOTS,WLAM,CANR,CHISQ,NDF,COEFR,     CANO 120
C                      COEFL,R)                                         CANO 130
C                                                                       CANO 140
C        DESCRIPTION OF PARAMETERS                                      CANO 150
C           N     - NUMBER OF OBSERVATIONS                              CANO 160
C           MP    - NUMBER OF LEFT HAND VARIABLES                       CANO 170
C           MQ    - NUMBER OF RIGHT HAND VARIABLES                      CANO 180
C           RR    - INPUT MATRIX (ONLY UPPER TRIANGULAR PORTION OF THE  CANO 190
C                   SYMMETRIC MATRIX OF M X M, WHERE M = MP + MQ)       CANO 200
C                   CONTAINING CORRELATION COEFFICIENTS.  (STORAGE MODE CANO 210
C                   OF 1)                                               CANO 220
C           ROOTS - OUTPUT VECTOR OF LENGTH MQ CONTAINING EIGENVALUES   CANO 230
C                   COMPUTED IN THE NROOT SUBROUTINE.                   CANO 240
C           WLAM  - OUTPUT VECTOR OF LENGTH MQ CONTAINING LAMBDA.       CANO 250
C           CANR  - OUTPUT VECTOR OF LENGTH MQ CONTAINING CANONICAL     CANO 260
C                   CORRELATIONS.                                       CANO 270
C           CHISQ - OUTPUT VECTOR OF LENGTH MQ CONTAINING THE           CANO 280
C                   VALUES OF CHI-SQUARES.                              CANO 290
C           NDF   - OUTPUT VECTOR OF LENGTH MQ CONTAINING THE DEGREES   CANO 300
C                   OF FREEDOM ASSOCIATED WITH CHI-SQUARES.             CANO 310
C           COEFR - OUTPUT MATRIX (MQ X MQ) CONTAINING MQ SETS OF       CANO 320
C                   RIGHT HAND COEFFICIENTS COLUMNWISE.                 CANO 330
C           COEFL - OUTPUT MATRIX (MP X MQ) CONTAINING MQ SETS OF       CANO 340
C                   LEFT HAND COEFFICIENTS COLUMNWISE.                  CANO 350
C           R     - WORK MATRIX (M X M)                                 CANO 360
C                                                                       CANO 370
C        REMARKS                                                        CANO 380
C           THE NUMBER OF LEFT HAND VARIABLES (MP) SHOULD BE GREATER    CANO 390
C           THAN OR EQUAL TO THE NUMBER OF RIGHT HAND VARIABLES (MQ).   CANO 400
C           THE VALUES OF CANONICAL CORRELATION, LAMBDA, CHI-SQUARE,    CANO 410
C           DEGREES OF FREEDOM, AND CANONICAL COEFFICIENTS ARE COMPUTED CANO 420
C           ONLY FOR THOSE EIGENVALUES IN ROOTS WHICH ARE GREATER THAN  CANO 430
C           ZERO.                                                       CANO 440
C                                                                       CANO 450
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CANO 460
C           MINV                                                        CANO 470
C           NROOT  (WHICH, IN TURN, CALLS THE SUBROUTINE EIGEN.)        CANO 480
C                                                                       CANO 490
C        METHOD                                                         CANO 500
C           REFER TO W. W. COOLEY AND P. R. LOHNES, 'MULTIVARIATE PRO-  CANO 510
C           CEDURES FOR THE BEHAVIORAL SCIENCES', JOHN WILEY AND SONS,  CANO 520
C           1962, CHAPTER 3.                                            CANO 530
C                                                                       CANO 540
C     ..................................................................CANO 550
C                                                                       CANO 560
      SUBROUTINE CANOR (N,MP,MQ,RR,ROOTS,WLAM,CANR,CHISQ,NDF,COEFR,     CANO 570
     1                  COEFL,R)                                        CANO 580
      DIMENSION RR(1),ROOTS(1),WLAM(1),CANR(1),CHISQ(1),NDF(1),COEFR(1),CANO 590
     1          COEFL(1),R(1)                                           CANO 600
C                                                                       CANO 610
C        ...............................................................CANO 620
C                                                                       CANO 630
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  CANO 640
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      CANO 650
C        STATEMENT WHICH FOLLOWS.                                       CANO 660
C                                                                       CANO 670
C     DOUBLE PRECISION RR,ROOTS,WLAM,CANR,CHISQ,COEFR,COEFL,R,DET,SUM   CANO 680
C                                                                       CANO 690
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    CANO 700
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      CANO 710
C        ROUTINE.                                                       CANO 720
C                                                                       CANO 730
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      CANO 740
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENT CANO 750
C        165 MUST BE CHANGED TO DSQRT.  ALOG IN STATEMENT 175 MUST BE   CANO 760
C        CHANGED TO DLOG.                                               CANO 770
C                                                                       CANO 780
C        ...............................................................CANO 790
C                                                                       CANO 800
C     PARTITION INTERCORRELATIONS AMONG LEFT HAND VARIABLES, BETWEEN    CANO 810
C     LEFT AND RIGHT HAND VARIABLES, AND AMONG RIGHT HAND VARIABLES.    CANO 820
C                                                                       CANO 830
      M=MP+MQ                                                           CANO 840
      N1=0                                                              CANO 850
      DO 105 I=1,M                                                      CANO 860
      DO 105 J=1,M                                                      CANO 870
      IF(I-J) 102, 103, 103                                             CANO 880
  102 L=I+(J*J-J)/2                                                     CANO 890
      GO TO 104                                                         CANO 900
  103 L=J+(I*I-I)/2                                                     CANO 910
  104 N1=N1+1                                                           CANO 920
  105 R(N1)=RR(L)                                                       CANO 930
      L=MP                                                              CANO 940
      DO 108 J=2,MP                                                     CANO 950
      N1=M*(J-1)                                                        CANO 960
      DO 108 I=1,MP                                                     CANO 970
      L=L+1                                                             CANO 980
      N1=N1+1                                                           CANO 990
  108 R(L)=R(N1)                                                        CANO1000
      N2=MP+1                                                           CANO1010
      L=0                                                               CANO1020
      DO 110 J=N2,M                                                     CANO1030
      N1=M*(J-1)                                                        CANO1040
      DO 110 I=1,MP                                                     CANO1050
      L=L+1                                                             CANO1060
      N1=N1+1                                                           CANO1070
  110 COEFL(L)=R(N1)                                                    CANO1080
      L=0                                                               CANO1090
      DO 120 J=N2,M                                                     CANO1100
      N1=M*(J-1)+MP                                                     CANO1110
      DO 120 I=N2,M                                                     CANO1120
      L=L+1                                                             CANO1130
      N1=N1+1                                                           CANO1140
  120 COEFR(L)=R(N1)                                                    CANO1150
C                                                                       CANO1160
C     SOLVE THE CANONICAL EQUATION                                      CANO1170
C                                                                       CANO1180
      L=MP*MP+1                                                         CANO1190
      K=L+MP                                                            CANO1200
      CALL MINV (R,MP,DET,R(L),R(K))                                    CANO1210
C                                                                       CANO1220
C        CALCULATE T = INVERSE OF R11 * R12                             CANO1230
C                                                                       CANO1240
      DO 140 I=1,MP                                                     CANO1250
      N2=0                                                              CANO1260
      DO 130 J=1,MQ                                                     CANO1270
      N1=I-MP                                                           CANO1280
      ROOTS(J)=0.0                                                      CANO1290
      DO 130 K=1,MP                                                     CANO1300
      N1=N1+MP                                                          CANO1310
      N2=N2+1                                                           CANO1320
  130 ROOTS(J)=ROOTS(J)+R(N1)*COEFL(N2)                                 CANO1330
      L=I-MP                                                            CANO1340
      DO 140 J=1,MQ                                                     CANO1350
      L=L+MP                                                            CANO1360
  140 R(L)=ROOTS(J)                                                     CANO1370
C                                                                       CANO1380
C        CALCULATE A = R21 * T                                          CANO1390
C                                                                       CANO1400
      L=MP*MQ                                                           CANO1410
      N3=L+1                                                            CANO1420
      DO 160 J=1,MQ                                                     CANO1430
      N1=0                                                              CANO1440
      DO 160 I=1,MQ                                                     CANO1450
      N2=MP*(J-1)                                                       CANO1460
      SUM=0.0                                                           CANO1470
      DO 150 K=1,MP                                                     CANO1480
      N1=N1+1                                                           CANO1490
      N2=N2+1                                                           CANO1500
  150 SUM=SUM+COEFL(N1)*R(N2)                                           CANO1510
      L=L+1                                                             CANO1520
  160 R(L)=SUM                                                          CANO1530
C                                                                       CANO1540
C        CALCULATE EIGENVALUES WITH ASSOCIATED EIGENVECTORS OF THE      CANO1550
C        INVERSE OF R22 * A                                             CANO1560
C                                                                       CANO1570
      L=L+1                                                             CANO1580
      CALL NROOT (MQ,R(N3),COEFR,ROOTS,R(L))                            CANO1590
C                                                                       CANO1600
C     FOR EACH VALUE OF I = 1, 2, ..., MQ, CALCULATE THE FOLLOWING      CANO1610
C     STATISTICS                                                        CANO1620
C                                                                       CANO1630
      DO 210 I=1,MQ                                                     CANO1640
C                                                                       CANO1650
C        TEST WHETHER EIGENVALUE IS GREATER THAN ZERO                   CANO1660
C                                                                       CANO1670
      IF(ROOTS(I)) 220, 220, 165                                        CANO1680
C                                                                       CANO1690
C        CANONICAL CORRELATION                                          CANO1700
C                                                                       CANO1710
  165 CANR(I)= SQRT(ROOTS(I))                                           CANO1720
C                                                                       CANO1730
C        CHI-SQUARE                                                     CANO1740
C                                                                       CANO1750
      WLAM(I)=1.0                                                       CANO1760
      DO 170 J=I,MQ                                                     CANO1770
  170 WLAM(I)=WLAM(I)*(1.0-ROOTS(J))                                    CANO1780
      FN=N                                                              CANO1790
      FMP=MP                                                            CANO1800
      FMQ=MQ                                                            CANO1810
  175 CHISQ(I)=-(FN-0.5*(FMP+FMQ+1.0))*ALOG(WLAM(I))                    CANO1820
C                                                                       CANO1830
C        DEGREES OF FREEDOM FOR CHI-SQUARE                              CANO1840
C                                                                       CANO1850
      N1=I-1                                                            CANO1860
      NDF(I)=(MP-N1)*(MQ-N1)                                            CANO1870
C                                                                       CANO1880
C        I-TH SET OF RIGHT HAND COEFFICIENTS                            CANO1890
C                                                                       CANO1900
      N1=MQ*(I-1)                                                       CANO1910
      N2=MQ*(I-1)+L-1                                                   CANO1920
      DO 180 J=1,MQ                                                     CANO1930
      N1=N1+1                                                           CANO1940
      N2=N2+1                                                           CANO1950
  180 COEFR(N1)=R(N2)                                                   CANO1960
C                                                                       CANO1970
C        I-TH SET OF LEFT HAND COEFFICIENTS                             CANO1980
C                                                                       CANO1990
      DO 200 J=1,MP                                                     CANO2000
      N1=J-MP                                                           CANO2010
      N2=MQ*(I-1)                                                       CANO2020
      K=MP*(I-1)+J                                                      CANO2030
      COEFL(K)=0.0                                                      CANO2040
      DO 190 JJ=1,MQ                                                    CANO2050
      N1=N1+MP                                                          CANO2060
      N2=N2+1                                                           CANO2070
  190 COEFL(K)=COEFL(K)+R(N1)*COEFR(N2)                                 CANO2080
  200 COEFL(K)=COEFL(K)/CANR(I)                                         CANO2090
  210 CONTINUE                                                          CANO2100
  220 RETURN                                                            CANO2110
      END                                                               CANO2120
```

Design Analysis (see Smoothing, Regression, Factorization)

In the Scientific Subroutine Package, analysis of variance is normally performed by calling the following three subroutines in sequence:

1. AVDAT - to place data in properly distributed positions of storage
2. AVCAL - to apply the operators sigma and delta in order to compute deviates for analysis of variance
3. MEANQ - to pool the deviates and compute sums of squares, degrees of freedom, and mean squares

Subroutine AVDAT

This subroutine places data for analysis of variance in properly distributed positions of storage.

The size of data array X required for an analysis-of-variance problem is calculated as follows:

$$n = \prod_{i=1}^{k} (L_i + 1) \qquad (1)$$

where L_i = number of levels of ith factor

k = number of factors

The input data placed in the lower part of the array X are moved temporarily to the upper part of the array. From there, the data are redistributed according to the equation (4) below. Prior to that, multipliers, s_j, to be used in finding proper positions of storage, are calculated as follows:

$$s_1 = 1 \qquad (2)$$

$$s_j = \prod_{i=1}^{j-1} (L_i + 1) \qquad (3)$$

where $j = 2, 3, \ldots, k$

Then a position for each data point is calculated by the following equation:

$$S = KOUNT_1 + \sum_{j=2}^{k} s_j \cdot (KOUNT_j - 1) \qquad (4)$$

where $KOUNT_j$ = value of jth subscript of the data to be stored.

The subroutine increments the value(s) of subscript(s) after each data point is stored.

```
C                                                                       AVDA  10
C     ...................................................................AVDA  20
C                                                                       AVDA  30
C     SUBROUTINE AVDAT                                                  AVDA  40
C                                                                       AVDA  50
C     PURPOSE                                                           AVDA  60
C        PLACE DATA FOR ANALYSIS OF VARIANCE IN PROPERLY DISTRIBUTED    AVDA  70
C        POSITIONS OF STORAGE.  THIS SUBROUTINE IS NORMALLY FOLLOWED    AVDA  80
C        BY CALLS TO AVCAL AND MEANQ SUBROUTINES IN THE PERFORMANCE     AVDA  90
C        OF ANALYSIS OF VARIANCE FOR A COMPLETE FACTORIAL DESIGN.       AVDA 100
C                                                                       AVDA 110
C     USAGE                                                             AVDA 120
C        CALL AVDAT (K,LEVEL,N,X,L,ISTEP,KOUNT)                         AVDA 130
C                                                                       AVDA 140
C     DESCRIPTION OF PARAMETERS                                         AVDA 150
C        K     - NUMBER OF VARIABLES (FACTORS). K MUST BE .GT. ONE.     AVDA 160
C        LEVEL - INPUT VECTOR OF LENGTH K CONTAINING LEVELS (CATE-      AVDA 170
C                GORIES) WITHIN EACH VARIABLE.                          AVDA 180
C        N     - TOTAL NUMBER OF DATA POINTS READ IN.                   AVDA 190
C        X     - WHEN THE SUBROUTINE IS CALLED, THIS VECTOR CONTAINS    AVDA 200
C                DATA IN LOCATIONS X(1) THROUGH X(N).  UPON RETURNINGAVDA 210
C                TO THE CALLING ROUTINE, THE VECTOR CONTAINS THE DATAAVDA 220
C                IN PROPERLY REDISTRIBUTED LOCATIONS OF VECTOR X.       AVDA 230
C                THE LENGTH OF VECTOR X IS CALCULATED BY (1) ADDING     AVDA 240
C                ONE TO EACH LEVEL OF VARIABLE AND (2) OBTAINING THE    AVDA 250
C                CUMULATIVE PRODUCT OF ALL LEVELS.  (THE LENGTH OF      AVDA 260
C                X = (LEVEL(1)+1)*(LEVEL(2)+1)*...*(LEVEL(K)+1).)       AVDA 270
C        L     - OUTPUT VARIABLE CONTAINING THE POSITION IN VECTOR X    AVDA 280
C                WHERE THE LAST INPUT DATA IS STORED.                   AVDA 290
C        ISTEP - OUTPUT VECTOR OF LENGTH K CONTAINING CONTROL STEPS     AVDA 300
C                WHICH ARE USED TO LOCATE DATA IN PROPER POSITIONS      AVDA 310
C                OF VECTOR X.                                           AVDA 320
C        KOUNT - WORKING VECTOR OF LENGTH K.                            AVDA 330
C                                                                       AVDA 340
C     REMARKS                                                           AVDA 350
C        INPUT DATA MUST BE ARRANGED IN THE FOLLOWING MANNER.           AVDA 360
C        CONSIDER THE 3-VARIABLE ANALYSIS OF VARIANCE DESIGN, WHERE     AVDA 370
C        ONE VARIABLE HAS 3 LEVELS AND THE OTHER TWO VARIABLES HAVE     AVDA 380
C        2 LEVELS.  THE DATA MAY BE REPRESENTED IN THE FORM X(I,J,K),AVDA 390
C        I=1,2,3  J=1,2  K=1,2.  IN ARRANGING DATA, THE INNER           AVDA 400
C        SUBSCRIPT, NAMELY I, CHANGES FIRST.  WHEN I=3, THE NEXT        AVDA 410
C        INNER SUBSCRIPT, J, CHANGES AND SO ON UNTIL I=3, J=2, AND      AVDA 420
C        K=2.                                                           AVDA 430
C                                                                       AVDA 440
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     AVDA 450
C        NONE                                                           AVDA 460
C                                                                       AVDA 470
C     METHOD                                                            AVDA 480
C        THE METHOD IS BASED ON THE TECHNIQUE DISCUSSED BY H. O.        AVDA 490
C        HARTLEY IN 'MATHEMATICAL METHODS FOR DIGITAL COMPUTERS',       AVDA 500
C        EDITED BY A. RALSTON AND H. WILF, JOHN WILEY AND SONS,         AVDA 510
C        1962, CHAPTER 20.                                              AVDA 520
C                                                                       AVDA 530
C     ...................................................................AVDA 540
C                                                                       AVDA 550
      SUBROUTINE AVDAT (K,LEVEL,N,X,L,ISTEP,KOUNT)                      AVDA 560
      DIMENSION LEVEL(1),X(1),ISTEP(1),KOUNT(1)                         AVDA 570
C                                                                       AVDA 580
C        ...............................................................AVDA 590
C                                                                       AVDA 600
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  AVDA 610
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      AVDA 620
C        STATEMENT WHICH FOLLOWS.                                       AVDA 630
C                                                                       AVDA 640
C     DOUBLE PRECISION X                                                AVDA 650
C                                                                       AVDA 660
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    AVDA 670
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      AVDA 680
C        ROUTINE.                                                       AVDA 690
C                                                                       AVDA 700
C        ...............................................................AVDA 710
C                                                                       AVDA 720
C     CALCULATE TOTAL DATA AREA REQUIRED                                AVDA 730
C                                                                       AVDA 740
      M=LEVEL(1)+1                                                      AVDA 750
      DO 105 I=2,K                                                      AVDA 760
  105 M=M*(LEVEL(I)+1)                                                  AVDA 770
C                                                                       AVDA 780
C     MOVE DATA TO THE UPPER PART OF THE ARRAY X                        AVDA 790
C     FOR THE PURPOSE OF REARRANGEMENT                                  AVDA 800
C                                                                       AVDA 810
      N1=M+1                                                            AVDA 820
      N2=N+1                                                            AVDA 830
      DO 107 I=1,N                                                      AVDA 840
      N1=N1-1                                                           AVDA 850
      N2=N2-1                                                           AVDA 860
  107 X(N1)=X(N2)                                                       AVDA 870
C                                                                       AVDA 880
C     CALCULATE MULTIPLIERS TO BE USED IN FINDING STORAGE LOCATIONS FOR AVDA 890
C     INPUT DATA                                                        AVDA 900
C                                                                       AVDA 910
      ISTEP(1)=1                                                        AVDA 920
      DO 110 I=2,K                                                      AVDA 930
  110 ISTEP(I)=ISTEP(I-1)*(LEVEL(I-1)+1)                                AVDA 940
      DO 115 I=1,K                                                      AVDA 950
  115 KOUNT(I)=1                                                        AVDA 960
C                                                                       AVDA 970
C     PLACE DATA IN PROPER LOCATIONS                                    AVDA 980
C                                                                       AVDA 990
      N1=N1-1                                                           AVDA1000
      DO 135 I=1,N                                                      AVDA1010
      L=KOUNT(1)                                                        AVDA1020
      DO 120 J=2,K                                                      AVDA1030
  120 L=L+ISTEP(J)*(KOUNT(J)-1)                                         AVDA1040
      N1=N1+1                                                           AVDA1050
      X(L)=X(N1)                                                        AVDA1060
      DO 130 J=1,K                                                      AVDA1070
      IF(KOUNT(J)-LEVEL(J)) 124, 125, 124                               AVDA1080
  124 KOUNT(J)=KOUNT(J)+1                                               AVDA1090
      GO TO 135                                                         AVDA1100
  125 KOUNT(J)=1                                                        AVDA1110
  130 CONTINUE                                                          AVDA1120
  135 CONTINUE                                                          AVDA1130
      RETURN                                                            AVDA1140
      END                                                               AVDA1150
```

Subroutine AVCAL

This subroutine performs the calculus for the general k-factor experiment: operator Σ and operator Δ. An example is presented in terms of $k=3$ to illustrate these operators.

Let x_{abc} denote the experimental reading from the ath level of factor A, the bth level of factor B, and the cth level of factor C. The symbols A, B, and C will also denote the number of levels for each factor so that $a = 1, 2, \ldots, A$; $b = 1, 2, \ldots, B$; and $c = 1, 2, \ldots, C$.

With regard to the first factor A:

operator $\sum_a \equiv$ sum over all levels $a = 1, 2, \ldots, A$, holding the other subscripts at constant levels, and

operator $\underset{a}{\Delta} \equiv$ multiply all items by A and subtract the result of $\sum_a$ from all items

In mathematical notation, these operators are defined as follows:

$$\sum_a x_{abc} \equiv X_{.bc} \equiv \sum_{a=1}^{A} x_{abc} \qquad (1)$$

$$\underset{a}{\Delta} x_{abc} \equiv A\, x_{abc} - X_{.bc} \qquad (2)$$

The operators Σ and Δ will be applied sequentially to all factors A, B, and C. Upon the completion of these operations, the storage array X contains deviates to be used for analysis-of-variance components in the subroutine MEANQ.

```
C                                                                       AVCA  10
C     ...................................................................AVCA  20
C                                                                       AVCA  30
C        SUBROUTINE AVCAL                                               AVCA  40
C                                                                       AVCA  50
C        PURPOSE                                                        AVCA  60
C           PERFORM THE CALCULUS OF A FACTORIAL EXPERIMENT USING        AVCA  70
C           OPERATOR SIGMA AND OPERATOR DELTA.  THIS SUBROUTINE IS      AVCA  80
C           PRECEDED BY SUBROUTINE ADVAT AND FOLLOWED BY SUBROUTINE     AVCA  90
C           MEANQ IN THE PERFORMANCE OF ANALYSIS OF VARIANCE FOR A      AVCA 100
C           COMPLETE FACTORIAL DESIGN.                                  AVCA 110
C                                                                       AVCA 120
C        USAGE                                                          AVCA 130
C           CALL AVCAL (K,LEVEL,X,L,ISTEP,LASTS)                        AVCA 140
C                                                                       AVCA 150
C        DESCRIPTION OF PARAMETERS                                      AVCA 160
C           K      - NUMBER OF VARIABLES (FACTORS). K MUST BE .GT. ONE. AVCA 170
C           LEVEL  - INPUT VECTOR OF LENGTH K CONTAINING LEVELS (CATE-  AVCA 180
C                    GORIES) WITHIN EACH VARIABLE.                      AVCA 190
C           X      - INPUT VECTOR CONTAINING DATA.  DATA HAVE BEEN PLACEDAVCA 200
C                    IN VECTOR X BY SUBROUTINE AVDAT.  THE LENGTH OF X  AVCA 210
C                    IS (LEVEL(1)+1)*(LEVEL(2)+1)*...*(LEVEL(K)+1).     AVCA 220
C           L      - THE POSITION IN VECTOR X WHERE THE LAST INPUT DATA AVCA 230
C                    IS LOCATED.  L HAS BEEN CALCULATED BY SUBROUTINE   AVCA 240
C                    AVDAT.                                             AVCA 250
C           ISTEP  - INPUT VECTOR OF LENGTH K CONTAINING STORAGE CONTROL AVCA 260
C                    STEPS WHICH HAVE BEEN CALCULATED BY SUBROUTINE     AVCA 270
C                    AVDAT.                                             AVCA 280
C           LASTS  - WORKING VECTOR OF LENGTH K.                        AVCA 290
C                                                                       AVCA 300
C        REMARKS                                                        AVCA 310
C           THIS SUBROUTINE MUST FOLLOW SUBROUTINE AVDAT.               AVCA 320
C                                                                       AVCA 330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  AVCA 340
C           NONE                                                        AVCA 350
C                                                                       AVCA 360
C        METHOD                                                         AVCA 370
C           THE METHOD IS BASED ON THE TECHNIQUE DISCUSSED BY H. O.     AVCA 380
C           HARTLEY IN 'MATHEMATICAL METHODS FOR DIGITAL COMPUTERS',    AVCA 390
C           EDITED BY A. RALSTON AND H. WILF, JOHN WILEY AND SONS,      AVCA 400
C           1962, CHAPTER 20.                                           AVCA 410
C                                                                       AVCA 420
C     ...................................................................AVCA 430
C                                                                       AVCA 440
      SUBROUTINE AVCAL (K,LEVEL,X,L,ISTEP,LASTS)                        AVCA 450
      DIMENSION LEVEL(1),X(1),ISTEP(1),LASTS(1)                         AVCA 460
C                                                                       AVCA 470
C        ................................................................AVCA 480
C                                                                       AVCA 490
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  AVCA 500
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      AVCA 510
C        STATEMENT WHICH FOLLOWS.                                       AVCA 520
C                                                                       AVCA 530
C     DOUBLE PRECISION X,SUM                                            AVCA 540
C                                                                       AVCA 550
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    AVCA 560
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      AVCA 570
C        ROUTINE.                                                       AVCA 580
C                                                                       AVCA 590
C        ................................................................AVCA 600
C                                                                       AVCA 610
C     CALCULATE THE LAST DATA POSITION OF EACH FACTOR                   AVCA 620
C                                                                       AVCA 630
      LASTS(1)=L+1                                                      AVCA 640
      DO 145 I=2,K                                                      AVCA 650
  145 LASTS(I)=LASTS(I-1)+ISTEP(I)                                      AVCA 660
C                                                                       AVCA 670
C     PERFORM CALCULUS OF OPERATION                                     AVCA 680
C                                                                       AVCA 690
  150 DO 175 I=1,K                                                      AVCA 700
      L=1                                                               AVCA 710
      LL=1                                                              AVCA 720
      SUM=0.0                                                           AVCA 730
      NN=LEVEL(I)                                                       AVCA 740
      FN=NN                                                             AVCA 750
      INCRE=ISTEP(I)                                                    AVCA 760
      LAST=LASTS(I)                                                     AVCA 770
C                                                                       AVCA 780
C     SIGMA OPERATION                                                   AVCA 790
C                                                                       AVCA 800
  155 DO 160 J=1,NN                                                     AVCA 810
      SUM=SUM+X(L)                                                      AVCA 820
  160 L=L+INCRE                                                         AVCA 830
      X(L)=SUM                                                          AVCA 840
C                                                                       AVCA 850
C     DELTA OPERATION                                                   AVCA 860
C                                                                       AVCA 870
      DO 165 J=1,NN                                                     AVCA 880
      X(LL)=FN*X(LL)-SUM                                                AVCA 890
  165 LL=LL+INCRE                                                       AVCA 900
      SUM=0.0                                                           AVCA 910
      IF(L-LAST) 167, 175, 175                                          AVCA 920
  167 IF(L-LAST+INCRE) 168, 168, 170                                    AVCA 930
  168 L=L+INCRE                                                         AVCA 940
      LL=LL+INCRE                                                       AVCA 950
      GO TO 155                                                         AVCA 960
  170 L=L+INCRE+1-LAST                                                  AVCA 970
      LL=LL+INCRE+1-LAST                                                AVCA 980
      GO TO 155                                                         AVCA 990
  175 CONTINUE                                                          AVCA1000
      RETURN                                                            AVCA1010
      END                                                               AVCA1020
```

Subroutine MEANQ

This subroutine performs the mean square operation for the general k-factor experiment in the following two steps:

1. Square each value of deviates for analysis of variance stored in the array X (the result of the operators Σ and Δ applied in the subroutine AVCAL).

2. Add the squared value into summation storage. In a three-factor experiment, for example, the squared value is added into one of seven storages ($7 = 2^3 - 1$) as shown in the first column of Table 1. The symbols A, B, and C in the first column denote factor A, factor B, and factor C.

After the mean square operation is completed for all values in the storage array X, the subroutine forms sums of squares of analysis of variance by dividing the totals of squared values by proper divisors. These divisors for the three-factor experiment mentioned above are shown in the second column of Table 1. The symbols A, B, and C in the second column denote the number of levels for each factor.

The subroutine, then, forms mean squares by dividing sums of squares by degrees of freedom. The third column of the summary table shows the degrees of freedom. The symbols A, B, and C denote the number of levels for each factor.

Table 1. Table Showing Summation Storages, Divisors to Form Sum of Squares, and Degrees of Freedom (Subroutine MEANQ)

Designation of Store and of Quantity Contained in it	Divisor Required to Form Sum of Squares of Analysis of Variance	Degrees of Freedom Required to Form Mean Squares
$(A)^2$	ABC·A	(A-1)
$(B)^2$	ABC·B	(B-1)
$(AB)^2$	ABC·AB	(A-1) (B-1)
$(C)^2$	ABC·C	(C-1)
$(AC)^2$	ABC·AC	(A-1) (C-1)
$(BC)^2$	ABC·BC	(B-1) (C-1)
$(ABC)^2$	ABC·ABC	(A-1) (B-1) (C-1)

```
C                                                                       MEAN  10
C     ...................................................................MEAN  20
C                                                                       MEAN  30
C       SUBROUTINE MEANQ                                                MEAN  40
C                                                                       MEAN  50
C       PURPOSE                                                         MEAN  60
C          COMPUTE SUM OF SQUARES, DEGREES OF FREEDOM, AND MEAN SQUARE  MEAN  70
C          USING THE MEAN SQUARE OPERATOR.  THIS SUBROUTINE NORMALLY    MEAN  80
C          FOLLOWS CALLS TO AVDAT AND AVCAL SUBROUTINES IN THE PER-     MEAN  90
C          FORMANCE OF ANALYSIS OF VARIANCE FOR A COMPLETE FACTORIAL    MEAN 100
C          DESIGN.                                                      MEAN 110
C                                                                       MEAN 120
C       USAGE                                                           MEAN 130
C          CALL MEANQ (K,LEVEL,X,GMEAN,SUMSQ,NDF,SMEAN,MSTEP,KOUNT,     MEAN 140
C                      LASTS)                                           MEAN 150
C                                                                       MEAN 160
C       DESCRIPTION OF PARAMETERS                                       MEAN 170
C          K     - NUMBER OF VARIABLES (FACTORS). K MUST BE .GT. ONE.   MEAN 180
C          LEVEL - INPUT VECTOR OF LENGTH K CONTAINING LEVELS (CATE-    MEAN 190
C                  GORIES) WITHIN EACH VARIABLE.                        MEAN 200
C          X     - INPUT VECTOR CONTAINING THE RESULT OF THE SIGMA AND  MEAN 210
C                  DELTA OPERATORS. THE LENGTH OF X IS                  MEAN 220
C                  (LEVEL(1)+1)*(LEVEL(2)+1)*...*(LEVEL(K)+1).          MEAN 230
C          GMEAN - OUTPUT VARIABLE CONTAINING GRAND MEAN.               MEAN 240
C          SUMSQ - OUTPUT VECTOR CONTAINING SUMS OF SQUARES.  THE       MEAN 250
C                  LENGTH OF SUMSQ IS 2 TO THE K-TH POWER MINUS ONE,    MEAN 260
C                  (2**K)-1.                                            MEAN 270
C          NDF   - OUTPUT VECTOR CONTAINING DEGREES OF FREEDOM.  THE    MEAN 280
C                  LENGTH OF NDF IS 2 TO THE K-TH POWER MINUS ONE,      MEAN 290
C                  (2**K)-1.                                            MEAN 300
C          SMEAN - OUTPUT VECTOR CONTAINING MEAN SQUARES.  THE          MEAN 310
C                  LENGTH OF SMEAN IS 2 TO THE K-TH POWER MINUS ONE,    MEAN 320
C                  (2**K)-1.                                            MEAN 330
C          MSTEP - WORKING VECTOR OF LENGTH K.                          MEAN 340
C          KOUNT - WORKING VECTOR OF LENGTH K.                          MEAN 350
C          LASTS - WORKING VECTOR OF LENGTH K.                          MEAN 360
C                                                                       MEAN 370
C       REMARKS                                                         MEAN 380
C          THIS SUBROUTINE MUST FOLLOW SUBROUTINE AVCAL                 MEAN 390
C                                                                       MEAN 400
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   MEAN 410
C          NONE                                                         MEAN 420
C                                                                       MEAN 430
C       METHOD                                                          MEAN 440
C          THE METHOD IS BASED ON THE TECHNIQUE DISCUSSED BY H. O.      MEAN 450
C          HARTLEY IN 'MATHEMATICAL METHODS FOR DIGITAL COMPUTERS',     MEAN 460
C          EDITED BY A. RALSTON AND H. WILF, JOHN WILEY AND SONS,       MEAN 470
C          1962, CHAPTER 20.                                            MEAN 480
C                                                                       MEAN 490
C     ...................................................................MEAN 500
C                                                                       MEAN 510
      SUBROUTINE MEANQ (K,LEVEL,X,GMEAN,SUMSQ,NDF,SMEAN,MSTEP,KOUNT,    MEAN 520
     1                  LASTS)                                          MEAN 530
      DIMENSION LEVEL(1),X(1),SUMSQ(1),NDF(1),SMEAN(1),MSTEP(1),        MEAN 540
     1          KOUNT(1),LASTS(1)                                       MEAN 550
C                                                                       MEAN 560
C     ...................................................................MEAN 570
C                                                                       MEAN 580
C       IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE   MEAN 590
C       C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION       MEAN 600
C       STATEMENT WHICH FOLLOWS.                                        MEAN 610
C                                                                       MEAN 620
C     DOUBLE PRECISION X,GMEAN,SUMSQ,SMEAN,FN1                          MEAN 630
C                                                                       MEAN 640
C       THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS     MEAN 650
C       APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS       MEAN 660
C       ROUTINE.                                                        MEAN 670
C                                                                       MEAN 680
C     ...................................................................MEAN 690
C                                                                       MEAN 700
C     CALCULATE TOTAL NUMBER OF DATA                                    MEAN 710
C                                                                       MEAN 720
      N=LEVEL(1)                                                        MEAN 730
      DO 150 I=2,K                                                      MEAN 740
  150 N=N*LEVEL(I)                                                      MEAN 750
C                                                                       MEAN 760
C     SET UP CONTROL FOR MEAN SQUARE OPERATOR                           MEAN 770
C                                                                       MEAN 780
      LASTS(1)=LEVEL(1)                                                 MEAN 790
      DO 178 I=2,K                                                      MEAN 800
  178 LASTS(I)=LEVEL(I)+1                                               MEAN 810
      NN=1                                                              MEAN 820
C                                                                       MEAN 830
C     CLEAR THE AREA TO STORE SUMS OF SQUARES                           MEAN 840
C                                                                       MEAN 850
      LL=(2**K)-1                                                       MEAN 860
      MSTEP(1)=1                                                        MEAN 870
      DO 180 I=2,K                                                      MEAN 880
  180 MSTEP(I)=MSTEP(I-1)*2                                             MEAN 890
      DO 185 I=1,LL                                                     MEAN 900
  185 SUMSQ(I)=0.0                                                      MEAN 910
C                                                                       MEAN 920
C     PERFORM MEAN SQUARE OPERATOR                                      MEAN 930
C                                                                       MEAN 940
      DO 190 I=1,K                                                      MEAN 950
  190 KOUNT(I)=0                                                        MEAN 960
  200 L=0                                                               MEAN 970
      DO 260 I=1,K                                                      MEAN 980
      IF(KOUNT(I)-LASTS(I)) 210, 250, 210                               MEAN 990
  210 IF(L) 220, 220, 240                                               MEAN1000
  220 KOUNT(I)=KOUNT(I)+1                                               MEAN1010
      IF(KOUNT(I)-LEVEL(I)) 230, 230, 250                               MEAN1020
  230 L=L+MSTEP(I)                                                      MEAN1030
      GO TO 260                                                         MEAN1040
  240 IF(KOUNT(I)-LEVEL(I)) 230, 260, 230                               MEAN1050
  250 KOUNT(I)=0                                                        MEAN1060
  260 CONTINUE                                                          MEAN1070
      IF(L) 285, 285, 270                                               MEAN1080
  270 SUMSQ(L)=SUMSQ(L)+X(NN)*X(NN)                                     MEAN1090
      NN=NN+1                                                           MEAN1100
      GO TO 200                                                         MEAN1110
C                                                                       MEAN1120
C     CALCULATE THE GRAND MEAN                                          MEAN1130
C                                                                       MEAN1140
  285 FN=N                                                              MEAN1150
      GMEAN=X(NN)/FN                                                    MEAN1160
C                                                                       MEAN1170
C     CALCULATE FIRST DIVISOR REQUIRED TO FORM SUM OF SQUARES AND SECOND MEAN1180
C     DIVISOR, WHICH IS EQUAL TO DEGREES OF FREEDOM, REQUIRED TO FORM   MEAN1190
C     MEAN SQUARES                                                      MEAN1200
C                                                                       MEAN1210
      DO 310 I=2,K                                                      MEAN1220
  310 MSTEP(I)=0                                                        MEAN1230
      NN=0                                                              MEAN1240
      MSTEP(1)=1                                                        MEAN1250
  320 ND1=1                                                             MEAN1260
      ND2=1                                                             MEAN1270
      DO 340 I=1,K                                                      MEAN1280
      IF(MSTEP(I)) 330, 340, 330                                        MEAN1290
```

```
330 ND1=ND1*LEVEL(I)                                    MEAN1300
    ND2=ND2*(LEVEL(I)-1)                                MEAN1310
340 CONTINUE                                            MEAN1320
    FN1=N*ND1                                           MEAN1330
    FN2=ND2                                             MEAN1340
    NN=NN+1                                             MEAN1350
    SUMSQ(NN)=SUMSQ(NN)/FN1                             MEAN1360
    NDF(NN)=ND2                                         MEAN1370
    SMEAN(NN)=SUMSQ(NN)/FN2                             MEAN1380
    IF(NN-LL) 345, 370, 370                             MEAN1390
345 DO 360 I=1,K                                        MEAN1400
    IF(MSTEP(I)) 347, 350, 347                          MEAN1410
347 MSTEP(I)=0                                          MEAN1420
    GO TO 360                                           MEAN1430
350 MSTEP(I)=1                                          MEAN1440
    GO TO 320                                           MEAN1450
360 CONTINUE                                            MEAN1460
370 RETURN                                              MEAN1470
    END                                                 MEAN1480
```

Discriminant Analysis

In the Scientific Subroutine Package, discriminant analysis is normally performed by calling the following three subroutines in sequence:

1. DMATX - to compute means of variables in each group and a pooled dispersion matrix
2. MINV - to invert the pooled dispersion matrix
3. DISCR - to compute coefficients of discriminant functions and evaluate the functions for each observation (individual)

Subroutine DMATX

This subroutine calculates means of variables in each group and a pooled dispersion matrix for the set of groups in a discriminant analysis.

For each group $k = 1, 2, \ldots, g$, the subroutine calculates means and sums of cross-products of deviations from means as shown below.

Means:

$$\bar{x}_{jk} = \frac{\sum_{i=1}^{n_k} x_{ijk}}{n_k} \tag{1}$$

where n_k = sample size in the kth group

$j = 1, 2, \ldots, m$ are variables

Sum of cross-products of deviations from means:

$$S_k = \left\{ s_{j\ell}^{k} \right\} = \sum (x_{ijk} - \bar{x}_{jk})\ (x_{i\ell k} - \bar{x}_{\ell k}) \tag{2}$$

where $j = 1, 2, \ldots, m$

$\ell = 1, 2, \ldots, m$

The pooled dispersion matrix is calculated as follows:

$$D = \frac{\sum_{k=1}^{g} S_k}{\sum_{k=1}^{g} n_k - g} \tag{3}$$

where g = number of groups

```
C
C     ..................................................................
C
C        SUBROUTINE DMATX
C
C        PURPOSE
C           COMPUTE MEANS OF VARIABLES IN EACH GROUP AND A POOLED
C           DISPERSION MATRIX FOR ALL THE GROUPS. NORMALLY THIS SUB-
C           ROUTINE IS USED IN THE PERFORMANCE OF DISCRIMINANT ANALYSIS.
C
C        USAGE
C           CALL DMATX (K,M,N,X,XBAR,D,CMEAN)
C
C        DESCRIPTION OF PARAMETERS
C           K     - NUMBER OF GROUPS
C           M     - NUMBER OF VARIABLES (MUST BE THE SAME FOR ALL
C                   GROUPS).
C           N     - INPUT VECTOR OF LENGTH K CONTAINING SAMPLE SIZES OF
C                   GROUPS.
C           X     - INPUT VECTOR CONTAINING DATA IN THE MANNER EQUIVA-
C                   LENT TO A 3-DIMENSIONAL FORTRAN ARRAY, X(1,1,1),
C                   X(2,1,1), X(3,1,1), ETC.  THE FIRST SUBSCRIPT IS
C                   CASE NUMBER, THE SECOND SUBSCRIPT IS VARIABLE NUMBER
C                   AND THE THIRD SUBSCRIPT IS GROUP NUMBER.  THE
C                   LENGTH OF VECTOR X IS EQUAL TO THE TOTAL NUMBER OF
C                   DATA POINTS, T*M, WHERE T = N(1)+N(2)+...+N(K).
C           XBAR  - OUTPUT MATRIX (M X K) CONTAINING MEANS OF VARIABLES
C                   IN K GROUPS.
C           D     - OUTPUT MATRIX (M X M) CONTAINING POOLED DISPERSION.
C           CMEAN - WORKING VECTOR OF LENGTH M.
C
C        REMARKS
C           THE NUMBER OF VARIABLES MUST BE GREATER THAN OR EQUAL TO
C           THE NUMBER OF GROUPS.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           REFER TO 'BMD COMPUTER PROGRAMS MANUAL', EDITED BY W. J.
C           DIXON, UCLA, 1964, AND T. W. ANDERSON, 'INTRODUCTION TO
C           MULTIVARIATE STATISTICAL ANALYSIS', JOHN WILEY AND SONS,
C           1958, SECTION 6.6-6.8.
C
C     ..................................................................
C
      SUBROUTINE DMATX (K,M,N,X,XBAR,D,CMEAN)
      DIMENSION N(1),X(1),XBAR(1),D(1),CMEAN(1)
C
C        ...............................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
C     DOUBLE PRECISION XBAR,D,CMEAN
C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        ...............................................................
C
C     INITIALIZATION
C
      MM=M*M
      DO 100 I=1,MM
  100 D(I)=0.0
C
C     CALCULATE MEANS
C
      N4=0
      L=0
      LM=0
      DO 160 NG=1,K
      N1=N(NG)
      FN=N1
      DO 130 J=1,M
      LM=LM+1
      XBAR(LM)=0.0
      DO 120 I=1,N1
      L=L+1
  120 XBAR(LM)=XBAR(LM)+X(L)
  130 XBAR(LM)=XBAR(LM)/FN
C
C     CALCULATE SUMS OF CROSS-PRODUCTS OF DEVIATIONS
C
      LMEAN=LM-M
      DO 150 I=1,N1
      LL=N4+I-N1
      DO 140 J=1,M
      LL=LL+N1
      N2=LMEAN+J
  140 CMEAN(J)=X(LL)-XBAR(N2)
      LL=0
      DO 150 J=1,M
      DO 150 JJ=1,M
      LL=LL+1
  150 D(LL)=D(LL)+CMEAN(J)*CMEAN(JJ)
  160 N4=N4+N1*M
C
C     CALCULATE THE POOLED DISPERSION MATRIX
C
      LL=-K
      DO 170 I=1,K
  170 LL=LL+N(I)
      FN=LL
      DO 180 I=1,MM
  180 D(I)=D(I)/FN
C
      RETURN
      END
```

Subroutine DISCR

This subroutine performs a discriminant analysis by calculating a set of linear functions that serve as indices for classifying an individual into one of K groups.

For all groups combined, the following are obtained.

Common means:

$$\bar{X}_j = \frac{\sum_{k=1}^{g} n_k \bar{x}_{jk}}{\sum_{k=1}^{g} n_k} \qquad (1)$$

where g = number of groups

j = 1, 2, ..., m are variables

n_k = sample size in the kth group

$\bar{x}_{jk}$ = mean of jth variable in k^{th} group

Generalized Mahalanobis D^2 statistics, V:

$$V = \sum_{i=1}^{m} \sum_{j=1}^{m} d_{ij} \sum_{k=1}^{g} n_k (\bar{x}_{ik} - \bar{X}_i)(\bar{x}_{jk} - \bar{X}_j) \qquad (2)$$

where d_{ij} = the inverse element of the pooled dispersion matrix D

V can be used as chi-square (under assumption of normality) with m(g-1) degrees of freedom, to test the hypothesis that the mean values are the same in all the g groups for these m variables. For each discriminant function $k_* = 1, 2, \ldots, g$, the following statistics are calculated.

Coefficients:

$$C_{ik*} = \sum_{j=1}^{m} d_{ij} \bar{x}_{jk} \qquad (3)$$

where $i = 1, 2, \ldots, m$

$k = k*$

Constant:

$$C_{ok*} = -1/2 \sum_{j=1}^{m} \sum_{l=1}^{m} d_{jl} \bar{x}_{jk} \bar{x}_{lk} \qquad (4)$$

For each ith case in each kth group, the following calculations are performed.

Discriminant functions:

$$f_{k*} = \sum_{j=1}^{m} C_{jk} x_{ijk} + C_{ok*} \qquad (5)$$

where $k* = 1, 2, \ldots, g$

Probability associated with largest discriminant function:

$$P_L = \frac{1}{\sum_{k*=1}^{g} e^{(f_{k*} - f_L)}} \qquad (6)$$

where f_L = the value of the largest discriminant function

L = the subscript of the largest discriminant function

```
C
C     ..................................................................
C
      SUBROUTINE DISCR
C
C     PURPOSE
C        COMPUTE A SET OF LINEAR FUNCTIONS WHICH SERVE AS INDICES
C        FOR CLASSIFYING AN INDIVIDUAL INTO ONE OF SEVERAL GROUPS.
C        NORMALLY THIS SUBROUTINE IS USED IN THE PERFORMANCE OF
C        DISCRIMINANT ANALYSIS.
C
C     USAGE
C        CALL DISCR (K,M,N,X,XBAR,D,CMEAN,V,C,P,LG)
C
C     DESCRIPTION OF PARAMETERS
C        K     - NUMBER OF GROUPS. K MUST BE GREATER THAN ONE.
C        M     - NUMBER OF VARIABLES
C        N     - INPUT VECTOR OF LENGTH K CONTAINING SAMPLE SIZES OF
C                GROUPS.
C        X     - INPUT VECTOR CONTAINING DATA IN THE MANNER EQUIVA-
C                LENT TO A 3-DIMENSIONAL FORTRAN ARRAY, X(1,1,1),
C                X(2,1,1), X(3,1,1), ETC.   THE FIRST SUBSCRIPT IS
C                CASE NUMBER, THE SECOND SUBSCRIPT IS VARIABLE NUMBER
C                AND THE THIRD SUBSCRIPT IS GROUP NUMBER.  THE
C                LENGTH OF VECTOR X IS EQUAL TO THE TOTAL NUMBER OF
C                DATA POINTS, T*M, WHERE T = N(1)+N(2)+...+N(K).
C        XBAR  - INPUT MATRIX (M X K) CONTAINING MEANS OF M VARIABLES
C                IN K GROUPS
C        D     - INPUT MATRIX (M X M) CONTAINING THE INVERSE OF
C                POOLED DISPERSION MATRIX.
C        CMEAN - OUTPUT VECTOR OF LENGTH M CONTAINING COMMON MEANS.
C        V     - OUTPUT VARIABLE CONTAINING GENERALIZED MAHALANOBIS
C                D-SQUARE.
C        C     - OUTPUT MATRIX (M+1 X K) CONTAINING THE COEFFICIENTS
C                OF DISCRIMINANT FUNCTIONS.  THE FIRST POSITION OF
C                EACH COLUMN (FUNCTION) CONTAINS THE VALUE OF THE
C                CONSTANT FOR THAT FUNCTION.
C        P     - OUTPUT VECTOR CONTAINING THE PROBABILITY ASSOCIATED
C                WITH THE LARGEST DISCRIMINANT FUNCTIONS OF ALL CASES
C                IN ALL GROUPS.  CALCULATED RESULTS ARE STORED IN THE
C                MANNER EQUIVALENT TO A 2-DIMENSIONAL AREA (THE
C                FIRST SUBSCRIPT IS CASE NUMBER, AND THE SECOND
C                SUBSCRIPT IS GROUP NUMBER).  VECTOR P HAS LENGTH
C                EQUAL TO THE TOTAL NUMBER OF CASES, T (T = N(1)+N(2)
C                +...+N(K)).
C        LG    - OUTPUT VECTOR CONTAINING THE SUBSCRIPTS OF THE
C                LARGEST DISCRIMINANT FUNCTIONS STORED IN VECTOR P.
C                THE LENGTH OF VECTOR LG IS THE SAME AS THE LENGTH
C                OF VECTOR P.
C
C     REMARKS
C        THE NUMBER OF VARIABLES MUST BE GREATER THAN OR EQUAL TO
C        THE NUMBER OF GROUPS.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        NONE
C
C     METHOD
C        REFER TO 'BMD COMPUTER PROGRAMS MANUAL', EDITED BY W. J.
C        DIXON, UCLA, 1964, AND T. W. ANDERSON, 'INTRODUCTION TO
C        MULTIVARIATE STATISTICAL ANALYSIS', JOHN WILEY AND SONS,
C        1958, SECTION 6.6-6.8.
C
C     ..................................................................
C
      SUBROUTINE DISCR (K,M,N,X,XBAR,D,CMEAN,V,C,P,LG)
      DIMENSION N(1),X(1),XBAR(1),D(1),CMEAN(1),C(1),P(1),LG(1)
C
C        ...............................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
C     DOUBLE PRECISION XBAR,D,CMEAN,V,C,SUM,P,PL
C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  EXP IN STATEMENT
C        250 MUST BE CHANGED TO DEXP.
C
C        ...............................................................
C
C     CALCULATE COMMON MEANS
C
      N1=N(1)
      DO 100 I=2,K
  100 N1=N1+N(I)
      FNT=N1
      DO 110 I=1,K
  110 P(I)=N(I)
      DO 130 I=1,M
      CMEAN(I)=0
      N1=I-M
      DO 120 J=1,K
      N1=N1+M
  120 CMEAN(I)=CMEAN(I)+P(J)*XBAR(N1)
  130 CMEAN(I)=CMEAN(I)/FNT
C
C     CALCULATE GENERALIZED MAHALANOBIS D SQUARE
C
      L=0
      DO 140 I=1,K
      DO 140 J=1,M
      L=L+1
  140 C(L)=XBAR(L)-CMEAN(J)
      V=0.0
      L=0
      DO 160 J=1,M
      DO 160 I=1,M
      N1=I-M
      N2=J-M
      SUM=0.0
      DO 150 IJ=1,K
      N1=N1+M
      N2=N2+M
  150 SUM=SUM+P(IJ)*C(N1)*C(N2)
      L=L+1
  160 V=V+D(L)*SUM
C
C     CALCULATE THE COEFFICIENTS OF DISCRIMINANT FUNCTIONS
C
      N2=0
      DO 190 KA=1,K
      DO 170 I=1,M
      N2=N2+1
  170 P(I)=XBAR(N2)
      IQ=(M+1)*(KA-1)+1
      SUM=0.0
      DO 180 J=1,M
      N1=J-M
      DO 180 L=1,M
      N1=N1+M
  180 SUM=SUM+D(N1)*P(J)*P(L)
      C(IQ)=-(SUM/2.0)
      DO 190 I=1,M
      N1=I-M
      IQ=IQ+1
      C(IQ)=0.0
      DO 190 J=1,M
      N1=N1+M
  190 C(IQ)=C(IQ)+D(N1)*P(J)
C
C     FOR EACH CASE IN EACH GROUP, CALCULATE..
C
C        DISCRIMINANT FUNCTIONS
C
      LBASE=0
      N1=0
      DO 270 KG=1,K
      NN=N(KG)
      DO 260 I=1,NN
      L=I-NN+LBASE
      DO 200 J=1,M
      L=L+NN
  200 D(J)=X(L)
      N2=0
      DO 220 KA=1,K
      N2=N2+1
      SUM=C(N2)
      DO 210 J=1,M
      N2=N2+1
  210 SUM=SUM+C(N2)*D(J)
  220 XBAR(KA)=SUM
C
C        THE LARGEST DISCRIMINANT FUNCTION
C
      L=1
      SUM=XBAR(1)
      DO 240 J=2,K
      IF(SUM-XBAR(J)) 230, 240, 240
  230 L=J
      SUM=XBAR(J)
  240 CONTINUE
C
C        PROBABILITY ASSOCIATED WITH THE LARGEST DISCRIMINANT FUNCTION
C
      PL=0.0
      DO 250 J=1,K
  250 PL=PL+ EXP(XBAR(J)-SUM)
      N1=N1+1
      LG(N1)=L
  260 P(N1)=1.0/PL
  270 LBASE=LBASE+NN*M
C
      RETURN
      END
```

Factor Analysis (see Eigenanalysis)

In the Scientific Subroutine Package, factor analysis is normally performed by calling the following five subroutines in sequence:

1. CORRE - to find means, standard deviations, and correlation matrix
2. EIGEN - to compute eigenvalues and associated eigenvectors of the correlation matrix
3. TRACE - to select the eigenvalues that are greater than or equal to the control value specified by the user
4. LOAD - to compute a factor matrix
5. VARMX - to perform varimax rotation of the factor matrix

The subroutine CORRE works in either of two ways: (1) it expects all observations in core, or (2) it triggers a user-provided input subroutine, DATA, to read one observation at a time into a work area. In either case, the user must provide a subroutine named DATA (see "Subroutines Required" in the comment cards designation of subroutine CORRE).

Subroutine TRACE

This subroutine finds k, the number of eigenvalues that are greater than or equal to the value of a specified constant. The given eigenvalues λ_1, λ_2, ..., λ_m must be arranged in descending order.

Cumulative percentages for these k eigenvalues are:

$$d_j = \sum_{i=1}^{j} \frac{\lambda_i}{m} \qquad (1)$$

where $j = 1, 2, \ldots, k$

m = number of eigenvalues (or variables)

$k \leq m$

```
C
C     ..................................................................
C
C        SUBROUTINE TRACE
C
C        PURPOSE
C           COMPUTE CUMULATIVE PERCENTAGE OF EIGENVALUES GREATER THAN
C           OR EQUAL TO A CONSTANT SPECIFIED BY THE USER.  THIS SUB-
C           ROUTINE NORMALLY OCCURS IN A SEQUENCE OF CALLS TO SUB-
C           ROUTINES CORRE, EIGEN, TRACE, LOAD, AND VARMX IN THE PER-
C           FORMANCE OF A FACTOR ANALYSIS.
C
C        USAGE
C           CALL TRACE (M,R,CON,K,C)
C
C        DESCRIPTION OF PARAMETERS
C           M     - NUMBER OF VARIABLES. M MUST BE > OR = TO 1
C           R     - INPUT MATRIX (SYMMETRIC AND STORED IN COMPRESSED
C                   FORM WITH ONLY UPPER TRIANGLE BY COLUMN IN CORE)
C                   CONTAINING EIGENVALUES IN DIAGONAL.  EIGENVALUES ARE
C                   ARRANGED IN DESCENDING ORDER.  THE ORDER OF MATRIX
C                   IS M BY M.  ONLY M*(M+1)/2 ELEMENTS ARE IN STORAGE.
C                   (STORAGE MODE OF 1)
C           CON   - A CONSTANT USED TO DECIDE HOW MANY EIGENVALUES TO
C                   RETAIN.  CUMULATIVE PERCENTAGE OF EIGENVALUES
C                   WHICH ARE GREATER THAN OR EQUAL TO THIS VALUE IS
C                   CALCULATED.
C           K     - OUTPUT VARIABLE CONTAINING THE NUMBER OF EIGENVALUES
C                   GREATER THAN OR EQUAL TO CON.  (K IS THE NUMBER OF
C                   FACTORS.)
C           C     - OUTPUT VECTOR OF LENGTH M CONTAINING CUMULATIVE
C                   PERCENTAGE OF EIGENVALUES WHICH ARE GREATER THAN
C                   OR EQUAL TO CON.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           EACH EIGENVALUE GREATER THAN OR EQUAL TO CON IS DIVIDED BY M
C           AND THE RESULT IS ADDED TO THE PREVIOUS TOTAL TO OBTAIN
C           THE CUMULATIVE PERCENTAGE FOR EACH EIGENVALUE.
C
C     ..................................................................
C
      SUBROUTINE TRACE (M,R,CON,K,C)
      DIMENSION R(1),C(1)
C
C        ................................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
C     DOUBLE PRECISION R,C
C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        ................................................................
C
      FM=M
      L=0
      DO 100 I=1,M
      L=L+I
  100 C(I)=R(L)
      K=0
C
C     TEST WHETHER I-TH EIGENVALUE IS GREATER
C     THAN OR EQUAL TO THE CONSTANT
C
      DO 110 I=1,M
      IF(C(I)-CON) 120, 105, 105
  105 K=K+1
  110 C(I)=C(I)/FM
C
C     COMPUTE CUMULATIVE PERCENTAGE OF EIGENVALUES
C
  120 DO 130 I=2,K
  130 C(I)=C(I)+C(I-1)
      RETURN
      END
```

Subroutine LOAD

This subroutine calculates the coefficients of each factor by multiplying the elements of each normalized eigenvector by the square root of the corresponding eigenvalue.

$$a_{ij} = v_{ij} \cdot \sqrt{\lambda_j} \qquad (1)$$

where $i = 1, 2, \ldots, m$ are variables

$j = 1, 2, \ldots, k$ are eigenvalues retained (see the subroutine TRACE)

$k \leq m$

```
                                                                        LOAD  10
 ......................................................................LOAD  20
                                                                        LOAD  30
      SUBROUTINE LOAD                                                   LOAD  40
                                                                        LOAD  50
      PURPOSE                                                           LOAD  60
         COMPUTE A FACTOR MATRIX (LOADING) FROM EIGENVALUES AND         LOAD  70
         ASSOCIATED EIGENVECTORS.  THIS SUBROUTINE NORMALLY OCCURS      LOAD  80
         IN A SEQUENCE OF CALLS TO SUBROUTINES CORRE, EIGEN, TRACE,     LOAD  90
         LOAD, AND VARMX IN THE PERFORMANCE OF A FACTOR ANALYSIS.       LOAD 100
                                                                        LOAD 110
      USAGE                                                             LOAD 120
         CALL LOAD (M,K,R,V)                                            LOAD 130
                                                                        LOAD 140
      DESCRIPTION OF PARAMETERS                                         LOAD 150
         M      - NUMBER OF VARIABLES.                                  LOAD 160
         K      - NUMBER OF FACTORS. K MUST BE GREATER THAN OR EQUAL    LOAD 170
                  TO 1 AND LESS THAN OR EQUAL TO M.                     LOAD 171
         R      - A MATRIX (SYMMETRIC AND STORED IN COMPRESSED FORM     LOAD 180
                  WITH ONLY UPPER TRIANGLE BY COLUMN IN CORE) CON-      LOAD 190
                  TAINING EIGENVALUES IN DIAGONAL.  EIGENVALUES ARE     LOAD 200
                  ARRANGED IN DESCENDING ORDER, AND FIRST K             LOAD 210
                  EIGENVALUES ARE USED BY THIS SUBROUTINE.  THE ORDER   LOAD 220
                  OF MATRIX R IS M BY M.  ONLY M*(M+1)/2 ELEMENTS ARE   LOAD 230
                  IN STORAGE.  (STORAGE MODE OF 1)                      LOAD 240
         V      - WHEN THIS SUBROUTINE IS CALLED, MATRIX V (M X M)      LOAD 250
                  CONTAINS EIGENVECTORS COLUMNWISE.  UPON RETURNING TOLOAD 260
                  THE CALLING PROGRAM, MATRIX V CONTAINS A FACTOR       LOAD 270
                  MATRIX (M X K).                                       LOAD 280
                                                                        LOAD 290
      REMARKS                                                           LOAD 300
         NONE                                                           LOAD 310
                                                                        LOAD 320
      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     LOAD 330
         NONE                                                           LOAD 340
                                                                        LOAD 350
      METHOD                                                            LOAD 360
         NORMALIZED EIGENVECTORS ARE CONVERTED TO THE FACTOR PATTERN    LOAD 370
         BY MULTIPLYING THE ELEMENTS OF EACH VECTOR BY THE SQUARE       LOAD 380
         ROOT OF THE CORRESPONDING EIGENVALUE.                          LOAD 390
                                                                        LOAD 400
 ......................................................................LOAD 410
                                                                        LOAD 420
      SUBROUTINE LOAD (M,K,R,V)                                         LOAD 430
      DIMENSION R(1),V(1)                                               LOAD 440
                                                                        LOAD 450
         ...............................................................LOAD 460
                                                                        LOAD 470
         IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  LOAD 480
         C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      LOAD 490
         STATEMENT WHICH FOLLOWS.                                       LOAD 500
                                                                        LOAD 510
      DOUBLE PRECISION R,V,SC                                           LOAD 520
                                                                        LOAD 530
         THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    LOAD 540
         APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      LOAD 550
         ROUTINE.                                                       LOAD 560
                                                                        LOAD 570
         THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      LOAD 580
         CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENT LOAD 590
         150 MUST BE CHANGED TO DSQRT.                                  LOAD 600
                                                                        LOAD 610
         ...............................................................LOAD 620
                                                                        LOAD 630
      L=0                                                               LOAD 640
      JJ=0                                                              LOAD 650
      DO 160 J=1,K                                                      LOAD 660
      JJ=JJ+J                                                           LOAD 670
  150 SC= SQRT(R(JJ))                                                   LOAD 680
      DO 160 I=1,M                                                      LOAD 690
      L=L+1                                                             LOAD 700
  160 V(L)=SC*V(L)                                                      LOAD 710
      RETURN                                                            LOAD 720
      END                                                               LOAD 730
```

Subroutine VARMX

This subroutine performs orthogonal rotations on an m by k factor matrix such that:

$$\sum_j \left\{ m \sum_i \left(a_{ij}^2 / h_i^2 \right)^2 - \left[\sum_i \left(a_{ij}^2 / h_i^2 \right) \right]^2 \right\} \qquad (1)$$

is a maximum, where $i = 1, 2, \ldots, m$ are variables, $j = 1, 2, \ldots, k$ are factors, a_{ij} is the loading for the ith variable on the jth factor, and h_i^2 is the communality of the ith variable defined below.

Communalities:

$$h_i^2 = \sum_{j=1}^{k} a_{ij}^2 \qquad (2)$$

where $i = 1, 2, \ldots, m$

Normalized factor matrix:

$$b_{ij} = a_{ij} \Big/ \sqrt{h_i^2} \qquad (3)$$

where $i = 1, 2, \ldots, m$

$j = 1, 2, \ldots, k$

Variance for factor matrix:

$$V_c = \sum_j \left\{ \left[m \sum_i \left(b_{ij}^2 \right)^2 - \left(\sum_i b_{ij}^2 \right)^2 \right] \Big/ m^2 \right\} \qquad (4)$$

where $c = 1, 2, \ldots$ (iteration cycle)

Convergence test:

$$\text{If} \quad V_c - V_{c-1} \leq 10^{-7} \qquad (5)$$

four successive times, the program stops rotation and performs the equation (28). Otherwise, the program repeats rotation of factors until the convergence test is satisfied.

Rotation of two factors:

The subroutine rotates two normalized factors (b_{ij}) at a time. 1 with 2, 1 with 3, ..., 1 with k, 2 with 3, ..., 2 with k, ..., k - 1 with k. This constitutes one iteration cycle.

Assuming that x and y are factors to be rotated, where x is the lower-numbered or left-hand factor, the following notation for rotating these two factors is used:

$$\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ x_m & y_m \end{bmatrix} \cdot \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} = \begin{bmatrix} X_1 & Y_1 \\ X_2 & Y_2 \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ X_m & Y_m \end{bmatrix} \quad (6)$$

where x_i and y_i are presently available normalized loadings, and X_i and Y_i, the desired normalized loadings, are functions of ϕ, the angle of rotation. The computational steps are (a) through (e) below.

(a) Calculation of NUM and DEN:

$$A = \sum_i (x_i + y_i)\ (x_i - y_i)$$

$$B = 2 \sum_i x_i y_i$$

$$C = \sum_i \left[(x_i + y_i)\ (x_i - y_i) + 2x_i y_i \right]$$

$$\left[(x_i + y_i)\ (x_i - y_i) - 2x_i y_i \right]$$

$$D = 4 \sum_i (x_i + y_i)\ (x_i - y_i)\ x_i y_i \quad (7)$$

$$NUM = D - 2AB/m$$

$$DEN = C - [\ (A + B)\ (A - B)\ ]\ /m$$

(b) Comparison of NUM and DEN:

The following four cases may arise.

NUM < DEN, go to (b1) below

NUM > DEN, go to (b2) below

(NUM + DEN) $\geq \epsilon$*, go to (b3) below

(NUM + DEN) $< \epsilon$, skip to the next rotation

* ϵ is a small tolerance factor.

(b1) $\tan 4\theta = |NUM|/|DEN|$ (8)

If $\tan 4\theta < \epsilon$ and

(i) DEN is positive, skip to the next rotation.

(ii) DEN is negative, set $\cos\phi = \sin\phi = (\sqrt{2})/2$ and go to (e) below.

If $\tan 4\theta \geq \epsilon$, calculate:

$$\cos 4\theta = 1\Big/\sqrt{1 + \tan^2 4\theta} \quad (9)$$

$$\sin 4\theta = \tan 4\theta \cdot \cos 4\theta \quad (10)$$

and go to (c) below.

(b2) $\text{ctn}\, 4\theta = |NUM|/|DEN|$ (11)

If $\text{ctn} 4\theta < \epsilon$, set $\cos 4\theta = 0$ and $\sin 4\theta = 1$. Go to (c) below.

If $\text{ctn} 4\theta \geq \epsilon$, calculate:

$$\sin 4\theta = 1/\sqrt{1 + \text{ctn}^2 4\theta} \quad (12)$$

$$\cos 4\theta = \text{ctn} 4\theta \cdot \sin 4\theta \quad (13)$$

and go to (c) below.

(b3) Set $\cos 4\theta = \sin 4\theta = (\sqrt{2})/2$ and go to (c) below.

(c) Determining $\cos\theta$ and $\sin\theta$:

$$\cos 2\theta = \sqrt{(1 + \cos 4\theta)/2} \quad (14)$$

$$\sin 2\theta = \sin 4\theta / 2\cos 2\theta \quad (15)$$

$$\cos\theta = \sqrt{(1 + \cos 2\theta)/2} \quad (16)$$

$$\sin\theta = \sin 2\theta / 2\cos\theta \quad (17)$$

(d) Determining $\cos\phi$ and $\sin\phi$:

(d1) If DEN is positive, set

$$\cos\phi = \cos\theta \quad (18)$$

$$\sin\phi = \sin\theta \quad (19)$$

and go to (d2) below.

If DEN is negative, calculate

$$\cos\phi = \frac{\sqrt{2}}{2} \cos\theta + \frac{\sqrt{2}}{2} \sin\theta \quad (20)$$

$$\sin\phi = \left| \frac{\sqrt{2}}{2} \cos\theta - \frac{\sqrt{2}}{2} \sin\theta \right| \quad (21)$$

and go to (d2) below.

(d2) If NUM is positive, set

$$\cos\phi = \left| \cos\phi \right| \quad (22)$$

$$\sin\phi = \left| \sin\phi \right| \quad (23)$$

and go to (e) below.

If NUM is negative, set

$$\cos\phi = \left| \cos\phi \right| \quad (24)$$

$$\sin\phi = -\left| \sin\phi \right| \quad (25)$$

(e) Rotation:

$$X_i = x_i \cos\phi + y_i \sin\phi \quad (26)$$

$$Y_i = x_i \sin\phi + y_i \cos\phi \quad (27)$$

where $i = 1, 2, \ldots, m$

After one cycle of $k(k-1)/2$ rotations is completed, the subroutine goes back to calculate the variance for the factor matrix by equation (4).

Denormalization:

$$a_{ij} = b_{ij} \cdot h_i \quad (28)$$

where $i = 1, 2, \ldots, m$

$j = 1, 2, \ldots, k$

Check on communalities:

Final communalities $$f_i^2 = \sum_{j=1}^{k} a_{ij}^2 \quad (29)$$

Difference $$d_i = h_i^2 - f_i^2 \quad (30)$$

where $i = 1, 2, \ldots, m$

```
C                                                                       VARM  10
C     ..................................................................VARM  20
C                                                                       VARM  30
C        SUBROUTINE VARMX                                               VARM  40
C                                                                       VARM  50
C        PURPOSE                                                        VARM  60
C           PERFORM ORTHOGONAL ROTATIONS OF A FACTOR MATRIX.  THIS      VARM  70
C           SUBROUTINE NORMALLY OCCURS IN A SEQUENCE OF CALLS TO SUB-   VARM  80
C           ROUTINES CORRE, EIGEN, TRACE, LOAD, VARMX IN THE PERFORMANCEVARM  90
C           OF A FACTOR ANALYSIS.                                       VARM 100
C                                                                       VARM 110
C        USAGE                                                          VARM 120
C           CALL VARMX (M,K,A,NC,TV,H,F,D,IER)                          VARM 130
C                                                                       VARM 140
C        DESCRIPTION OF PARAMETERS                                      VARM 150
C           M     - NUMBER OF VARIABLES AND NUMBER OF ROWS OF MATRIX A. VARM 160
C           K     - NUMBER OF FACTORS.                                  VARM 170
C           A     - INPUT IS THE ORIGINAL FACTOR MATRIX, AND OUTPUT IS  VARM 180
C                   THE ROTATED FACTOR MATRIX.  THE ORDER OF MATRIX A   VARM 190
C                   IS M X K.                                           VARM 200
C           NC    - OUTPUT VARIABLE CONTAINING THE NUMBER OF ITERATION  VARM 210
C                   CYCLES PERFORMED.                                   VARM 220
C           TV    - OUTPUT VECTOR CONTAINING THE VARIANCE OF THE FACTOR VARM 230
C                   MATRIX FOR EACH ITERATION CYCLE.  THE VARIANCE PRIORVARM 240
C                   TO THE FIRST ITERATION CYCLE IS ALSO CALCULATED.    VARM 250
C                   THIS MEANS THAT NC+1 VARIANCES ARE STORED IN VECTOR VARM 260
C                   TV.  MAXIMUM NUMBER OF ITERATION CYCLES ALLOWED IN  VARM 270
C                   THIS SUBROUTINE IS 50.  THEREFORE, THE LENGTH OF    VARM 280
C                   VECTOR TV IS 51.                                    VARM 290
C           H     - OUTPUT VECTOR OF LENGTH M CONTAINING THE ORIGINAL   VARM 300
C                   COMMUNALITIES.                                      VARM 310
C           F     - OUTPUT VECTOR OF LENGTH M CONTAINING THE FINAL      VARM 320
C                   COMMUNALITIES.                                      VARM 330
C           D     - OUTPUT VECTOR OF LENGTH M CONTAINING THE DIFFERENCESVARM 340
C                   BETWEEN THE ORIGINAL AND FINAL COMMUNALITIES.       VARM 350
C           IER   - ERROR INDICATOR                                     VARM 351
C                   IER=0 - NO ERROR                                    VARM 352
C                   IER=1 - CONVERGENCE WAS NOT ACHIEVED IN 50 CYCLES   VARM 353
C                           OF ROTATION                                 VARM 354
C                                                                       VARM 360
C        REMARKS                                                        VARM 370
C           IF VARIANCE COMPUTED AFTER EACH ITERATION CYCLE DOES NOT    VARM 380
C           INCREASE FOR FOUR SUCCESSIVE TIMES, THE SUBROUTINE STOPS    VARM 390
C           ROTATION.                                                   VARM 400
C                                                                       VARM 410
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  VARM 420
C           NONE                                                        VARM 430
C                                                                       VARM 440
C        METHOD                                                         VARM 450
C           KAISER'S VARIMAX ROTATION AS DESCRIBED IN 'COMPUTER PROGRAM VARM 460
C           FOR VARIMAX ROTATION IN FACTOR ANALYSIS' BY THE SAME AUTHOR,VARM 470
C           EDUCATIONAL AND PSYCHOLOGICAL MEASUREMENT, VOL XIX, NO. 3,  VARM 480
C           1959.                                                       VARM 490
C                                                                       VARM 500
C     ..................................................................VARM 510
C                                                                       VARM 520
      SUBROUTINE VARMX (M,K,A,NC,TV,H,F,D,IER)                          VARM 530
      DIMENSION A(1),TV(1),H(1),F(1),D(1)                               VARM 540
C                                                                       VARM 550
C     ..................................................................VARM 560
C                                                                       VARM 570
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  VARM 580
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      VARM 590
C                                                                       VARM 600
C     DOUBLE PRECISION A,TV,H,F,D,TVLT,CONS,AA,BB,CC,DD,U,T,B,COS4T,    VARM 610
C    1                 SIN4T,TAN4T,SINP,COSP,CTN4T,COS2T,SIN2T,COST,SINTVARM 620
C                                                                       VARM 630
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    VARM 640
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      VARM 650
C        ROUTINE.                                                       VARM 660
C                                                                       VARM 670
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      VARM 680
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENTSVARM 690
C        115, 290, 330, 350, AND 355 MUST BE CHANGED TO DSQRT.  ABS IN  VARM 700
C        STATEMENTS 260, 320, AND 375 MUST BE CHANGED TO DABS.          VARM 710
C                                                                       VARM 720
C     ..................................................................VARM 730
C                                                                       VARM 740
C     INITIALIZATION                                                    VARM 750
C                                                                       VARM 760
      IER=0                                                             VARM 761
      EPS=0.00116                                                       VARM 770
      TVLT=0.0                                                          VARM 780
      LL=K-1                                                            VARM 790
      NV=1                                                              VARM 800
      NC=0                                                              VARM 810
      FN=M                                                              VARM 820
      FFN=FN*FN                                                         VARM 830
      CONS=0.707106                                                     VARM 840
C                                                                       VARM 850
C     CALCULATE ORIGINAL COMMUNALITIES                                  VARM 860
C                                                                       VARM 870
      DO 110 I=1,M                                                      VARM 880
      H(I)=0.0                                                          VARM 890
      DO 110 J=1,K                                                      VARM 900
      L=M*(J-1)+I                                                       VARM 910
  110 H(I)=H(I)+A(L)*A(L)                                               VARM 920
C                                                                       VARM 930
C     CALCULATE NORMALIZED FACTOR MATRIX                                VARM 940
C                                                                       VARM 950
      DO 120 I=1,M                                                      VARM 960
  115 H(I)= SQRT(H(I))                                                  VARM 970
      DO 120 J=1,K                                                      VARM 980
      L=M*(J-1)+I                                                       VARM 990
  120 A(L)=A(L)/H(I)                                                    VARM1000
      GO TO 132                                                         VARM1010
C                                                                       VARM1020
C     CALCULATE VARIANCE FOR FACTOR MATRIX                              VARM1030
C                                                                       VARM1040
  130 NV=NV+1                                                           VARM1050
      TVLT=TV(NV-1)                                                     VARM1060
  132 TV(NV)=0.0                                                        VARM1070
      DO 150 J=1,K                                                      VARM1080
      AA=0.0                                                            VARM1090
      BB=0.0                                                            VARM1100
      LB=M*(J-1)                                                        VARM1110
      DO 140 I=1,M                                                      VARM1120
      L=LB+I                                                            VARM1130
      CC=A(L)*A(L)                                                      VARM1140
      AA=AA+CC                                                          VARM1150
  140 BB=BB+CC*CC                                                       VARM1160
  150 TV(NV)=TV(NV)+(FN*BB-AA*AA)/FFN                                   VARM1170
      IF(NV-51)160,155,155                                              VARM1180
  155 IER=1                                                             VARM1181
      GO TO 430                                                         VARM1182
C                                                                       VARM1190
C     PERFORM CONVERGENCE TEST                                          VARM1200
C                                                                       VARM1210
```

```
  16C IF(ITV(NV)-IVLT)-(1.E-7)) 17C, 17C, 19C
  17C NC=NC+1
      IF(NC-3) 15C, 15C, 43C
C
C     ROTATION OF THE FACTORS CONTINUES UP TO
C     THE STATEMENT 12C.
C
  15C DO 42C J=1,LL
      L1=M*(J-1)
      II=J+1
C
C        CALCULATE NUM AND DEN
C
      DO 42C K1=II,K
      L2=M*(K1-1)
      AA=0.0
      BB=0.0
      CC=0.0
      DD=0.0
      DO 23C I=1,M
      L3=L1+I
      L4=L2+I
      L=(A(L3)+A(L4))*(A(L3)-A(L4))
      T=A(L3)*A(L4)
      T=T+T
      CC=CC+(L+T)*(L-T)
      DD=DD+2.0*L*T
      AA=AA+L
  23C BB=BB+T
      T=DD-2.0*AA*BB/FN
      B=CC-(AA*AA-BB*BB)/FN
C
C        COMPARISON OF NUM AND DEN
C
      IF(T-B) 26C, 24C, 32C
  24C IF((T+B)-EPS) 42C, 25C, 25C
C
C        NUM + DEN IS GREATER THAN OR EQUAL TO THE
C        TOLERANCE FACTOR
C
  25C COS4T=COSS
      SIN4T=COSS
      GO TO 35C
C
C        NUM IS LESS THAN DEN
C
  26C TAN4T= ABS(T)/ ABS(B)
      IF(TAN4T-EPS) 30C, 29C, 29C
  29C COS4T=1.0/ SQRT(1.0+TAN4T*TAN4T)
      SIN4T=TAN4T*COS4T
      GO TO 35C
  30C IF(B) 31C, 42C, 42C
  31C SINP=COSS
      COSP=COSS
      GO TO 40C
C
C        NUM IS GREATER THAN DEN
C
  32C CTN4T= ABS(T/B)
      IF(CTN4T-EPS) 34C, 33C, 33C
  33C SIN4T=1.0/ SQRT(1.0+CTN4T*CTN4T)
      COS4T=CTN4T*SIN4T
      GO TO 35C
  34C COS4T=0.0
      SIN4T=1.0
C
C        DETERMINE COS THETA AND SIN THETA
C
  35C COS2T= SQRT((1.0+COS4T)/2.0)
      SIN2T=SIN4T/(2.0*COS2T)
  355 COS1= SQRT((1.0+COS2T)/2.0)
      SIN1=SIN2T/(2.0*COS1)
C
C        DETERMINE COS PHI AND SIN PHI
C
      IF(B) 37C, 37C, 36C
  36C COSP=COS1
      SINP=SIN1
      GO TO 38C
  37C COSP=COS1*COS1-COS*SIN1
  375 SINP= ABS(COS1*COS1-COS*SIN1)
  38C IF(T) 39C, 39C, 40C
  39C SINP=-SINP
C
C        PERFORM ROTATION
C
  40C DO 41C I=1,M
      L3=L1+I
      L4=L2+I
      AA=A(L3)*COSP+A(L4)*SINP
      A(L4)=-A(L3)*SINP+A(L4)*COSP
  41C A(L3)=AA
  42C CONTINUE
      GO TO 13C
C
C     DENORMALIZE VARIMAX LOADINGS
C
  43C DO 44C I=1,M
      DO 44C J=1,K
      L=M*(J-1)+I
  44C A(L)=A(L)*F(I)
C
C     CHECK ON COMMUNALITIES
C
      NC=NV-1
      DO 45C I=1,M
  45C F(I)=F(I)*F(I)
      DO 47C I=1,M
      F(I)=0.0
      DO 46C J=1,K
      L=M*(J-1)+I
  46C F(I)=F(I)+A(L)*A(L)
  47C F(I)=F(I)-F(I)
      RETURN
      END
```

Time Series (see Smoothing)

Subroutine AUTO

This subroutine calculates the autocovariances for lags 0, 1, 2, ..., (L-1), given a time series of observations $A_1, A_2, \ldots, A_n$ and a number L.

$$R_j = \frac{1}{n-j+1} \sum_{i=1}^{n-j+1} (A_i - \text{AVER})\,(A_{i+j-1} - \text{AVER}) \qquad (1)$$

$$\text{where AVER} = \frac{1}{n} \sum_{i=1}^{n} A_i$$

n = number of observations in time series A

j = 1, 2, 3, ..., L represent time lags 0, 1, 2, ..., (L-1)

```
C                                                                       AUTO  10
C     ..................................................................AUTO  20
C                                                                       AUTO  30
C        SUBROUTINE AUTO                                                AUTO  40
C                                                                       AUTO  50
C        PURPOSE                                                        AUTO  60
C           TO FIND AUTOCOVARIANCES OF SERIES A FOR LAGS 0 TO L-1.      AUTO  70
C                                                                       AUTO  80
C        USAGE                                                          AUTO  90
C           CALL AUTO (A,N,L,R)                                         AUTO 100
C                                                                       AUTO 110
C        DESCRIPTION OF PARAMETERS                                      AUTO 120
C           A     - INPUT VECTOR OF LENGTH N CONTAINING THE TIME SERIES AUTO 130
C                   WHOSE AUTOCOVARIANCE IS DESIRED.                    AUTO 140
C           N     - LENGTH OF THE VECTOR A.                             AUTO 150
C           L     - AUTOCOVARIANCE IS CALCULATED FOR LAGS OF 0, 1, 2,...AUTO 160
C                   L-1.                                                AUTO 170
C           R     - OUTPUT VECTOR OF LENGTH L CONTAINING AUTOCOVARIANCES AUTO 180
C                   OF SERIES A.                                        AUTO 190
C                                                                       AUTO 200
C        REMARKS                                                        AUTO 210
C           THE LENGTH OF R IS DIFFERENT FROM THE LENGTH OF A.  N MUST  AUTO 220
C           BE GREATER THAN L.  IF NOT, R(1) IS SET TO ZERO AND RETURN  AUTO 230
C           IS MADE TO THE CALLING PROGRAM.                             AUTO 240
C                                                                       AUTO 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  AUTO 260
C           NONE                                                        AUTO 270
C                                                                       AUTO 280
C        METHOD                                                         AUTO 290
C           DESCRIBED IN R.B. BLACKMAN AND J.W. TUKEY, "THE MEASURMENT  AUTO 300
C        OF POWER SPECTRA", DOVER PUBLICATIONS INC., NEW YORK, 1959.    AUTO 310
C                                                                       AUTO 320
C     ..................................................................AUTO 330
C                                                                       AUTO 340
      SUBROUTINE AUTO (A,N,L,R)                                         AUTO 350
      DIMENSION A(1),R(1)                                               AUTO 360
C                                                                       AUTO 370
C     CALCULATE AVERAGE OF TIME SERIES A                                AUTO 380
C                                                                       AUTO 390
      AVER=0.0                                                          AUTO 400
      IF(N-L) 50,50,100                                                 AUTO 410
   50 R(1)=0.0                                                          AUTO 420
      RETURN                                                            AUTO 430
  100 DO 110 I=1,N                                                      AUTO 440
  110 AVER=AVER+A(I)                                                    AUTO 450
      FN=N                                                              AUTO 460
      AVER=AVER/FN                                                      AUTO 470
C                                                                       AUTO 480
C     CALCULATE AUTOCOVARIANCES                                         AUTO 490
C                                                                       AUTO 500
      DO 130 J=1,L                                                      AUTO 510
      NJ=N-J+1                                                          AUTO 520
      SUM=0.0                                                           AUTO 530
      DO 120 I=1,NJ                                                     AUTO 540
      IJ=I+J-1                                                          AUTO 550
  120 SUM=SUM+(A(I)-AVER)*(A(IJ)-AVER)                                  AUTO 560
      FNJ=NJ                                                            AUTO 570
  130 R(J)=SUM/FNJ                                                      AUTO 580
      RETURN                                                            AUTO 590
      END                                                               AUTO 600
```

Subroutine CROSS

This subroutine calculates the cross covariances of series B lagging and leading A, given two time series $A_1, A_2, \ldots, A_n$ and $B_1, B_2, \ldots, B_n$ and given a number L.

(a) B lags A:

$$R_j = \frac{1}{n-j+1} \sum_{i=1}^{n-j+1} (A_i - AVERA)(B_{i+j-1} - AVERB) \tag{1}$$

(b) B leads A:

$$S_j = \frac{1}{n-j+1} \sum_{i=1}^{n-j+1} (A_{i+j-1} - AVERA)(B_i - AVERB) \tag{2}$$

where $AVERA = \frac{1}{n} \sum_{i=1}^{n} A_i$

$AVERB = \frac{1}{n} \sum_{i=1}^{n} B_i$

n = number of observations in each series

j = 1, 2, ..., L represent time lags (or leads) of 0, 1, 2, ..., (L-1)

```
C
C     ..................................................................
C
C        SUBROUTINE CROSS
C
C        PURPOSE
C           TO FIND THE CROSSCOVARIANCES OF SERIES A WITH SERIES B
C           (WHICH LEADS AND LAGS A).
C
C        USAGE
C           CALL CROSS (A,B,N,L,R,S)
C
C        DESCRIPTION OF PARAMETERS
C           A     - INPUT VECTOR OF LENGTH N CONTAINING FIRST TIME
C                   SERIES.
C           B     - INPUT VECTOR OF LENGTH N CONTAINING SECOND TIME
C                   SERIES.
C           N     - LENGTH OF SERIES A AND B.
C           L     - CROSSCOVARIANCE IS CALCULATED FOR LAGS AND LEADS OF
C                   0, 1, 2,..., L-1.
C           R     - OUTPUT VECTOR OF LENGTH L CONTAINING CROSSCOVARI-
C                   ANCES OF A WITH B, WHERE B LAGS A.
C           S     - OUTPUT VECTOR OF LENGTH L CONTAINING CROSSCOVARI-
C                   ANCES OF A WITH B, WHERE B LEADS A.
C
C        REMARKS
C           N MUST BE GREATER THAN L.  IF NOT, R(1) AND S(1) ARE SET TO
C           ZERO AND RETURN IS MADE TO THE CALLING PROGRAM.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DESCRIBED IN R.B. BLACKMAN AND J.W. TUKEY, 'THE MEASUREMENT
C        OF POWER SPECTRA', DOVER PUBLICATIONS INC., NEW YORK, 1959.
C
C     ..................................................................
C
      SUBROUTINE CROSS (A,B,N,L,R,S)
      DIMENSION A(1),B(1),R(1),S(1)
C
C     CALCULATE AVERAGES OF SERIES A AND B
C
      FN=N
      AVERA=0.0
      AVERB=0.0
      IF(N-L)50,50,100
   50 R(1)=0.0
      S(1)=0.0
      RETURN
  100 DO 110 I=1,N
      AVERA=AVERA+A(I)
  110 AVERB=AVERB+B(I)
      AVERA=AVERA/FN
      AVERB=AVERB/FN
C
C     CALCULATE CROSSCOVARIANCES OF SERIES A AND B
C
      DO 130 J=1,L
      NJ=N-J+1
      SUMR=0.0
      SUMS=0.0
      DO 120 I=1,NJ
      IJ=I+J-1
      SUMR=SUMR+(A(I)-AVERA)*(B(IJ)-AVERB)
  120 SUMS=SUMS+(A(IJ)-AVERA)*(B(I)-AVERB)
      FNJ=NJ
      R(J)=SUMR/FNJ
  130 S(J)=SUMS/FNJ
      RETURN
      END
```

Subroutine SMO

This subroutine calculates the smoothed or filtered series, given a time series $A_1, A_2, \ldots, A_n$, a selection integer L, and a weighting series $W_1, W_2, \ldots, W_m$.

$$R_i = \sum_{j=1}^{m} A_p \cdot W_j \qquad (1)$$

where $p = j \cdot L - L + k$

$k = i - IL + 1$

$i = IL$ to IH

$$IL = \frac{L(m-1)}{2} + 1 \qquad (2)$$

$$IH = n - \frac{L(m-1)}{2} \qquad (3)$$

L = a given selection integer. For example, L = 4 applies weights to every 4th item of the time series.

m = number of weights. Must be an odd integer. (If m is an even integer, any fraction resulting from the calculation of $\frac{L(m-1)}{2}$ in (2) and (3) above will be truncated.)

n = number of items in the time series

From IL to IH elements of the vector R are filled with the smoothed series, and other elements with zeros.

```
C                                                                       SMO  10
C     ..................................................................SMO  20
C                                                                       SMO  30
C                                                                       SMO  40
C        SUBROUTINE SMO                                                 SMO  50
C                                                                       SMO  60
C        PURPOSE                                                        SMO  70
C           TO SMOOTH OR FILTER SERIES A BY WEIGHTS W.                  SMO  80
C                                                                       SMO  90
C        USAGE                                                          SMO 100
C           CALL SMO (A,N,W,M,L,R)                                      SMO 110
C                                                                       SMO 120
C        DESCRIPTION OF PARAMETERS                                      SMO 130
C           A - INPUT VECTOR OF LENGTH N CONTAINING TIME SERIES DATA.   SMO 140
C           N - LENGTH OF SERIES A.                                     SMO 150
C           W - INPUT VECTOR OF LENGTH M CONTAINING WEIGHTS.            SMO 160
C           M - NUMBER OF ITEMS IN WEIGHT VECTOR.  M MUST BE AN ODD     SMO 170
C               INTEGER.  (IF M IS AN EVEN INTEGER, ANY FRACTION        SMO 180
C               RESULTING FROM THE CALCULATION OF (L*(M-1))/2 IN (1)    SMO 190
C               AND (2) BELOW WILL BE TRUNCATED.)                       SMO 200
C           L - SELECTION INTEGER.  FOR EXAMPLE, L=12 MEANS THAT WEIGHTSSMO 210
C               ARE APPLIED TO EVERY 12-TH ITEM OF A.  L=1 APPLIES      SMO 220
C               WEIGHTS TO SUCCESSIVE ITEMS OF A.  FOR MONTHLY DATA,    SMO 230
C               L=12 GIVES YEAR-TO-YEAR AVERAGES AND L=1 GIVES MONTH-TO-SMO 240
C               MONTH AVERAGES.                                         SMO 250
C           R - OUTPUT VECTOR OF LENGTH N.  FROM IL TO IH ELEMENTS OF   SMO 260
C               THE VECTOR R ARE FILLED WITH THE SMOOTHED SERIES AND    SMO 270
C               OTHER ELEMENTS WITH ZERO, WHERE                         SMO 280
C                     IL=(L*(M-1))/2+1  ................ (1)            SMO 290
C                     IH=N-(L*(M-1))/2  ................ (2)            SMO 300
C                                                                       SMO 310
C        REMARKS                                                        SMO 320
C           N MUST BE GREATER THAN OR EQUAL TO THE PRODUCT OF L*M.      SMO 330
C                                                                       SMO 340
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SMO 350
C           NONE                                                        SMO 360
C                                                                       SMO 370
C        METHOD                                                         SMO 380
C           REFER TO THE ARTICLE 'FORTRAN SUBROUTINES FOR TIME SERIES   SMO 390
C           ANALYSIS', BY J. R. HEALY AND B. P. BOGERT, COMMUNICATIONS  SMO 400
C           OF ACM, V.6, NO.1, JANUARY, 1963.                           SMO 410
C                                                                       SMO 420
C     ..................................................................SMO 430
C                                                                       SMO 440
      SUBROUTINE SMO (A,N,W,M,L,R)                                      SMO 450
      DIMENSION A(1),W(1),R(1)                                          SMO 460
C                                                                       SMO 470
C     INITIALIZATION                                                    SMO 480
C                                                                       SMO 490
      DO 110 I=1,N                                                      SMO 500
  110 R(I)=0.0                                                          SMO 510
      IL=(L*(M-1))/2+1                                                  SMO 520
      IH=N-(L*(M-1))/2                                                  SMO 530
C                                                                       SMO 540
C     SMOOTH SERIES A BY WEIGHTS W                                      SMO 550
C                                                                       SMO 560
      DO 120 I=IL,IH                                                    SMO 570
      K=I-IL+1                                                          SMO 580
      DO 120 J=1,M                                                      SMO 590
      IP=(J*L)-L+K                                                      SMO 600
  120 R(I)=R(I)+A(IP)*W(J)                                              SMO 610
      RETURN                                                            SMO 620
      END
```

Subroutine EXSMO

This subroutine calculates a smoothed series S_1, S_2, ..., S_{NX}, given time series X_1, X_2, ..., X_{NX} and a smoothing constant α. Also, at the end of the computation, the coefficients A, B, and C are given for the expression $A + B(T) + C(T)^2/2$. This expression can be used to find estimates of the smoothed series for a given number of time periods, T, ahead.

The subroutine has the following two stages for $i = 1, 2, \ldots, NX$, starting with A, B, and C either given by the user or provided automatically by the subroutine (see below).

(a) Find S_i for one period ahead:

$$S_i = A + B + 0.5C \tag{1}$$

(b) Update coefficients A, B, and C:

$$A = X_i + (1 - \alpha)^3 (S_i - X_i) \tag{2}$$

$$B = B + C - 1.5 (\alpha^2) (2 - \alpha) (S_i - X_i) \tag{3}$$

$$C = C - (\alpha^3) (S_i - X_i) \tag{4}$$

where α = smoothing constant specified by the user

$(0.0 < \alpha < 1.0)$

If coefficients A, B, and C are not all zero (0.0), take given values as initial values. However, if $A = B = C = 0.0$, generate initial values of A, B, and C as follows:

$$C = X_1 - 2X_2 + X_3 \tag{5}$$

$$B = X_2 - X_1 - 1.5C \tag{6}$$

$$A = X_1 - B - 0.5C \tag{7}$$

```
C
C     ..................................................................
C
C        SUBROUTINE EXSMO
C
C        PURPOSE
C           TO FIND THE TRIPLE EXPONENTIAL SMOOTHED SERIES S OF THE
C           GIVEN SERIES X.
C
C        USAGE
C           CALL EXSMO (X,NX,AL,A,B,C,S)
C
C        DESCRIPTION OF PARAMETERS
C           X      - INPUT VECTOR OF LENGTH NX CONTAINING TIME SERIES
C                    DATA WHICH IS TO BE EXPONENTIALLY SMOOTHED.
C           NX     - THE NUMBER OF ELEMENTS IN X.
C           AL     - SMOOTHING CONSTANT, ALPHA.  AL MUST BE GREATER THAN
C                    ZERO AND LESS THAN ONE.
C           A,B,C  - COEFFICIENTS OF THE PREDICTION EQUATION WHERE S IS
C                    PREDICTED T PERIODS HENCE BY
C                              A + B*T + C*T*T/2.
C                    AS INPUT-- IF A=B=C=0, PROGRAM WILL PROVIDE INITIAL
C                    VALUES.  IF AT LEAST ONE OF A,B,C IS NOT ZERO,
C                    PROGRAM WILL TAKE GIVEN VALUES AS INITIAL VALUES.
C                    AS OUTPUT-- A,B,C CONTAIN LATEST, UPDATED COEFFI-
C                    CIENTS OF PREDICTION.
C           S      - OUTPUT VECTOR OF LENGTH NX CONTAINING TRIPLE
C                    EXPONENTIALLY SMOOTHED TIME SERIES.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           REFER TO R. G. BROWN, 'SMOOTHING, FORECASTING AND PREDICTION
C           OF DISCRETE TIME SERIES', PRENTICE-HALL, N.J., 1963,
C           PP. 140 TO 144.
C
C     ..................................................................
C
      SUBROUTINE EXSMO (X,NX,AL,A,B,C,S)
      DIMENSION X(1),S(1)
C
C     IF A=B=C=0.0, GENERATE INITIAL VALUES OF A, B, AND C
C
      IF(A) 140, 110, 140
  110 IF(B) 140, 120, 140
  120 IF(C) 140, 130, 140
  130 C=X(1)-2.0*X(2)+X(3)
      B=X(2)-X(1)-1.5*C
      A=X(1)-B-0.5*C
C
  140 BE=1.0-AL
      BECUB=BE*BE*BE
      ALCUB=AL*AL*AL
C
C     DO THE FOLLOWING FOR I=1 TO NX
C
      DO 150 I=1,NX
C
C        FIND S(I) FOR ONE PERIOD AHEAD
C
      S(I)=A+B+0.5*C
C
C        UPDATE COEFFICIENTS A, B, AND C
C
      DIF=S(I)-X(I)
      A=X(I)+BECUB*DIF
      B=B+C-1.5*AL*AL*(2.0-AL)*DIF
  150 C=C-ALCUB*DIF
      RETURN
      END
```

Nonparametric Statistics

Subroutine KOLMO

Given a sample of n independent and identically distributed random variables $X_1, X_2, \ldots, X_n$ with continuous cumulative distribution function F(x), this subroutine tests the difference in absolute value between the empirical distribution $F_n(x)$ and theoretical distribution F(x), using Kolmogorov-Smirnov's limiting distribution.

For this purpose:

1. The order statistics $\{x_{(i)}\}$ are determined from the set $\{x_i\}$ by sorting $\{x_i\}$ into a nondecreasing sequence.
2. The empirical cumulative distribution function $F_n(x)$ is computed. This is the following step-function:

$$F_n(x) = \begin{cases} 0 & x < x_{(1)} \\ k/n & x_{(k)} \le x < x_{(k+1)};\ k=1,\ldots,n-1 \\ 1 & x_{(n)} \le x \end{cases}$$

3. The maximum deviation D_n in absolute value between the empirical and theoretical distribution is computed:

$$D_n = \max_{-\infty<x<\infty} \left|F_n(x) - F(x)\right|$$

Since $F_n(x)$ and F(x) are nondecreasing functions, the result is:

$$D_n = \max_{1\le k\le n} \left|F_n\left[x_{(k)}\right] - F\left[x_{(k)}\right]\right|$$

D_n is a random variable, and L(z) is the limiting cumulative distribution function of $n^{1/2} D_n$:

$$\lim_{n\to\infty} \text{Prob}\,\{n^{1/2} D_n < z\} = L(z)$$

4. Finally,

$$z = n^{1/2} D_n$$

and the probability of being greater than or equal to the computed value of $n^{1/2} D_n$

$$P = 1 - L(z)$$

are computed.

Generally, theoretical distribution functions are to be included by the user, as specified in the program. However, four functions are evaluated in KOLMO, as follows:

$$\int_{-\infty}^{x} dF(t) = F(x) \qquad (1)$$

is evaluated at the points of the set $\{X_{(i)}\}$, where F(x) is one of the following:

- The normal pdf with mean u and variance s^2
- The exponential pdf with mean u and variance s^2
- The Cauchy pdf with median u, and first quartile s - u
- The uniform pdf with endpoints u and s

Any user-written pdf should evaluate (1) above, using the parameters u and s at his convenience. Instructions given in the program KOLMO should be followed.

Lilliefors (1967) notes that critical values determined by this test are not correct when one or more parameters are estimated from the sample. The user should refer to his article for notes on approximations which may be considered if such estimates are used.

For references see:

(1) W. Feller, "On the Kolmogorov - Smirnov limit theorems for empirical distributions", Annals of Math. Stat., 19, pp. 177-189.

(2) N. Smirnov, "Table for estimating the goodness of fit of empirical distributions", Annals of Math. Stat., 19, pp. 279-281.

(3) R. Von Mises, Mathematical Theory of Probability and Statistics. Academic Press, New York, 1964, pp. 490-493.

(4) B.V. Gnedenko, The Theory of Probability. Chelsea Publishing Co., New York, 1962, pp. 384-401.

(5) H.W. Lilliefors, "On the Kolmogorov-Smirnov test for normality with mean and variance unknown", J.A.S.A., 62 (1967), pp. 399-402.

```
C
C     ..................................................................
C
C        SUBROUTINE KOLMO
C
C        PURPOSE
C           TESTS THE DIFFERENCE BETWEEN EMPIRICAL AND THEORETICAL
C           DISTRIBUTIONS  USING THE KOLMOGOROV-SMIRNOV TEST
C
C        USAGE
C           CALL KOLMO(X,N,Z,PROB,IFCOD,U,S,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X    - INPUT VECTOR OF N INDEPENDENT OBSERVATIONS.  ON
C                  RETURN FROM KOLMO, X HAS BEEN SORTED INTO A
C                  MONOTONIC NON-DECREASING SEQUENCE.
C           N    - NUMBER OF OBSERVATIONS IN X
C           Z    - OUTPUT VARIABLE CONTAINING THE GREATEST VALUE WITH
C                  RESPECT TO X OF  SQRT(N)*ABS(FN(X)-F(X)) WHERE
C                  F(X) IS A  THEORETICAL DISTRIBUTION FUNCTION AND
C                  FN(X) AN EMPIRICAL DISTRIBUTION FUNCTION.
C           PROB - OUTPUT VARIABLE CONTAINING THE PROBABILITY OF
C                  THE STATISTIC BEING GREATER THAN OR EQUAL TO Z IF
C                  THE HYPOTHESIS THAT X IS FROM THE DENSITY UNDER
C                  CONSIDERATION IS TRUE.  E.G., PROB = 0.05 IMPLIES
C                  THAT ONE CAN REJECT THE NULL HYPOTHESIS THAT THE SET
C                  X IS FROM THE DENSITY UNDER CONSIDERATION WITH 5 PER
C                  CENT PROBABILITY OF BEING INCORRECT.  PROB = 1. -
C                  SMIRN(Z).
C           IFCOD- A CODE DENOTING THE PARTICULAR THEORETICAL
C                  PROBABILITY DISTRIBUTION FUNCTION BEING CONSIDERED.
C                  = 1---F(X) IS THE NORMAL PDF.
C                  = 2---F(X) IS THE EXPONENTIAL PDF.
C                  = 3---F(X) IS THE CAUCHY PDF.
C                  = 4---F(X) IS THE UNIFORM PDF.
C                  = 5---F(X) IS USER SUPPLIED.
C           U    - WHEN IFCOD IS 1 OR 2, U IS THE MEAN OF THE DENSITY
C                  GIVEN ABOVE.
C                  WHEN IFCOD IS 3, U IS THE MEDIAN OF THE CAUCHY
C                  DENSITY.
C                  WHEN IFCOD IS 4, U IS THE LEFT ENDPOINT OF THE
C                  UNIFORM DENSITY.
C                  WHEN IFCOD IS 5, U IS USER SPECIFIED.
C           S    - WHEN IFCOD IS 1 OR 2, S IS THE STANDARD DEVIATION OF
C                  DENSITY GIVEN ABOVE, AND SHOULD BE POSITIVE.
C                  WHEN IFCOD IS 3, U - S SPECIFIES THE FIRST QUARTILE
C                  OF THE CAUCHY DENSITY.  S SHOULD BE NON-ZERO.
C                  IF IFCOD IS 4, S IS THE RIGHT ENDPOINT OF THE UNIFORM
C                  DENSITY.  S SHOULD BE GREATER THAN U.
C                  IF IFCOD IS 5, S IS USER SPECIFIED.
C           IER  - ERROR INDICATOR WHICH IS NON-ZERO IF S VIOLATES ABOVE
C                  CONVENTIONS.  ON RETURN NO TEST HAS BEEN MADE, AND X
C                  AND Y HAVE BEEN SORTED INTO MONOTONIC NON-DECREASING
C                  SEQUENCES.  IER IS SET TO ZERO ON ENTRY TO KOLMO.
C                  IER IS CURRENTLY SET TO ONE IF THE USER-SUPPLIED PDF
C                  IS REQUESTED FOR TESTING.  THIS SHOULD BE CHANGED
C                  (SEE REMARKS) WHEN SOME PDF IS SUPPLIED BY THE USER.
C
C        REMARKS
C           N SHOULD BE GREATER THAN OR EQUAL TO 100.  (SEE THE
C           MATHEMATICAL DESCRIPTION GIVEN FOR THE PROGRAM SMIRN,
C           CONCERNING ASYMPTOTIC FORMULAE)  ALSO, PROBABILITY LEVELS
C           DETERMINED BY THIS PROGRAM WILL NOT BE CORRECT IF THE
C           SAME SAMPLES ARE USED TO ESTIMATE PARAMETERS FOR THE
C           CONTINUOUS DISTRIBUTIONS WHICH ARE USED IN THIS TEST.
C           (SEE THE MATHEMATICAL DESCRIPTION FOR THIS PROGRAM)
C           F(X) SHOULD BE A CONTINUOUS FUNCTION.
C           ANY USER SUPPLIED CUMULATIVE PROBABILITY DISTRIBUTION
C           FUNCTION SHOULD BE CODED BEGINNING WITH STATEMENT 26 BELOW,
C           AND SHOULD RETURN TO STATEMENT 27.
C
C           DOUBLE PRECISION USAGE---IT IS DOUBTFUL THAT THE USER WILL
C           WISH TO PERFORM THIS TEST USING DOUBLE PRECISION ACCURACY.
C           IF ONE WISHES TO COMMUNICATE WITH KOLMO IN A DOUBLE
C           PRECISION PROGRAM, HE SHOULD CALL THE FORTRAN SUPPLIED
C           PROGRAM SNGL(X) PRIOR TO CALLING KOLMO, AND CALL THE
C           FORTRAN SUPPLIED PROGRAM DBLE(X) AFTER EXITING FROM KOLMO.
C           (NOTE THAT SUBROUTINE SMIRN DOES HAVE DOUBLE PRECISION
C           CAPABILITY AS SUPPLIED BY THIS PACKAGE.)
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           SMIRN, NOTR, AND ANY USER SUPPLIED SUBROUTINES REQUIRED.
C
C        METHOD
C           FOR REFERENCE, SEE (1) W. FELLER--ON THE KOLMOGOROV-SMIRNOV
C           LIMIT THEOREMS FOR EMPIRICAL DISTRIBUTIONS--
C           ANNALS OF MATH. STAT., 19, 1948.  177-189,
C           (2) N. SMIRNOV--TABLE FOR ESTIMATING THE GOODNESS OF FIT
C           OF EMPIRICAL DISTRIBUTIONS--ANNALS OF MATH. STAT., 19,
C           1948.  279-281.
C           (3) R. VON MISES--MATHEMATICAL THEORY OF PROBABILITY AND
C           STATISTICS--ACADEMIC PRESS, NEW YORK, 1964.  490-493,
C           (4) B.V. GNEDENKO--THE THEORY OF PROBABILITY--CHELSEA
C           PUBLISHING COMPANY, NEW YORK, 1962.  384-401.
C
C     ..................................................................
      SUBROUTINE KOLMO(X,N,Z,PROB,IFCOD,U,S,IER)
      DIMENSION X(1)
C
C        NON DECREASING ORDERING OF X(I)'S  (DUBY METHOD)
C
      IER=0
      DO 5 I=2,N
      IF(X(I)-X(I-1))1,5,5
    1 TEMP=X(I)
      IM=I-1
      DO 3 J=1,IM
      L=I-J
      IF(TEMP-X(L))2,4,4
    2 X(L+1)=X(L)
    3 CONTINUE
      X(1)=TEMP
      GO TO 5
    4 X(L+1)=TEMP
    5 CONTINUE
C
C        COMPUTES MAXIMUM DEVIATION DN IN ABSOLUTE VALUE BETWEEN
C        EMPIRICAL AND THEORETICAL DISTRIBUTIONS
C
      NM1=N-1
      XN=N
      DN=0.0
      FS=0.0
      IL=1
      DO 7  I=IL,NM1
      J=I
      IF(X(J)-X(J+1))9,7,9
    7 CONTINUE
    8 J=N
    9 IL=J+1
      FI=FS
      FS=FLOAT(J)/XN
      IF(IFCOD-2)10,13,17
   10 IF(S)11,11,12
   11 IER=1
      GO TO 29
   12 Z =(X(J)-U)/S
      CALL NOTR(Z,Y,0)
      GO TO 27
   13 IF(S)11,11,14
   14 Z=(X(J)-U)/S+1.0
      IF(Z)15,15,16
   15 Y=0.0
      GO TO 27
   16 Y=1.-EXP(-Z)
      GO TO 27
   17 IF(IFCOD-4)18,20,26
   18 IF(S)19,11,19
   19 Y=ATAN((X(J)-U)/S)*0.3183099+0.5
      GO TO 27
   20 IF(S-U)11,11,21
   21 IF(X(J)-U)22,22,23
   22 Y=0.0
      GO TO 27
   23 IF(X(J)-S)25,25,24
   24 Y=1.0
      GO TO 27
   25 Y=(X(J)-U)/(S-U)
      GO TO 27
   26 IER=1
      GO TO 29
   27 EI=ABS(Y-FI)
      ES=ABS(Y-FS)
      DN=AMAX1(DN,EI,ES)
      IF(IL-N)6,8,28
C
C        COMPUTES Z=DN*SQRT(N)  AND  PROBABILITY
C
   28 Z=DN*SQRT(XN)
      CALL SMIRN(Z,PROB)
      PROB=1.0-PROB
   29 RETURN
      END
```

Subroutine KOLM2

Given a sample of n i.i.d. (independent and identically distributed) random variables X, and a sample of m i.i.d. random variables Y, this subroutine tests the difference between the two empirical distribution functions $F_n(x)$ and $G_m(y)$ using Kolmogorov-Smirnov's limiting distribution. For this purpose:

1. The sets X and Y are sorted into the ordered sets $\{X_{(i)}\}$ and $\{Y_{(i)}\}$, which are nondecreasing sequences.

2. The empirical cumulative distribution functions $F_n(x)$ for the set X, and $G_m(y)$ for the set Y are computed. For example,

$$F_n(x) = \begin{cases} 0 & x < x_{(1)} \\ k/n & x_{(k)} \le x < x_{(k+1)}; \; k=1,\ldots,n-1 \\ 1 & x_{(n)} \le x \end{cases}$$

3. The maximum difference in absolute value between the two sample distribution functions is computed:

$$D_{m,n} = \max_{x,\,y} \left| F_n(x) - G_m(y) \right|$$

The statistic $\sqrt{\frac{mn}{m+n}}\, D_{m,n}$ is a random variable with limiting cumulative distribution function L(z), which is described under "Subroutine SMIRN" in this manual. That is,

$$\lim_{m,n \to \infty,\,\infty} \text{Prob}\left\{ \sqrt{\frac{mn}{m+n}}\, D_{m,n} < z \right\} = L(z)$$

4. Finally, the probability (asymptotic) of the statistic $\sqrt{\frac{mn}{m+n}}\, D_{m,n}$ being not less than its computed value, under the assumption of equality of the two theoretical distribution functions from which X and Y were taken, is computed:

$$P = 1 - L(z)$$

For reference see:

(1) W. Feller, "On the Kolmogorov-Smirnov limit theorems for empirical distributions", Annals of Math. Stat., 19, pp. 177-189.

(2) N. Smirnov, "Table for estimating the goodness of fit of empirical distributions", Annals of Math. Stat., 19, pp. 279-281.

(3) B. V. Gnedenko, The Theory of Probability. Chelsea Publishing Co. New York, 1962, pp. 384-401.


```
C                                                                       KLM2  10
C     ..................................................................KLM2  20
C                                                                       KLM2  30
C        SUBROUTINE KOLM2                                               KLM2  40
C                                                                       KLM2  50
C        PURPOSE                                                        KLM2  60
C                                                                       KLM2  70
C           TESTS THE DIFFERENCE BETWEEN TWO SAMPLE DISTRIBUTION        KLM2  80
C           FUNCTIONS USING THE KOLMOGOROV-SMIRNOV TEST                 KLM2  90
C                                                                       KLM2 100
C        USAGE                                                          KLM2 110
C           CALL KOLM2(X,Y,N,M,Z,PROB)                                  KLM2 120
C                                                                       KLM2 130
C        DESCRIPTION OF PARAMETERS                                      KLM2 140
C           X    - INPUT VECTOR OF N INDEPENDENT OBSERVATIONS.  ON      KLM2 150
C                  RETURN FROM KOLM2, X HAS BEEN SORTED INTO A          KLM2 160
C                  MONOTONIC NON-DECREASING SEQUENCE.                   KLM2 170
C           Y    - INPUT VECTOR OF M INDEPENDENT OBSERVATIONS.  ON      KLM2 180
C                  RETURN FROM KOLM2, Y HAS BEEN SORTED INTO A          KLM2 190
C                  MONOTONIC NON-DECREASING SEQUENCE.                   KLM2 200
C           N    - NUMBER OF OBSERVATIONS IN X                          KLM2 210
C           M    - NUMBER OF OBSERVATIONS IN Y                          KLM2 220
C           Z    - OUTPUT VARIABLE CONTAINING THE GREATEST VALUE WITH   KLM2 230
C                  RESPECT TO THE SPECTRUM OF X AND Y OF                KLM2 240
C                  SQRT((M*N)/(M+N))*ABS(FN(X)-GM(Y)) WHERE             KLM2 250
C                  FN(X) IS THE EMPIRICAL DISTRIBUTION FUNCTION OF THE  KLM2 260
C                  SET (X) AND GM(Y) IS THE EMPIRICAL DISTRIBUTION      KLM2 270
C                  FUNCTION OF THE SET (Y).                             KLM2 280
C           PROB - OUTPUT VARIABLE CONTAINING THE PROBABILITY OF        KLM2 290
C                  THE STATISTIC BEING GREATER THAN OR EQUAL TO Z IF    KLM2 300
C                  THE HYPOTHESIS THAT X AND Y ARE FROM THE SAME PDF IS KLM2 310
C                  TRUE.  E.G., PROB= 0.05 IMPLIES THAT ONE CAN REJECT  KLM2 320
C                  THE NULL HYPOTHESIS THAT THE SETS X AND Y ARE FROM   KLM2 330
C                  THE SAME DENSITY WITH 5 PER CENT PROBABILITY OF BEINGKLM2 340
C                  INCORRECT.  PROB = 1. - SMIRN(Z).                    KLM2 350
C                                                                       KLM2 360
C                                                                       KLM2 370
C        REMARKS                                                        KLM2 380
C           N AND M SHOULD BE GREATER THAN OR EQUAL TO 100.  (SEE THE   KLM2 390
C           MATHEMATICAL DESCRIPTION FOR THIS SUBROUTINE AND FOR THE    KLM2 400
C           SUBROUTINE SMIRN, CONCERNING ASYMPTOTIC FORMULAE).          KLM2 410
C                                                                       KLM2 420
C           DOUBLE PRECISION USAGE---IT IS DOUBTFUL THAT THE USER WILL  KLM2 430
C           WISH TO PERFORM THIS TEST USING DOUBLE PRECISION ACCURACY.  KLM2 440
C           IF ONE WISHES TO COMMUNICATE WITH KOLM2 IN A DOUBLE         KLM2 450
C           PRECISION PROGRAM, HE SHOULD CALL THE FORTRAN SUPPLIED      KLM2 460
C           PROGRAM SNGL(X) PRIOR TO CALLING KOLM2, AND CALL THE        KLM2 470
C           FORTRAN SUPPLIED PROGRAM DBLE(X) AFTER EXITING FROM KOLM2.  KLM2 480
C           (NOTE THAT SUBROUTINE SMIRN DOES HAVE DOUBLE PRECISION      KLM2 490
C           CAPABILITY AS SUPPLIED BY THIS PACKAGE.)                    KLM2 500
C                                                                       KLM2 510
C                                                                       KLM2 520
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  KLM2 530
C           SMIRN                                                       KLM2 540
C                                                                       KLM2 550
C        METHOD                                                         KLM2 560
C           FOR REFERENCE, SEE (1) W. FELLER--ON THE KOLMOGOROV-SMIRNOV KLM2 570
C           LIMIT THEOREMS FOR EMPIRICAL DISTRIBUTIONS--                KLM2 580
C           ANNALS OF MATH. STAT., 19, 1948.  177-189,                  KLM2 590
C           (2) N. SMIRNOV--TABLE FOR ESTIMATING THE GOODNESS OF FIT    KLM2 600
C           OF EMPIRICAL DISTRIBUTIONS--ANNALS OF MATH. STAT., 19,      KLM2 610
C           1948.  279-281.                                             KLM2 620
C           (3) R. VON MISES--MATHEMATICAL THEORY OF PROBABILITY AND    KLM2 630
C           STATISTICS--ACADEMIC PRESS, NEW YORK, 1964.  490-493,       KLM2 640
C           (4) B.V. GNEDENKO--THE THEORY OF PROBABILITY--CHELSEA       KLM2 650
C           PUBLISHING COMPANY, NEW YORK, 1962.  384-401.               KLM2 660
C     ..................................................................KLM2 670
C                                                                       KLM2 680
      SUBROUTINE KOLM2(X,Y,N,M,Z,PROB)                                  KLM2 690
      DIMENSION X(1),Y(1)                                               KLM2 700
C                                                                       KLM2 710
C        SORT X INTO ASCENDING SEQUENCE                                 KLM2 720
C                                                                       KLM2 730
      DO 5 I=2,N                                                        KLM2 740
      IF(X(I)-X(I-1))1,5,5                                              KLM2 750
    1 TEMP=X(I)                                                         KLM2 760
      IM=I-1                                                            KLM2 770
      DO 3 J=1,IM                                                       KLM2 780
      L=I-J                                                             KLM2 790
      IF(TEMP-X(L))2,4,4                                                KLM2 800
    2 X(L+1)=X(L)                                                       KLM2 810
    3 CONTINUE                                                          KLM2 820
      X(1)=TEMP                                                         KLM2 830
      GO TO 5                                                           KLM2 840
    4 X(L+1)=TEMP                                                       KLM2 850
    5 CONTINUE                                                          KLM2 860
C                                                                       KLM2 870
C        SORT Y INTO ASCENDING SEQUENCE                                 KLM2 880
C                                                                       KLM2 890
      DO 10 I=2,M                                                       KLM2 900
      IF(Y(I)-Y(I-1))6,10,10                                            KLM2 910
    6 TEMP=Y(I)                                                         KLM2 920
      IM=I-1                                                            KLM2 930
      DO 8  J=1,IM                                                      KLM2 940
      L=I-J                                                             KLM2 950
      IF(TEMP-Y(L))7,9,9                                                KLM2 960
    7 Y(L+1)=Y(L)                                                       KLM2 970
    8 CONTINUE                                                          KLM2 980
      Y(1)=TEMP                                                         KLM2 990
      GO TO 10                                                          KLM21000
    9 Y(L+1)=TEMP                                                       KLM21010
   10 CONTINUE                                                          KLM21020
C                                                                       KLM21030
C        CALCULATE D = ABS(FN-GM) OVER THE SPECTRUM OF X AND Y          KLM21040
C                                                                       KLM21050
      XN=FLOAT(N)                                                       KLM21060
      XN1=1./XN                                                         KLM21070
      XM=FLOAT(M)                                                       KLM21080
      XM1=1./XM                                                         KLM21090
      D=0.0                                                             KLM21100
      I=0                                                               KLM21110
```

```
      J=0                                                KLM21120
      K=0                                                KLM21130
      L=0                                                KLM21140
   11 IF(X(I+1)-Y(J+1))12,13,18                          KLM21150
   12 K=1                                                KLM21160
      GO TO 14                                           KLM21170
   13 K=0                                                KLM21180
   14 I=I+1                                              KLM21190
      IF(I-N)15,21,21                                    KLM21200
   15 IF(X(I+1)-X(I))14,14,16                            KLM21210
   16 IF(K)17,18,17                                      KLM21220
C                                                        KLM21230
C        CHOOSE THE MAXIMUM DIFFERENCE, D                KLM21240
C                                                        KLM21250
   17 D=AMAX1(D,ABS(FLOAT(I)*XN1-FLOAT(J)*XM1))          KLM21260
      IF(L)22,11,22                                      KLM21270
   18 J=J+1                                              KLM21280
      IF(J-M)19,20,20                                    KLM21290
   19 IF(Y(J+1)-Y(J))18,18,17                            KLM21300
   20 L=1                                                KLM21310
      GO TO 17                                           KLM21320
   21 L=1                                                KLM21330
      GO TO 16                                           KLM21340
C                                                        KLM21350
C        CALCULATE THE STATISTIC Z                       KLM21360
C                                                        KLM21370
   22 Z=D*SQRT((XN*XM)/(XN+XM))                          KLM21380
C                                                        KLM21390
C        CALCULATE THE PROBABILITY ASSOCIATED WITH Z     KLM21400
C                                                        KLM21410
      CALL SMIRN(Z,PROB)                                 KLM21420
      PROB=1.0-PROB                                      KLM21430
      RETURN                                             KLM21440
      END                                                KLM21450
```

Subroutine SMIRN

This subroutine computes the values of Kolmogorov-Smirnov's limiting distribution for a given argument x.

$$L(x) = \begin{cases} 0 & x \le 0 \\ 1 - 2 \sum_{k=1}^{\infty} (-1)^{k-1} \exp(-2k^2x^2) & x > 0 \end{cases} \qquad (1)$$

L(x) is the limit (Kolmogorov) of the cumulative distribution function of $\sqrt{n}\, D_n$, and of (Smirnov) $[mn/(m+n)]^{1/2} D_{m,n}$ where:

D_n is the maximum, over all x, of the difference $|F_n(x) - F(x)|$ between the sample distribution function $F_n(x)$ and the continuous theoretical distribution function F(x), and

$D_{m,n}$ is the maximum, over all x, of the difference between the two sample distribution functions $F_m(x)$ and $G_n(x)$, from two independent samples of sizes m and n.

When x is very small, the series (1) converges slowly, but, using Jacobi's Theta-functions $\theta_2(u,t)$ and $\theta_4(u,t)$:

$$\theta_2(u,t) = 2 \sum_{k=0}^{\infty} \exp[i\pi(k+1/2)^2 t] \cos[(2k+1)u]$$

$$\theta_4(u,t) = 1 - 2 \sum_{k=0}^{\infty} (-1)^{k-1} \exp(i\pi k^2 t) \cos(2ku)$$

and using the Jacobi imaginary transformation

$$\theta_4(0,t) = (-it)^{-1/2} \theta_2(0, -1/t)$$

it follows that:

$$L(x) = \theta_4(0, 2ix^2/\pi)$$

$$= (\sqrt{2\pi}/x) \sum_{k=1}^{\infty} \exp[-(2k-1)^2 \pi^2/8x^2]$$

which converges quickly when x is small. The computation here uses, with errors $E_i(x)$, $i = 1, 2$:

$$
L(x) = \begin{cases}
0 & x \le 0.27 \\[2ex]
(\sqrt{2\pi}/x) \displaystyle\sum_{k=1}^{3} \exp\left[-(2k-1)^2 \pi^2 / 8x^2\right] + E_1(x); & 0.27 < x < 1.0 \\[2ex]
1 - 2 \displaystyle\sum_{k=1}^{4} (-1)^{k-1} \exp\left(-2k^2 x^2\right) + E_2(x) & 1.0 \le x < 3.1 \\[2ex]
1 & 3.1 \le x < \infty
\end{cases}
$$

where:

$$E_1(x) \le 6\,(10^{-15}) \quad \text{when } x < 1$$

$$E_2(x) < 10^{-20} \quad \text{when } x \ge 1$$

For reference see:

(1) E. T. Whittaker and G. N. Watson, A Course of Modern Analysis. Cambridge University Press, Cambridge, England, 1952, pp. 462-476.

(2) W. Feller, "On the Kolmogorov-Smirnov limit theorems for empirical distributions", Annals of Math. Stat. 19, pp. 177-189.

(3) N. Smirnov, "Table for estimating the goodness of fit of empirical distributions", Annals of Math. Stat. 19, pp. 279-281.

```
C                                                                       SMIR  10
C ......................................................................SMIR  20
C                                                                       SMIR  30
C     SUBROUTINE SMIRN                                                  SMIR  40
C                                                                       SMIR  50
C     PURPOSE                                                           SMIR  60
C        COMPUTES VALUES OF THE LIMITING DISTRIBUTION FUNCTION FOR      SMIR  70
C        THE KOLMOGOROV-SMIRNOV STATISTIC.                              SMIR  80
C                                                                       SMIR  90
C     USAGE                                                             SMIR 100
C        CALL SMIRN(X,Y)                                                SMIR 110
C                                                                       SMIR 120
C     DESCRIPTION OF PARAMETERS                                         SMIR 130
C        X    - THE ARGUMENT OF THE SMIRN FUNCTION                      SMIR 140
C        Y    - THE RESULTANT SMIRN FUNCTION VALUE                      SMIR 150
C                                                                       SMIR 160
C     REMARKS                                                           SMIR 170
C        Y IS SET TO ZERO IF X IS NOT GREATER THAN 0.27, AND IS SET     SMIR 180
C        TO ONE IF X IS NOT LESS THAN 3.1.  ACCURACY TESTS WERE MADE    SMIR 190
C        REFERRING TO THE TABLE GIVEN IN THE REFERENCE BELOW.           SMIR 200
C        THE ARGUMENTS, X= 0.62, AND X = 1.87 GAVE RESULTS WHICH        SMIR 210
C        DIFFER FROM THE SMIRNOV TABLES BY 2.9 AND 1.9 IN THE 5TH       SMIR 220
C        DECIMAL PLACE.  ALL OTHER RESULTS SHOWED SMALLER ERRORS,       SMIR 230
C        AND ERROR SPECIFICATIONS ARE GIVEN IN THE ACCURACY TABLES      SMIR 240
C        IN THIS MANUAL.  IN DOUBLE PRECISION MODE, THESE SAME          SMIR 250
C        ARGUMENTS RESULTED IN DIFFERENCES FROM TABLED VALUES BY 3      SMIR 260
C        AND 2 IN THE 5TH DECIMAL PLACE.  IT IS NOTED IN                SMIR 270
C        LINDGREN (REFERENCE BELOW) THAT FOR HIGH SIGNIFICANCE LEVELSSMIR 280
C        (SAY, .01 AND .05) ASYMPTOTIC FORMULAS GIVE VALUES WHICH ARESMIR 290
C        TOO HIGH ( BY 1.5 PER CENT WHEN N = 80).  THAT IS, AT HIGH     SMIR 300
C        SIGNIFICANCE LEVELS, THE HYPOTHESIS OF NO DIFFERENCE WILL BESMIR 310
C        REJECTED TOO SELDOM USING ASYMPTOTIC FORMULAS.                 SMIR 320
C                                                                       SMIR 330
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     SMIR 340
C        NONE                                                           SMIR 350
C                                                                       SMIR 360
C                                                                       SMIR 370
C     METHOD                                                            SMIR 380
C        THE METHOD IS DESCRIBED BY W. FELLER-ON THE KOLMOGOROV-        SMIR 390
C        SMIRNOV LIMIT THEOREMS FOR EMPIRICAL DISTRIBUTIONS- ANNALS     SMIR 400
C        OF MATH. STAT., 19, 1948, 177-189, BY N. SMIRNOV--TABLE        SMIR 410
C        FOR ESTIMATING THE GOODNESS OF FIT OF EMPIRICAL                SMIR 420
C        DISTRIBUTIONS- ANNALS OF MATH. STAT., 19, 1948, 279-281,       SMIR 430
C        AND GIVEN IN LINDGREN, STATISTICAL THEORY, THE MACMILLAN       SMIR 440
C        COMPANY, N. Y., 1962.                                          SMIR 450
C                                                                       SMIR 460
C ......................................................................SMIR 470
C                                                                       SMIR 480
      SUBROUTINE SMIRN(X,Y)                                             SMIR 490
C     DOUBLE PRECISION X,Q1,Q2,Q4,Q8,Y                                  SMIR 500
C                                                                       SMIR 510
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE CSMIR 520
C        IN COLUMN ONE OF THE DOUBLE PRECISION CARD ABOVE SHOULD BE     SMIR 530
C        REMOVED, AND THE C IN COLUMN ONE OF THE STATEMENTS NUMBERED    SMIR 540
C        C   3, C   5, AND C   8 SHOULD BE REMOVED, AND THESE CARDS     SMIR 550
C        SHOULD REPLACE THE STATEMENTS NUMBERED 3, 5, AND 8,            SMIR 560
C        RESPECTIVELY.  ALL ROUTINES CALLING THIS ROUTINE MUST ALSO     SMIR 570
C        PROVIDE DOUBLE PRECISION ARGUMENTS TO THIS ROUTINE.            SMIR 580
C                                                                       SMIR 590
C ......................................................................SMIR 600
      IF(X-.27)1,1,2                                                    SMIR 610
    1 Y=0.0                                                             SMIR 620
      GO TO 9                                                           SMIR 630
    2 IF(X-1.0)3,6,6                                                    SMIR 640
    3 Q1=EXP(-1.2337011/X**2)                                           SMIR 650
C   3 Q1=DEXP(-1.2337005513617D/X**2)                                   SMIR 660
      Q2=Q1*Q1                                                          SMIR 670
      Q4=Q2*Q2                                                          SMIR 680
      Q8=Q4*Q4                                                          SMIR 690
      IF(Q8-1.0E-25)4,5,5                                               SMIR 700
    4 Q8=0.0                                                            SMIR 710
    5 Y=(2.5066282/X)*Q1*(1.0+Q8*(1.0+Q8*Q8))                           SMIR 720
C   5 Y=(2.5066282746310D1/X)*Q1*(1.0D0+Q8*(1.0D0+Q8*Q8))               SMIR 730
      GO TO 9                                                           SMIR 740
    6 IF(X-3.1)6,7,7                                                    SMIR 750
    7 Y=1.0                                                             SMIR 760
      GO TO 9                                                           SMIR 770
    6 Q1=EXP(-2.0*X*X)                                                  SMIR 780
C   6 Q1=DEXP(-2.0D0*X*X)                                               SMIR 790
      Q2=Q1*Q1                                                          SMIR 800
      Q4=Q2*Q2                                                          SMIR 810
      Q8=Q4*Q4                                                          SMIR 820
      Y=1.0-2.0*(Q1-Q4+Q8*(Q1-Q8))                                      SMIR 830
    9 RETURN                                                            SMIR 840
      END                                                               SMIR 850
```

Subroutine CHISQ

This subroutine calculates degrees of freedom and chi-square for a given contingency table A of observed frequencies with n rows (conditions) and m columns (groups). The degrees of freedom are:

$$\text{d.f.} = (n - 1)\ (m - 1) \tag{1}$$

If one or more cells contain an expected value less than 1.0, chi-square will be computed, but the error code will be set to one.

The following totals are computed:

$$T_i = \sum_{j=1}^{m} A_{ij};\ i = 1, 2, \ldots, n \text{ (row totals)} \tag{2}$$

$$T_j = \sum_{i=1}^{n} A_{ij};\ j = 1, 2, \ldots, m \text{ (column totals)} \tag{3}$$

$$GT = \sum_{i=1}^{n} T_i \quad \text{(grand total)} \tag{4}$$

Chi-square is obtained for two cases.

(a) For 2 x 2 table:

$$\chi^2 = \frac{GT\left(\left|A_{11}A_{22} - A_{12}A_{21}\right| - \frac{GT}{2}\right)^2}{(A_{11}+A_{12})(A_{21}+A_{22})(A_{11}+A_{21})(A_{12}+A_{22})} \tag{5}$$

(b) For other contingency tables:

$$\chi^2 = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{\left(A_{ij} - E_{ij}\right)^2}{E_{ij}} \tag{6}$$

where $E_{ij} = \dfrac{T_i T_j}{GT}$

$i = 1, 2, \ldots, n$

$j = 1, 2, \ldots, m$

```
C                                                                       CHIS  10
C     ...................................................................CHIS  20
C                                                                       CHIS  30
C       SUBROUTINE CHISQ                                                CHIS  40
C                                                                       CHIS  50
C       PURPOSE                                                         CHIS  60
C          COMPUTE CHI-SQUARE FROM A CONTINGENCY TABLE                  CHIS  70
C                                                                       CHIS  80
C       USAGE                                                           CHIS  90
C          CALL CHISQ(A,N,M,CS,NDF,IERR,TR,TC)                          CHIS 100
C                                                                       CHIS 110
C       DESCRIPTION OF PARAMETERS                                       CHIS 120
C          A    - INPUT MATRIX, N BY M, CONTAINING CONTINGENCY TABLE    CHIS 130
C          N    - NUMBER OF ROWS IN A                                   CHIS 140
C          M    - NUMBER OF COLUMNS IN A                                CHIS 150
C          CS   - CHI-SQUARE (OUTPUT)                                   CHIS 160
C          NDF  - NUMBER OF DEGREES OF FREEDOM (OUTPUT)                 CHIS 170
C          IERR - ERROR CODE (OUTPUT)                                   CHIS 180
C                   0 - NORMAL CASE                                     CHIS 190
C                   1 - EXPECTED VALUE IS LESS THAN 1.0 IN ONE OR       CHIS 200
C                       MORE CELLS                                      CHIS 210
C                   3 - NUMBER OF DEGREES OF FREEDOM IS ZERO            CHIS 220
C          TR   - WORK VECTOR OF LENGTH N                               CHIS 230
C          TC   - WORK VECTOR OF LENGTH M                               CHIS 240
C                                                                       CHIS 250
C       REMARKS                                                         CHIS 260
C          IF ONE OR MORE CELLS CONTAIN AN EXPECTED VALUE (I.E.,        CHIS 270
C          THEORETICAL VALUE) LESS THAN 1.0, CHI-SQUARE WILL BE         CHIS 280
C          COMPUTED, BUT ERROR CODE WILL BE SET TO 1.                   CHIS 290
C          SEE REFERENCE GIVEN BELOW.                                   CHIS 300
C          CHI-SQUARE IS SET TO ZERO IF EITHER N OR M IS ONE (ERROR     CHIS 310
C          CODE 3).                                                     CHIS 320
C                                                                       CHIS 330
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   CHIS 340
C          NONE                                                         CHIS 350
C                                                                       CHIS 360
C       METHOD                                                          CHIS 370
C          DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE    CHIS 380
C          BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,           CHIS 390
C          CHAPTER 6 AND CHAPTER 8.                                     CHIS 400
C                                                                       CHIS 410
C     ...................................................................CHIS 420
C                                                                       CHIS 430
      SUBROUTINE CHISQ(A,N,M,CS,NDF,IERR,TR,TC)                         CHIS 440
      DIMENSION A(1),TR(1),TC(1)                                        CHIS 450
C                                                                       CHIS 460
      NM=N*M                                                            CHIS 470
      IERR=0                                                            CHIS 480
      CS=0.0                                                            CHIS 490
C                                                                       CHIS 500
C        FIND DEGREES OF FREEDOM                                        CHIS 510
C                                                                       CHIS 520
      NDF=(N-1)*(M-1)                                                   CHIS 530
      IF(NDF) 5,5,10                                                    CHIS 540
    5 IERR=3                                                            CHIS 550
      RETURN                                                            CHIS 560
C                                                                       CHIS 570
C        COMPUTE TOTALS OF ROWS                                         CHIS 580
C                                                                       CHIS 590
   10 DO 90 I=1,N                                                       CHIS 600
      TR(I)=0.0                                                         CHIS 610
      IJ=I-N                                                            CHIS 620
      DO 90 J=1,M                                                       CHIS 630
      IJ=IJ+N                                                           CHIS 640
   90 TR(I)=TR(I)+A(IJ)                                                 CHIS 650
C                                                                       CHIS 660
C        COMPUTE TOTALS OF COLUMNS                                      CHIS 670
C                                                                       CHIS 680
      IJ=0                                                              CHIS 690
      DO 100 J=1,M                                                      CHIS 700
      TC(J)=0.0                                                         CHIS 710
      DO 100 I=1,N                                                      CHIS 720
      IJ=IJ+1                                                           CHIS 730
  100 TC(J)=TC(J)+A(IJ)                                                 CHIS 740
C                                                                       CHIS 750
C        COMPUTE GRAND TOTAL                                            CHIS 760
C                                                                       CHIS 770
      GT=0.0                                                            CHIS 780
      DO 110 I=1,N                                                      CHIS 790
  110 GT=GT+TR(I)                                                       CHIS 800
C                                                                       CHIS 810
C        COMPUTE CHI SQUARE FOR 2 BY 2 TABLE (SPECIAL CASE)             CHIS 820
C                                                                       CHIS 830
      IF(NM-4) 130,120,130                                              CHIS 840
  120 CS=GT*(ABS(A(1)*A(4)-A(2)*A(3))-GT/2.0)**2  /(TC(1)*TC(2)*TR(1)   CHIS 850
     1*TR(2))                                                           CHIS 860
      RETURN                                                            CHIS 870
C                                                                       CHIS 880
C        COMPUTE CHI SQUARE FOR OTHER CONTINGENCY TABLES                CHIS 890
C                                                                       CHIS 900
  130 IJ=0                                                              CHIS 910
      DO 140 J=1,M                                                      CHIS 920
      DO 140 I=1,N                                                      CHIS 930
      IJ=IJ+1                                                           CHIS 940
      E=TR(I)*TC(J)/GT                                                  CHIS 950
      IF(E-1.0) 135, 140, 140                                           CHIS 960
  135 IERR=1                                                            CHIS 970
  140 CS=CS+(A(IJ)-E)*(A(IJ)-E)/E                                       CHIS 980
      RETURN                                                            CHIS 990
      END                                                               CHIS1000
```

Subroutine KRANK

The subroutine computes the Kendall rank correlation coefficient, given two vectors of n observations for two variables, A and B. The observations on each variable are ranked from 1 to n. Tied observations are assigned the average of the tied ranks. Ranks are sorted in sequence of variable A.

A correction factor for ties is obtained:

$$T_a = \sum \frac{t(t-1)}{2} \quad \text{for variable A}$$

$$T_b = \sum \frac{t(t-1)}{2} \quad \text{for variable B} \qquad (1)$$

where t = number of observations tied for a given rank

The Kendall rank correlation coefficient is then computed for the following two cases:

(a) if T_a and T_b are zero,

$$\tau = \frac{S}{\frac{1}{2} n (n-1)} \qquad (2)$$

where n = number of ranks

S = total score calculated for ranks in variable B by selecting each rank in turn, adding 1 for each larger rank to its right, subtracting 1 for each smaller rank to its right

(b) if T_a and/or T_b are not zero,

$$\tau = \frac{S}{\sqrt{\frac{1}{2} n (n-1) - T_a}\ \sqrt{\frac{1}{2} n (n-1) - T_b}} \qquad (3)$$

The standard deviation is calculated:

$$s = \sqrt{\frac{2(2n+5)}{9n(n-1)}} \qquad (4)$$

The significance of τ can be measured by

$$z = \frac{\tau}{s} \qquad (5)$$

```
C
C     ..................................................................
C
C        SUBROUTINE KRANK
C
C        PURPOSE
C           TEST CORRELATION BETWEEN TWO VARIABLES BY MEANS OF KENDALL
C           RANK CORRELATION COEFFICIENT
C
C        USAGE
C           CALL KRANK(A,B,R,N,TAU,SD,Z,NR)
C
C        DESCRIPTION OF PARAMETERS
C           A   - INPUT VECTOR OF N OBSERVATIONS FOR FIRST VARIABLE
C           B   - INPUT VECTOR OF N OBSERVATIONS FOR SECOND VARIABLE
C           R   - OUTPUT VECTOR OF RANKED DATA OF LENGTH 2*N. SMALLEST
C                 OBSERVATION IS RANKED 1, LARGEST IS RANKED N. TIES
C                 ARE ASSIGNED AVERAGE OF TIED RANKS.
C           N   - NUMBER OF OBSERVATIONS
C           TAU - KENDALL RANK CORRELATION COEFFICIENT (OUTPUT)
C           SD  - STANDARD DEVIATION (OUTPUT)
C           Z   - TEST OF SIGNIFICANCE OF TAU IN TERMS OF NORMAL
C                 DISTRIBUTION (OUTPUT)
C           NR  - CODE, 0 FOR UNRANKED DATA IN A AND B, 1 FOR RANKED
C                 DATA IN A AND B (INPUT)
C
C        REMARKS
C           SD AND Z ARE SET TO ZERO IF N IS LESS THAN TEN
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           RANK
C           TIE
C
C        METHOD
C           DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE
C           BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,
C           CHAPTER 9
C     ..................................................................
      SUBROUTINE KRANK(A,B,R,N,TAU,SD,Z,NR)
      DIMENSION A(1),B(1),R(1)
C
      SD=0.0
      Z=0.0
      FN=N
      FN1=N*(N-1)
C
C        DETERMINE WHETHER DATA IS RANKED
C
      IF(NR-1) 5, 10, 5
C
C        RANK DATA IN A AND B VECTORS AND ASSIGN TIED OBSERVATIONS
C        AVERAGE OF TIED RANKS
C
    5 CALL RANK (A,R,N)
      CALL RANK (B,R(N+1),N)
      GO TO 40
C
C        MOVE RANKED DATA TO R VECTOR
C
   10 DO 20 I=1,N
   20 R(I)=A(I)
      DO 30 I=1,N
      J=I+N
   30 R(J)=B(I)
C
C        SORT RANK VECTOR R IN SEQUENCE OF VARIABLE A
C
   40 ISORT=0
      DO 50 I=2,N
      IF(R(I)-R(I-1)) 45,50,50
   45 ISORT=ISORT+1
      RSAVE=R(I)
      R(I)=R(I-1)
      R(I-1)=RSAVE
      I2=I+N
      SAVER=R(I2)
      R(I2)=R(I2-1)
      R(I2-1)=SAVER
   50 CONTINUE
      IF(ISORT) 40,55,40
C
C        COMPUTE S ON VARIABLE B. STARTING WITH THE FIRST RANK, ADD 1
C        TO S FOR EACH LARGER RANK TO ITS RIGHT AND SUBTRACT 1 FOR EACH
C        SMALLER RANK.  REPEAT FOR ALL RANKS.
C
   55 S=0.0
      NM=N-1
      DO 60 I=1,NM
      J=N+I
      DO 60 L=I,N
      K=N+L
      IF(R(K)-R(J)) 56,60,57
   56 S=S-1.0
      GO TO 60
   57 S=S+1.0
   60 CONTINUE
C
C        COMPUTE TIED SCORE INDEX FOR BOTH VARIABLES
C
      KT=2
      CALL TIE(R,N,KT,TA)
      CALL TIE(R(N+1),N,KT,TB)
C
C        COMPUTE TAU
C
      IF(TA) 70,65,70
   65 IF(TB) 70,67,70
   67 TAU=S/(0.5*FN1)
      GO TO 80
   70 TAU=S/((SQRT(0.5*FN1-TA))*(SQRT(0.5*FN1-TB)))
C
C        COMPUTE STANDARD DEVIATION AND Z IF N IS 10 OR LARGER
C
   80 IF(N-10) 90,85,85
   85 SD=(SQRT((2.0*(FN+FN+5.0))/(9.0*FN1)))
      Z=TAU/SD
   90 RETURN
      END
```

Subroutine MPAIR

This subroutine performs the Wilcoxon matched-pairs signed-ranks test, given two vectors of n observations of matched samples. The computational steps are as follows:

1. For each matched pair, determine the signed difference (d_i) between the two scores.
2. Rank these d's without respect to sign. In doing so, (a) disregard those d's equal to zero (0), and (b) with tied d's, assign the average of the tied ranks.
3. Affix to each rank the sign (+ or -) of the d in step 1 above.
4. Compute T = the sum of positive ranks or the sum of negative ranks, whichever is smaller.
5. Compute mean:

$$\mu = \frac{K(K+1)}{4} \tag{1}$$

where K = the total number of d's which are not zero (0).

6. Compute standard deviation:

$$\sigma = \sqrt{\frac{K(K+1)\ (2K+1)}{24}} \tag{2}$$

7. Compute Z:

$$Z = \frac{T - \mu}{\sigma} \tag{3}$$

Z is approximately normally distributed with zero mean and unit variance.

8. Compute the probability, p, associated with the value as extreme as Z by the subroutine NDTR.

Note: p is the probability for a one-tailed test; thus for a two-tailed test, the user should double the value of p.

For reference see:

(1) S. Siegel, Nonparametric Statistics for the Behavioral Sciences, McGraw-Hill, New York, 1956, chapter 5.

(2) F. Wilcoxon, Some Rapid Approximate Statistical Procedures, American Cyanamid Co., Stamford, Conn., 1949.

```
C
C     ..................................................................
C
C        SUBROUTINE MPAIR
C
C        PURPOSE
C           PERFORM THE WILCOXON MATCHED-PAIRS SIGNED-RANKS TEST, GIVEN
C           TWO VECTORS OF N OBSERVATIONS OF THE MATCHED SAMPLES.
C
C        USAGE
C           CALL MPAIR (N,A,B,K,T,Z,P,D,E,L,IE)
C
C        DESCRIPTION OF PARAMETERS
C           N - NUMBER OF OBSERVATIONS IN THE VECTORS A AND B
C           A - INPUT VECTOR OF LENGTH N CONTAINING DATA FROM THE FIRST
C               SAMPLE
C           B - INPUT VECTOR OF LENGTH N CONTAINING DATA FROM THE SECOND
C               SAMPLE
C           K - OUTPUT VARIABLE CONTAINING THE NUMBER OF PAIRS OF THE
C               MATCHED SAMPLES WHOSE DIFFERENCES ARE NON ZERO (0)
C           T - OUTPUT VARIABLE CONTAINING THE SUM OF THE RANKS OF PLUS
C               OR MINUS DIFFERENCES, WHICHEVER IS SMALLER
C           Z - VALUE OF THE STANDARDIZED NORMAL SCORE COMPUTED FOR THE
C               WILCOXON MATCHED-PAIRS SIGNED-RANKS TEST
C           P - COMPUTED PROBABILITY OF OBTAINING A VALUE OF Z AS
C               EXTREME AS THE ONE FOUND BY THE TEST
C           D - WORKING VECTOR OF LENGTH N
C           E - WORKING VECTOR OF LENGTH N
C           L - WORKING VECTOR OF LENGTH N
C           IE- 1, IF SAMPLES A AND B ARE IDENTICAL.
C               0 OTHERWISE.  IF IE=1, THEN T=P=0, AND Z=-10**75
C
C        REMARKS
C           THE COMPUTED PROBABILTY IS FOR A ONE-TAILED TEST.
C           MULTIPLYING P BY 2 WILL GIVE THE VALUE FOR A TWO-TAILED
C           TEST.
C
C        SUBROUTINES AND FUNCTIONS SUBPROGRAMS REQUIRED
C           RANK
C           NDTR
C
C        METHOD
C           REFER TO DIXON AND MASSEY, AN INTRODUCTION TO STATISTICAL
C           ANALYSIS (MC GRAW-HILL, 1957)
C
C     ..................................................................
C
      SUBROUTINE MPAIR (N,A,B,K,T,Z,P,D,E,L,IE)
C
      DIMENSION A(1),B(1),D(1),E(1),L(1)
C
      IE=0
      K=N
C
C        FIND DIFFERENCES OF MATCHED-PAIRS
C
      BIG=0.0
      DO 55 I=1,N
      DIF=A(I)-B(I)
      IF(DIF) 10, 20, 30
C
C        DIFFERENCE HAS A NEGATIVE SIGN (-)
C
   10 L(I)=1
      GO TO 40
C
C        DIFFERENCE IS ZERO (0)
C
   20 L(I)=2
      K=K-1
      GO TO 40
C
C        DIFFERENCE HAS A POSITIVE SIGN (+)
C
   30 L(I)=3
C
   40 DIF= ABS(DIF)
      IF(BIG-DIF) 45, 50, 50
   45 BIG=DIF
   50 D(I)=DIF
C
   55 CONTINUE
      IF(K) 57,57,59
   57 IE=1
      T=0.0
      Z=-1.0E75
      P=0
      GO TO 100
C
C        STORE A LARGE VALUE IN PLACE OF 0 DIFFERENCE IN ORDER TO
C        ASSIGN A LARGE RANK (LARGER THAN K), SO THAT ABSOLUTE VALUES
C        OF SIGNED DIFFERENCES WILL BE PROPERLY RANKED
C
   59 BIG=BIG*2.0
      DO 65 I=1,N
      IF(L(I)-2) 65, 60, 65
   60 D(I)=BIG
   65 CONTINUE
C
      CALL RANK (D,E,N)
C
C        FIND SUMS OF RANKS OF (+) DIFFERENCES AND (-) DIFFERENCES
C
      SUMP=0.0
      SUMM=0.0
      DO 80 I=1,N
      IF(L(I)-2) 70, 80, 75
   70 SUMM=SUMM+E(I)
      GO TO 80
   75 SUMP=SUMP+E(I)
   80 CONTINUE
C
C        SET T = SMALLER SUM
C
      IF(SUMP-SUMM) 85, 85, 90
   85 T=SUMP
      GO TO 95
   90 T=SUMM
C
C        COMPUTE MEAN, STANDARD DEVIATION, AND Z
C
   95 FK=K
      U=FK*(FK+1.0)/4.0
      S= SQRT((FK*(FK+1.0)*(2.0*FK+1.0))/24.0)
      Z=(T-U)/S
C
C        COMPUTE THE PROBABILITY OF A VALUE AS EXTREME AS Z
C
      CALL NDTR (Z,P,BIG)
C
  100 RETURN
      END
```

Subroutine QTEST

This subroutine determines the Cochran Q-test statistic, given a matrix A of dichotomous data with n rows (sets) and m columns (groups).

Row and column totals are calculated:

$$L_i = \sum_{j=1}^{m} A_{ij} \quad \text{(row totals)} \tag{1}$$

where $i = 1, 2, \ldots, n$

$$G_j = \sum_{i=1}^{n} A_{ij} \quad \text{(column totals)} \tag{2}$$

where $j = 1, 2, \ldots, m$

The Cochran Q statistic is computed:

$$Q = \frac{(m-1)\left[m\sum_{j=1}^{m} G_j^2 - \left(\sum_{j=1}^{m} G_j\right)^2\right]}{m\sum_{i=1}^{n} L_i - \sum_{i=1}^{n} L_i^2} \tag{3}$$

The degrees of freedom are:

$$\text{d.f.} = m - 1 \tag{4}$$

```
C
C     ..................................................................
C
C        SUBROUTINE QTEST
C
C        PURPOSE
C           TEST WHETHER THREE OR MORE MATCHED GROUPS OF DICHOTOMOUS
C           DATA DIFFER SIGNIFICANTLY BY THE COCHRAN Q-TEST
C
C        USAGE
C           CALL QTEST(A,N,M,Q,NDF)
C
C        DESCRIPTION OF PARAMETERS
C           A   - INPUT MATRIX, N BY M, OF DICHOTOMOUS DATA (0 AND 1)
C           N   - NUMBER OF SETS IN EACH GROUP
C           M   - NUMBER OF GROUPS
C           Q   - COCHRAN Q STATISTIC (OUTPUT)
C           NDF - NUMBER OF DEGREES OF FREEDOM (OUTPUT)
C
C        REMARKS
C           M MUST BE THREE OR GREATER
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE
C           BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,
C           CHAPTER 7
C
C     ..................................................................
      SUBROUTINE QTEST(A,N,M,Q,NDF)
      DIMENSION A(1)
C
C        COMPUTE SUM OF SQUARES OF ROW TOTALS, RSQ, AND GRAND TOTAL OF
C        ALL ELEMENTS, GD
C
      RSQ=0.0
      GD=0.0
      DO 20 I=1,N
      TR=0.0
      IJ=I-N
      DO 10 J=1,M
      IJ=IJ+N
   10 TR=TR+A(IJ)
      GD=GD+TR
   20 RSQ=RSQ+TR*TR
C
C        COMPUTE SUM OF SQUARES OF COLUMN TOTALS, CSQ
C
      CSQ=0.0
      IJ=0
      DO 40 J=1,M
      TC=0.0
      DO 30 I=1,N
      IJ=IJ+1
   30 TC=TC+A(IJ)
   40 CSQ=CSQ+TC*TC
C
C        COMPUTE COCHRAN Q TEST VALUE
C
      FM=M
      Q=(FM-1.0)*(FM*CSQ-GD*GD)/(FM*GD-RSQ)
C
C        FIND DEGREES OF FREEDOM
C
      NDF=M-1
      RETURN
      END
```

Subroutine RANK

```
C
C     ..................................................................
C
C        SUBROUTINE RANK
C
C        PURPOSE
C           RANK A VECTOR OF VALUES
C
C        USAGE
C           CALL RANK(A,R,N)
C
C        DESCRIPTION OF PARAMETERS
C           A - INPUT VECTOR OF N VALUES
C           R - OUTPUT VECTOR OF LENGTH N. SMALLEST VALUE IS RANKED 1,
C                 LARGEST IS RANKED N. TIES ARE ASSIGNED AVERAGE OF TIED
C                 RANKS
C           N - NUMBER OF VALUES
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           VECTOR IS SEARCHED FOR SUCCESSIVELY LARGER ELEMENTS. IF TIES
C           OCCUR, THEY ARE LOCATED AND THEIR RANK VALUE COMPUTED.
C           FOR EXAMPLE, IF 2 VALUES ARE TIED FOR SIXTH RANK, THEY ARE
C           ASSIGNED A RANK OF 6.5 (=(6+7)/2)
C
C     ..................................................................
      SUBROUTINE RANK(A,R,N)
      DIMENSION A(1),R(1)
C
C        INITIALIZATION
C
      DO 10 I=1,N
   10 R(I)=0.0
C
C        FIND RANK OF DATA
C
      DO 100 I=1,N
C
C        TEST WHETHER DATA POINT IS ALREADY RANKED
C
      IF(R(I)) 20, 20, 100
C
C        DATA POINT TO BE RANKED
C
   20 SMALL=0.0
      EQUAL=0.0
      X=A(I)
      DO 50 J=1,N
      IF(A(J)-X) 30, 40, 50
C
C        COUNT NUMBER OF DATA POINTS WHICH ARE SMALLER
C
   30 SMALL=SMALL+1.0
      GO TO 50
C
C        COUNT NUMBER OF DATA POINTS WHICH ARE EQUAL
C
   40 EQUAL=EQUAL+1.0
      R(J)=-1.0
   50 CONTINUE
C
C        TEST FOR TIE
C
      IF(EQUAL-1.0) 60, 60, 70
C
C        STORE RANK OF DATA POINT WHERE NO TIE
C
   60 R(I)=SMALL+1.0
      GO TO 100
C
C        CALCULATE RANK OF TIED DATA POINTS
C
   70 P=SMALL + (EQUAL + 1.0)*0.5
      DO 90 J=1,N
      IF(R(J)+1.0) 80, 80, 90
   80 R(J)=P
   90 CONTINUE
  100 CONTINUE
      RETURN
      END
```

Subroutine SIGNT

This subroutine performs a sign test, given two vectors of n observations of two matched samples. The computational steps are as follows:

1. Find the sign of the difference between the two numbers of each pair.
2. Find K = the number of pairs whose differences are not zero (0). Disregard those pairs whose differences are zero (0).
3. Find M = the number of (+) differences or the number of (-) differences, whichever is fewer.
4. If K is less than or equal to 25, the probability associated with a value as small as the observed value of M is given by the binomial distribution as:

$$p = \sum_{i=0}^{M} \binom{K}{i} P^{i} Q^{K-i} \qquad (1)$$

where $P = Q = 0.5$

$$\binom{K}{i} = \frac{K!}{i!\ (K-i)!}$$

The equation (1) is rewritten to simplify the calculation as follows:

$$p = \frac{\sum_{i=0}^{M} \binom{K}{i}}{2^{K}} \qquad (2)$$

If K is greater than 25, the probability is calculated by the normal approximation to the binomial distribution as follows:

Mean: $$\mu = KP = \frac{1}{2} K \qquad (3)$$

Standard deviation: $$\sigma = \sqrt{KPQ} = \frac{1}{2}\sqrt{K} \qquad (4)$$

Value of Z: $$Z = \frac{(M+0.5) - \mu}{\sigma} \qquad (5)$$

Then the probability, p, associated with the value as extreme as Z is obtained by the subroutine NDTR.

Note: p computed by this subroutine is the probability for a one-tailed test; thus for a two-tailed test, the user should double the value of p.

For reference see S. Siegel, Nonparametric Statistics for the Behavioral Sciences, McGraw-Hill, New York, 1956, chapter 5.

```
C
C     ..................................................................
C
C        SUBROUTINE SIGNT
C
C        PURPOSE
C           TO PERFORM A NON-PARAMETRIC SIGN TEST, GIVEN TWO SETS OF
C           MATCHED OBSERVATIONS.  IT TESTS THE NULL HYPOTHESIS THAT THE
C           DIFFERENCES BETWEEN EACH PAIR OF MATCHED OBSERVATIONS HAS A
C           MEDIAN EQUAL TO ZERO.
C
C        USAGE
C           CALL SIGNT (N,A,B,K,M,P,IE)
C
C        DESCRIPTION OF PARAMETERS
C           N - NUMBER OF OBSERVATIONS IN SETS A AND B
C           A - INPUT VECTOR OF LENGTH N CONTAINING DATA FROM THE FIRST
C               SAMPLE, A
C           B - INPUT VECTOR OF LENGTH N CONTAINING DATA FROM THE SECOND
C               SAMPLE, B
C           K - OUTPUT VARIABLE CONTAINING THE NUMBER OF PAIRS OF
C               OBSERVATIONS FROM THE TWO SAMPLES WHOSE DIFFERENCES ARE
C               NON-ZERO
C           M - OUTPUT VARIABLE CONTAINING THE NUMBER OF PLUS OR MINUS
C               DIFFERENCES, WHICHEVER IS FEWER.
C           P - COMPUTED PROBABILITY OF AS FEW AS M NUMBER OF PAIRS
C               HAVING THE SAME SIGN, ASSUMING THAT THE SAMPLES CAME
C               FROM THE SAME POPULATION.
C           IE- 0, IF THERE IS NO ERROR.
C               1, IF K IS ZERO.  IN THIS CASE, P IS SET TO 1.0 AND
C               M TO 0.
C
C        REMARKS
C           IF K IS LESS THAN OR EQUAL TO 25, THE PROBABILITY WILL BE
C           COMPUTED USING THE BINOMIAL DISTRIBUTION.  IF K IS GREATER
C           THAN 25, THE PROBABILITY WILL BE COMPUTED USING THE NORMAL
C           APPROXIMATION TO THE BINOMIAL DISTRIBUTION.
C           P COMPUTED IS THE PROBABILITY FOR A ONE-TAILED TEST.  THUS,
C           FOR A TWO TAILED TEST, DOUBLE THE VALUE FOR P.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NDTR
C
C        METHOD
C           REFER TO DIXON AND MASSEY, INTRODUCTION TO STATISTICAL
C           ANALYSIS (MCGRAW-HILL, 1957).
C
C     ..................................................................
C
      SUBROUTINE SIGNT (N,A,B,K,M,P,IE)
C
      DIMENSION A(1),B(1)
      DOUBLE PRECISION FN,FD
C
C        INITIALIZATION
C
      IE=0
      K=0
      MPLUS=0
      MMINS=0
C
C        FIND (+) OR (-) DIFFERENCE
C
      DO 40 I=1,N
      D=A(I)-B(I)
      IF(D) 20, 40, 30
C
C        (-) DIFFERENCE
C
   20 K=K+1
      MMINS=MMINS+1
      GO TO 40
C
C        (+) DIFFERENCE
C
   30 K=K+1
      MPLUS=MPLUS+1
C
   40 CONTINUE
      IF(K) 41,41,42
   41 IE=1
      P=1.0
      M=0
      GO TO 95
   42 FK=K
C
C        FIND THE NUMBER OF FEWER SIGNS
C
      IF(MPLUS-MMINS) 45, 45, 50
   45 M=MPLUS
      GO TO 55
   50 M=MMINS
C
C        TEST WHETHER K IS GREATER THAN 25
C
   55 IF(K-25) 60, 60, 77
C
C        K IS LESS THAN OR EQUAL TO 25
C
   60 P=1.0
      IF(M) 75, 75, 65
   65 FN=1.0
      FD=1.0
      DO 70 I=1,M
      FI=I
      FN=FN*(FK-(FI-1.0))
      FD=FD*FI
   70 P=P+FN/FD
C
   75 P=P/(2.0**K)
      GO TO 95
C
C        K IS GREATER THAN 25.  COMPUTE MEAN, STANDARD DEVIATION, AND Z
C
   77 U=0.5*FK
      S=0.5*SQRT(FK)
      FM=M
      IF(FM-U) 80, 85, 85
   80 CON=0.5
      GO TO 90
   85 CON=0.0
   90 Z=(FM+CON-U)/S
C
C        COMPUTE P ASSOCIATED WITH THE VALUE AS EXTREME AS Z
C
      CALL NDTR (Z,P,D)
C
   95 RETURN
      END
```

Subroutine SRANK

This subroutine measures the correlation between two variables by means of the Spearman rank correlation coefficient, given two vectors of n observations for the variables.

The observations on each variable are ranked from 1 to n. Tied observations are assigned the average of the tied ranks.

The sum of squares of rank differences is calculated:

$$D = \sum_{i=1}^{n} (A_i - B_i)^2 \qquad (1)$$

where A_i = first ranked vector

B_i = second ranked vector

n = number of ranks

A correction factor for ties is obtained:

$$T_a = \sum \frac{t^3 - t}{12} \quad \text{over variable A} \qquad (2)$$

$$T_b = \sum \frac{t^3 - t}{12} \quad \text{over variable B}$$

where t = number of observations tied for a given rank

The Spearman rank correlation coefficient is then computed for the following two cases:

(a) if T_a and T_b are zero,

$$r_s = 1 - \frac{6D}{n^3 - n} \qquad (3)$$

(b) if T_a and/or T_b are not zero,

$$r_s = \frac{X + Y - D}{2\sqrt{XY}} \qquad (4)$$

$$\text{where } X = \frac{n^3 - n}{12} - T_a \qquad (5)$$

$$Y = \frac{n^3 - n}{12} - T_b \qquad (6)$$

The statistic used to measure the significance of r_s is:

$$t = r_s \sqrt{\frac{n - 2}{1 - r_s^2}} \qquad (7)$$

The degrees of freedom are:

$$d.f. = n - 2 \qquad (8)$$

```
C                                                                       SRAN  10
C  ...................................................................SRAN  20
C                                                                       SRAN  30
C     SUBROUTINE SRANK                                                  SRAN  40
C                                                                       SRAN  50
C     PURPOSE                                                           SRAN  60
C        TEST CORRELATION BETWEEN TWO VARIABLES BY MEANS OF SPEARMAN    SRAN  70
C        RANK CORRELATION COEFFICIENT                                   SRAN  80
C                                                                       SRAN  90
C     USAGE                                                             SRAN 100
C        CALL SRANK(A,B,R,N,RS,T,NDF,NR)                                SRAN 110
C                                                                       SRAN 120
C     DESCRIPTION OF PARAMETERS                                         SRAN 130
C        A   - INPUT VECTOR OF N OBSERVATIONS FOR FIRST VARIABLE        SRAN 140
C        B   - INPUT VECTOR OF N OBSERVATIONS FOR SECOND VARIABLE       SRAN 150
C        R   - OUTPUT VECTOR FOR RANKED DATA, LENGTH IS 2*N. SMALLESTSRAN 160
C              OBSERVATION IS RANKED 1, LARGEST IS RANKED N. TIES       SRAN 170
C              ARE ASSIGNED AVERAGE OF TIED RANKS.                      SRAN 180
C        N   - NUMBER OF OBSERVATIONS                                   SRAN 190
C        RS  - SPEARMAN RANK CORRELATION COEFFICIENT (OUTPUT)           SRAN 200
C        T   - TEST OF SIGNIFICANCE OF RS (OUTPUT)                      SRAN 210
C        NDF - NUMBER OF DEGREES OF FREEDOM (OUTPUT)                    SRAN 220
C        NR  - CODE, 0 FOR UNRANKED DATA IN A AND B, 1 FOR RANKED       SRAN 230
C              DATA IN A AND B (INPUT)                                  SRAN 240
C                                                                       SRAN 250
C     REMARKS                                                           SRAN 260
C        T IS SET TO ZERO IF N IS LESS THAN TEN                         SRAN 270
C                                                                       SRAN 280
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     SRAN 290
C        RANK                                                           SRAN 300
C        TIE                                                            SRAN 310
C                                                                       SRAN 320
C     METHOD                                                            SRAN 330
C        DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE      SRAN 340
C        BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,             SRAN 350
C        CHAPTER 9                                                      SRAN 360
C                                                                       SRAN 370
C  ...................................................................SRAN 380
C                                                                       SRAN 390
      SUBROUTINE SRANK(A,B,R,N,RS,T,NDF,NR)                             SRAN 400
      DIMENSION A(1),B(1),R(1)                                          SRAN 410
C                                                                       SRAN 420
      FNNN=N*N*N-N                                                      SRAN 430
C                                                                       SRAN 440
C        DETERMINE WHETHER DATA IS RANKED                               SRAN 450
C                                                                       SRAN 460
      IF(NR-1) 5, 10, 5                                                 SRAN 470
C                                                                       SRAN 480
C        RANK DATA IN A AND B VECTORS AND ASSIGN TIED OBSERVATIONS      SRAN 490
C        AVERAGE OF TIED RANKS                                          SRAN 500
C                                                                       SRAN 510
    5 CALL RANK (A,R,N)                                                 SRAN 520
      CALL RANK (B,R(N+1),N)                                            SRAN 530
      GO TO 40                                                          SRAN 540
C                                                                       SRAN 550
C        MOVE RANKED DATA TO R VECTOR                                   SRAN 560
C                                                                       SRAN 570
   10 DO 20 I=1,N                                                       SRAN 580
   20 R(I)=A(I)                                                         SRAN 590
      DO 30 I=1,N                                                       SRAN 600
      J=I+N                                                             SRAN 610
   30 R(J)=B(I)                                                         SRAN 620
C                                                                       SRAN 630
C        COMPUTE SUM OF SQUARES OF RANK DIFFERENCES                     SRAN 640
C                                                                       SRAN 650
   40 D=0.0                                                             SRAN 660
      DO 50 I=1,N                                                       SRAN 670
      J=I+N                                                             SRAN 680
   50 D=D+(R(I)-R(J))*(R(I)-R(J))                                       SRAN 690
C                                                                       SRAN 700
C        COMPUTE TIED SCORE INDEX                                       SRAN 710
C                                                                       SRAN 720
      KT=1                                                              SRAN 730
      CALL TIE (R,N,KT,TSA)                                             SRAN 740
      CALL TIE (R(N+1),N,KT,TSB)                                        SRAN 750
C                                                                       SRAN 760
C        COMPUTE SPEARMAN RANK CORRELATION COEFFICIENT                  SRAN 770
C                                                                       SRAN 780
      IF(TSA) 60,55,60                                                  SRAN 790
   55 IF(TSB) 60,57,60                                                  SRAN 800
   57 RS=1.0-6.0*D/FNNN                                                 SRAN 810
      GO TO 70                                                          SRAN 820
   60 X=FNNN/12.0-TSA                                                   SRAN 830
      Y=X+TSA-TSB                                                       SRAN 840
      RS=(X+Y-D)/(2.0*(SQRT(X*Y)))                                      SRAN 850
C                                                                       SRAN 860
C        COMPUTE T AND DEGREES OF FREEDOM IF N IS 10 OR LARGER          SRAN 870
C                                                                       SRAN 880
      T=0.0                                                             SRAN 890
   70 IF(N-10) 80,75,75                                                 SRAN 900
   75 T=RS*SQRT(FLOAT(N-2)/(1.0-RS*RS))                                 SRAN 910
   80 NDF=N-2                                                           SRAN 920
      RETURN                                                            SRAN 930
      END                                                               SRAN 940
```

Subroutine TIE

```
C     ..................................................................
C        SUBROUTINE TIE
C
C        PURPOSE
C           CALCULATE CORRECTION FACTOR DUE TO TIES
C
C        USAGE
C           CALL TIE(R,N,KT,T)
C
C        DESCRIPTION OF PARAMETERS
C           R  - INPUT VECTOR OF RANKS OF LENGTH N CONTAINING VALUES
C                1 TO N
C           N  - NUMBER OF RANKED VALUES
C           KT - INPUT CODE FOR CALCULATION OF CORRECTION FACTOR
C                     1   SOLVE EQUATION 1
C                     2   SOLVE EQUATION 2
C           T  - CORRECTION FACTOR (OUTPUT)
C                   EQUATION 1   T=SUM(CT**3-CT)/12
C                   EQUATION 2   T=SUM(CT*(CT-1)/2)
C                 WHERE CT IS THE NUMBER OF OBSERVATIONS TIED FOR A
C                       GIVEN RANK
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           VECTOR IS SEARCHED FOR SUCCESSIVELY LARGER RANKS. TIES ARE
C           COUNTED AND CORRECTION FACTOR 1 OR 2 SUMMED.
C     ..................................................................
      SUBROUTINE TIE(R,N,KT,T)
      DIMENSION R(1)
C
C        INITIALIZATION
C
      T=0.0
      Y=0.0
    5 X=1.0E38
      IND=0
C
C        FIND NEXT LARGEST RANK
C
      DO 30 I=1,N
      IF(R(I)-Y) 30,30,10
   10 IF(R(I)-X) 20,30,30
   20 X=R(I)
      IND=IND+1
   30 CONTINUE
C
C        IF ALL RANKS HAVE BEEN TESTED, RETURN
C
      IF(IND) 90,90,40
   40 Y=X
      CT=0.0
C
C        COUNT TIES
C
      DO 60 I=1,N
      IF(R(I)-X) 60,50,60
   50 CT=CT+1.0
   60 CONTINUE
C
C        CALCULATE CORRECTION FACTOR
C
      IF(CT) 70,5,70
   70 IF(KT-1) 75,80,75
   75 T=T+CT*(CT-1.)/2.0
      GO TO 5
   80 T=T+(CT*CT*CT-CT)/12.0
      GO TO 5
   90 RETURN
      END
```

Subroutine TWOAV

This subroutine determines the Friedman two-way analysis of variance statistic, given a matrix A with n rows (groups) and m columns (cases). Data in each group is ranked from 1 to m. Tied observations are assigned the average of the tied ranks.

The sum of ranks is calculated:

$$R_j = \sum_{i=1}^{n} A_{ij} \qquad (1)$$

Friedman's statistic is then computed:

$$\chi_r^2 = \frac{12}{nm(m+1)} \sum_{j=1}^{m} (R_j)^2 - 3n(m+1) \qquad (2)$$

The degrees of freedom are:

$$\text{d.f.} = m - 1 \qquad (3)$$

```
C     ..................................................................
C        SUBROUTINE TWOAV
C
C        PURPOSE
C           TEST WHETHER A NUMBER OF SAMPLES ARE FROM THE SAME
C           POPULATION BY THE FRIEDMAN TWO-WAY ANALYSIS OF VARIANCE TEST
C
C        USAGE
C           CALL TWOAV(A,R,N,M,W,XR,NDF,NR)
C
C        DESCRIPTION OF PARAMETERS
C           A   - INPUT MATRIX, N BY M, OF ORIGINAL DATA
C           R   - OUTPUT MATRIX, N BY M, OF RANKED DATA
C           N   - NUMBER OF GROUPS
C           M   - NUMBER OF CASES IN EACH GROUP
C           W   - WORK AREA OF LENGTH 2*M
C           XR  - FRIEDMAN STATISTIC (OUTPUT)
C           NDF - NUMBER OF DEGREES OF FREEDOM (OUTPUT)
C           NR  - CODE, 0 FOR UNRANKED DATA IN A, 1 FOR RANKED DATA
C                 IN A (INPUT)
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           RANK
C
C        METHOD
C           DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE
C           BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,
C           CHAPTER 7
C     ..................................................................
      SUBROUTINE TWOAV (A,R,N,M,W,XR,NDF,NR)
      DIMENSION A(1),R(1),W(1)
C
C        DETERMINE WHETHER DATA IS RANKED
C
      IF(NR-1) 10, 30, 10
C
C        RANK DATA IN EACH GROUP AND ASSIGN TIED OBSERVATIONS AVERAGE
C        OF TIED RANK
C
   10 DO 20 I=1,N
      IJ=I-N
      IK=IJ
      DO 15 J=1,M
      IJ=IJ+N
   15 W(J)=A(IJ)
      CALL RANK (W,W(M+1),M)
      DO 20 J=1,M
      IK=IK+N
      IW=M+J
   20 R(IK)=W(IW)
      GO TO 35
   30 NM=N*M
      DO 32 I=1,NM
   32 R(I)=A(I)
C
C        CALCULATE SUM OF SQUARES OF SUMS OF RANKS
C
   35 RTSQ=0.0
      IR=0
      DO 50 J=1,M
      RT=0.0
      DO 40 I=1,N
      IR=IR+1
   40 RT=RT+R(IR)
   50 RTSQ=RTSQ+RT*RT
C
C        CALCULATE FRIEDMAN TEST VALUE, XR
C
      FNM=N*(M+1)
      FM=M
      XR=(12.0/(FM*FNM))*RTSQ-3.0*FNM
C
C        FIND DEGREES OF FREEDOM
C
      NDF=M-1
      RETURN
      END
```

Subroutine UTEST

This subroutine tests whether two independent groups are from the same population by means of the Mann-Whitney U-test, given an input vector A with the smaller group preceding the larger group. The scores for both groups are ranked together in ascending order. Tied observations are assigned the average of the tied ranks.

The sum of ranks in the larger group, R2, is calculated. The U statistic is then computed as follows:

$$U' = n_1 n_2 + \frac{n_2 (n_2 + 1)}{2} - R_2 \qquad (1)$$

where n_1 = number of cases in smaller group

n_2 = number of cases in larger group

$$U = n_1 n_2 - U'$$

$$\text{if } U' < U, \text{ set } U = U' \qquad (2)$$

A correction factor for ties is obtained:

$$T = \sum \frac{t^3 - t}{12} \qquad (3)$$

where t = number of observations tied for a given rank

The standard deviation is computed for two cases:

(a) if T = 0

$$s = \sqrt{\frac{n_1 n_2 (n_1 + n_2 + 1)}{12}} \qquad (4)$$

(b) if T > 0

$$s = \sqrt{\left(\frac{n_1 n_2}{N(N-1)}\right)\left(\frac{N^3 - N}{12} - T\right)} \qquad (5)$$

where N = total number of cases $(n_1 + n_2)$

The significance of U can be measured by

$$Z = \frac{U - \overline{X}}{s} \qquad (6)$$

where $\overline{X}$ = mean $= \frac{n_1 n_2}{2}$

Z is set to zero if n_2 is less than 20.

```
C                                                                       UTES  10
C     ......................................................................UTES  20
C                                                                       UTES  30
C        SUBROUTINE UTEST                                               UTES  40
C                                                                       UTES  50
C        PURPOSE                                                        UTES  60
C           TEST WHETHER TWO INDEPENDENT GROUPS ARE FROM THE SAME       UTES  70
C           POPULATION BY MEANS OF MANN-WHITNEY U-TEST                  UTES  80
C                                                                       UTES  90
C        USAGE                                                          UTES 100
C           CALL UTEST(A,R,N1,N2,U,Z,IER)                               UTES 110
C                                                                       UTES 120
C        DESCRIPTION OF PARAMETERS                                      UTES 130
C           A  - INPUT VECTOR OF CASES CONSISTING OF TWO INDEPENDENT    UTES 140
C                GROUPS . SMALLER GROUP PRECEDES LARGER GROUP. LENGTH   UTES 150
C                IS N1+N2.                                              UTES 160
C           R  - OUTPUT VECTOR OF RANKS. SMALLEST VALUE IS RANKED 1,    UTES 170
C                LARGEST IS RANKED N. TIES ARE ASSIGNED AVERAGE OF TIED UTES 180
C                RANKS. LENGTH IS N1+N2.                                UTES 190
C           N1 - NUMBER OF CASES IN SMALLER GROUP                       UTES 200
C           N2 - NUMBER OF CASES IN LARGER GROUP                        UTES 210
C           U  - STATISTIC USED TO TEST HOMOGENEITY OF THE TWO          UTES 220
C                GROUPS (OUTPUT)                                        UTES 230
C           Z  - MEASURE OF SIGNIFICANCE OF U IN TERMS OF NORMAL        UTES 240
C                DISTRIBUTION (OUTPUT)                                  UTES 250
C           IER- 0, IF NO ERROR.                                        UTES 251
C              - 1, IF ALL VALUES OF ONE GROUP ARE TIED.                UTES 252
C                                                                       UTES 260
C        REMARKS                                                        UTES 270
C           Z IS SET TO ZERO IF N2 IS LESS THAN 20                      UTES 280
C                                                                       UTES 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  UTES 300
C           RANK                                                        UTES 310
C           TIE                                                         UTES 320
C                                                                       UTES 330
C        METHOD                                                         UTES 340
C           DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE   UTES 350
C           BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,          UTES 360
C           CHAPTER 6                                                   UTES 370
C                                                                       UTES 380
C     ......................................................................UTES 390
C                                                                       UTES 400
      SUBROUTINE UTEST(A,R,N1,N2,U,Z)                                   UTES 410
      DIMENSION A(1),R(1)                                               UTES 420
C                                                                       UTES 430
C        RANK SCORES FROM BOTH GROUP TOGETHER IN ASCENDING ORDER, AND   UTES 440
C        ASSIGN TIED OBSERVATIONS AVERAGE OF TIED RANKS                 UTES 450
C                                                                       UTES 460
      N=N1+N2                                                           UTES 470
      CALL RANK(A,R,N)                                                  UTES 480
      Z=0.0                                                             UTES 490
C                                                                       UTES 500
C        SUM RANKS IN LARGER GROUP                                      UTES 510
C                                                                       UTES 520
      R2=0.0                                                            UTES 530
      NP=N1+1                                                           UTES 540
      DO 10 I=NP,N                                                      UTES 550
   10 R2=R2+R(I)                                                        UTES 560
C                                                                       UTES 570
C        CALCULATE U                                                    UTES 580
C                                                                       UTES 590
      FNX=N1*N2                                                         UTES 600
      FN=N                                                              UTES 610
      FN2=N2                                                            UTES 620
      UP=FNX+FN2*((FN2+1.0)/2.0)-R2                                     UTES 630
      U=FNX-UP                                                          UTES 640
      IF(UP-U) 20,20,30                                                 UTES 650
   20 U=UP                                                              UTES 660
C                                                                       UTES 670
C        TEST FOR N2 LESS THAN 20                                       UTES 680
C                                                                       UTES 690
   30 IF(N2-20) 80,40,40                                                UTES 700
C                                                                       UTES 710
C        COMPUTE STANDARD DEVIATION                                     UTES 720
C                                                                       UTES 730
   40 KT=1                                                              UTES 740
      CALL TIE(R,N,KT,TS)                                               UTES 750
      IF(TS) 50,60,50                                                   UTES 760
   50 IF (TS-(FN*FN*FN-FN)/12)52,51,52                                  UTES 761
   51 IER=1                                                             UTES 762
      GO TO 80                                                          UTES 763
   52 S=SQRT((FNX/(FN*(FN-1.0)))*(((FN*FN*FN-FN)/12.0)-TS))             UTES 770
      GO TO 70                                                          UTES 780
   60 S=SQRT(FN*(FN+1.0)/12.0)                                          UTES 790
C                                                                       UTES 800
C        COMPUTE Z                                                      UTES 810
C                                                                       UTES 820
   70 Z=(U-FNX*0.5)/S                                                   UTES 830
   80 RETURN                                                            UTES 840
      END                                                               UTES 850
```

Subroutine WTEST

This subroutine computes the Kendall coefficient of concordance, given a matrix A of n rows (variables) and m columns (cases). The observations on all variables are ranked from 1 to m. Tied observations are assigned the average of the tied ranks.

A correction factor for ties is obtained:

$$T = \sum_{i=1}^{n} \frac{t^3 - t}{12} \tag{1}$$

where t = number of observations tied for a given rank

Sums of ranks are calculated:

$$Y_j = \sum_{i=1}^{n} R_{ij} \tag{2}$$

where j = 1, 2, . . . , m

From these, the mean of sums of ranks is found:

$$\bar{R} = \frac{\sum_{j=1}^{m} Y_j}{m} \tag{3}$$

The sum of squares of deviations is derived:

$$s = \sum_{j=1}^{m} (Y_j - \bar{R})^2 \tag{4}$$

The Kendall coefficient of concordance is then computed:

$$W = \frac{s}{\frac{1}{12} n^2 (m^3 - m) - nT} \tag{5}$$

For m larger than 7, chi-square is:

$$\chi^2 = n(m-1)W \tag{6}$$

The degrees of freedom are:

$$\text{d.f.} = n - 1 \tag{7}$$

```
C                                                                       WTES  10
C     ..................................................................WTES  20
C                                                                       WTES  30
C        SUBROUTINE WTEST                                               WTES  40
C                                                                       WTES  50
C        PURPOSE                                                        WTES  60
C           TEST DEGREE OF ASSOCIATION AMONG A NUMBER OF VARIABLES BY   WTES  70
C           THE KENDALL COEFFICIENT OF CONCORDANCE                      WTES  80
C                                                                       WTES  90
C        USAGE                                                          WTES 100
C           CALL WTEST(A,R,N,M,WA,W,CS,NDF,NR)                          WTES 110
C                                                                       WTES 120
C        DESCRIPTION OF PARAMETERS                                      WTES 130
C           A   - INPUT MATRIX, N BY M, OF ORIGINAL DATA                WTES 140
C           R   - OUTPUT MATRIX, N BY M, OF RANKED DATA.SMALLEST VALUE  WTES 150
C                 IS RANKED 1, LARGEST IS RANKED N. TIES ARE ASSIGNED   WTES 160
C                 AVERAGE OF TIED RANKS                                 WTES 170
C           N   - NUMBER OF VARIABLES                                   WTES 180
C           M   - NUMBER OF CASES                                       WTES 190
C           WA  - WORK AREA VECTOR OF LENGTH 2*M                        WTES 200
C           W   - KENDALL COEFFICIENT OF CONCORDANCE(OUTPUT)            WTES 210
C           CS  - CHI-SQUARE (OUTPUT)                                   WTES 220
C           NDF - NUMBER OF DEGREES OF FREEDOM (OUTPUT)                 WTES 230
C           NR  - CODE, 0 FOR UNRANKED DATA IN A, 1 FOR RANKED DATA     WTES 240
C                 IN A (INPUT)                                          WTES 250
C                                                                       WTES 260
C        REMARKS                                                        WTES 270
C           CHI-SQUARE IS SET TO ZERO IF M IS 7 OR SMALLER              WTES 280
C                                                                       WTES 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  WTES 300
C           RANK                                                        WTES 310
C           TIE                                                         WTES 320
C                                                                       WTES 330
C        METHOD                                                         WTES 340
C           DESCRIBED IN S. SIEGEL, 'NONPARAMETRIC STATISTICS FOR THE   WTES 350
C           BEHAVIORAL SCIENCES', MCGRAW-HILL, NEW YORK, 1956,          WTES 360
C           CHAPTER 9                                                   WTES 370
C     ..................................................................WTES 380
C                                                                       WTES 390
C                                                                       WTES 400
      SUBROUTINE WTEST (A,R,N,M,WA,W,CS,NDF,NR)                         WTES 410
      DIMENSION A(1),R(1),WA(1)                                         WTES 420
C                                                                       WTES 430
      FM=M                                                              WTES 440
      FN=N                                                              WTES 450
C                                                                       WTES 460
C        DETERMINE WHETHER DATA IS RANKED                               WTES 470
C        RANK DATA FOR ALL VARIABLES ASSIGNING TIED OBSERVATIONS AVERAGEWTES 480
C        OF TIED RANKS AND COMPUTE CORRECTION FOR TIED SCORES           WTES 490
C                                                                       WTES 500
      T=0.0                                                             WTES 510
      KT=1                                                              WTES 520
      DO 20 I=1,N                                                       WTES 530
      IJ=I-N                                                            WTES 540
      IK=IJ                                                             WTES 550
      IF(NR-1) 5,2,5                                                    WTES 560
    2 DO 3 J=1,M                                                        WTES 570
      IJ=IJ+N                                                           WTES 580
      K=M+J                                                             WTES 590
    3 WA(K)=A(IJ)                                                       WTES 600
      GO TO 15                                                          WTES 610
    5 DO 10 J=1,M                                                       WTES 620
      IJ=IJ+N                                                           WTES 630
   10 WA(J)=A(IJ)                                                       WTES 640
      CALL RANK(WA,WA(M+1),M)                                           WTES 650
   15 CALL TIE(WA(M+1),M,KT,TI)                                         WTES 660
      T=T+TI                                                            WTES 670
      DO 20 J=1,M                                                       WTES 680
      IK=IK+N                                                           WTES 690
      IW=M+J                                                            WTES 700
   20 R(IK)=WA(IW)                                                      WTES 710
C                                                                       WTES 720
C        CALCULATE VECTOR OF SUMS OF RANKS                              WTES 730
C                                                                       WTES 740
      IR=0                                                              WTES 750
      DO 40 J=1,M                                                       WTES 760
      WA(J)=0.0                                                         WTES 770
      DO 40 I=1,N                                                       WTES 780
      IR=IR+1                                                           WTES 790
   40 WA(J)=WA(J)+R(IR)                                                 WTES 800
C                                                                       WTES 810
C        COMPUTE MEAN OF SUMS OF RANKS                                  WTES 820
C                                                                       WTES 830
      SM=0.0                                                            WTES 840
      DO 50 J=1,M                                                       WTES 850
   50 SM=SM+WA(J)                                                       WTES 860
      SM=SM/FM                                                          WTES 870
C                                                                       WTES 880
C        COMPUTE SUM OF SQUARES OF DEVIATIONS                           WTES 890
C                                                                       WTES 900
      S=0.0                                                             WTES 910
      DO 60 J=1,M                                                       WTES 920
   60 S=S+(WA(J)-SM)*(WA(J)-SM)                                         WTES 930
C                                                                       WTES 940
C        COMPUTE W                                                      WTES 950
C                                                                       WTES 960
      W=S/(((FN*FN)*(FM*FM*FM-FM)/12.0)-FN*T)                           WTES 970
C                                                                       WTES 980
C        COMPUTE DEGREES OF FREEDOM AND CHI-SQUARE IF M IS OVER 7       WTES 990
C                                                                       WTES1000
      CS=0.0                                                            WTES1010
      NDF=0                                                             WTES1020
      IF(M-7) 70,70,65                                                  WTES1030
   65 CS=FN*(FM-1.0)*W                                                  WTES1040
      NDF=M-1                                                           WTES1050
   70 RETURN                                                            WTES1060
      END                                                               WTES1070
```

Generation of Random Variates; Distribution Functions

Generators

Subroutine RANDU

```
C
C     ..................................................................
C
C        SUBROUTINE RANDU
C
C        PURPOSE
C           COMPUTES UNIFORMLY DISTRIBUTED RANDOM REAL NUMBERS BETWEEN
C           0 AND 1.0 AND RANDOM INTEGERS BETWEEN ZERO AND
C           2**31. EACH ENTRY USES AS INPUT AN INTEGER RANDOM NUMBER
C           AND PRODUCES A NEW INTEGER AND REAL RANDOM NUMBER.
C
C        USAGE
C           CALL RANDU(IX,IY,YFL)
C
C        DESCRIPTION OF PARAMETERS
C           IX - FOR THE FIRST ENTRY THIS MUST CONTAIN ANY ODD INTEGER
C                NUMBER WITH NINE OR LESS DIGITS. AFTER THE FIRST ENTRY,
C                IX SHOULD BE THE PREVIOUS VALUE OF IY COMPUTED BY THIS
C                SUBROUTINE.
C           IY - A RESULTANT INTEGER RANDOM NUMBER REQUIRED FOR THE NEXT
C                ENTRY TO THIS SUBROUTINE. THE RANGE OF THIS NUMBER IS
C                BETWEEN ZERO AND 2**31
C           YFL- THE RESULTANT UNIFORMLY DISTRIBUTED, FLOATING POINT,
C                RANDOM NUMBER IN THE RANGE 0 TO 1.0
C
C        REMARKS
C           THIS SUBROUTINE IS SPECIFIC TO SYSTEM/360 AND WILL PRODUCE
C           2**29 TERMS BEFORE REPEATING. THE REFERENCE BELOW DISCUSSES
C           SEEDS (65539 HERE), RUN PROBLEMS, AND PROBLEMS CONCERNING
C           RANDOM DIGITS USING THIS GENERATION SCHEME. MACLAREN AND
C           MARSAGLIA, JACM 12, P. 83-89, DISCUSS CONGRUENTIAL
C           GENERATION METHODS AND TESTS. THE USE OF TWO GENERATORS OF
C           THE RANDU TYPE, ONE FILLING A TABLE AND ONE PICKING FROM THE
C           TABLE, IS OF BENEFIT IN SOME CASES. 65549 HAS BEEN
C           SUGGESTED AS A SEED WHICH HAS BETTER STATISTICAL PROPERTIES
C           FOR HIGH ORDER BITS OF THE GENERATED DEVIATE.
C           SEEDS SHOULD BE CHOSEN IN ACCORDANCE WITH THE DISCUSSION
C           GIVEN IN THE REFERENCE BELOW. ALSO, IT SHOULD BE NOTED THAT
C           IF FLOATING POINT RANDOM NUMBERS ARE DESIRED,AS ARE
C           AVAILABLE FROM RANDU, THE RANDOM CHARACTERISTICS OF THE
C           FLOATING POINT DEVIATES ARE MODIFIED AND IN FACT THESE
C           DEVIATES HAVE HIGH PROBABILITY OF HAVING A TRAILING LOW
C           ORDER ZERO BIT IN THEIR FRACTIONAL PART.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           POWER RESIDUE METHOD DISCUSSED IN IBM MANUAL C20-8011,
C           RANDOM NUMBER GENERATION AND TESTING
C
C     ..................................................................
C
      SUBROUTINE RANDU(IX,IY,YFL)
      IY=IX*65539
      IF(IY)5,6,6
    5 IY=IY+2147483647+1
    6 YFL=IY
      YFL=YFL*.4656613E-9
      RETURN
      END
```

Subroutine GAUSS

This subroutine computes a normally distributed random number with a given mean and standard deviation.

An approximation to normally distributed random numbers Y can be found from a sequence of uniform random numbers† using the formula:

$$Y = \frac{\sum_{i=1}^{K} X_i - \frac{K}{2}}{\sqrt{K/12}} \qquad (1)$$

where X_i is a uniformly distributed random number, $0 < X_i < 1$

K is the number of values X_i to be used

Y approaches a true normal distribution asymptotically as K approaches infinity. For this subroutine, K was chosen as 12 to reduce execution time. Equation (1) thus becomes:

$$Y = \sum_{i=1}^{12} X_i - 6.0$$

The adjustment for the required mean and standard deviation is then:

$$Y' = Y * S + AM \qquad (2)$$

where Y' is the required normally distributed random number

S is the required standard deviation

AM is the required mean

```
C
C     ..................................................................
C
C        SUBROUTINE GAUSS
C
C        PURPOSE
C           COMPUTES A NORMALLY DISTRIBUTED RANDOM NUMBER WITH A GIVEN
C           MEAN AND STANDARD DEVIATION
C
C        USAGE
C           CALL GAUSS(IX,S,AM,V)
C
C        DESCRIPTION OF PARAMETERS
C           IX -IX MUST CONTAIN AN ODD INTEGER NUMBER WITH NINE OR
C               LESS DIGITS ON THE FIRST ENTRY TO GAUSS. THEREAFTER
C               IT WILL CONTAIN A UNIFORMLY DISTRIBUTED INTEGER RANDOM
C               NUMBER GENERATED BY THE SUBROUTINE FOR USE ON THE NEXT
C               ENTRY TO THE SUBROUTINE.
C           S  -THE DESIRED STANDARD DEVIATION OF THE NORMAL
C               DISTRIBUTION.
C           AM -THE DESIRED MEAN OF THE NORMAL DISTRIBUTION
C           V  -THE VALUE OF THE COMPUTED NORMAL RANDOM VARIABLE
C
C        REMARKS
C           THIS SUBROUTINE USES RANDU WHICH IS MACHINE SPECIFIC
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           RANDU
C
C        METHOD
C           USES 12 UNIFORM RANDOM NUMBERS TO COMPUTE NORMAL RANDOM
C           NUMBERS BY CENTRAL LIMIT THEOREM. THE RESULT IS THEN
C           ADJUSTED TO MATCH THE GIVEN MEAN AND STANDARD DEVIATION.
C           THE UNIFORM RANDOM NUMBERS COMPUTED WITHIN THE SUBROUTINE
C           ARE FOUND BY THE POWER RESIDUE METHOD.
C
C     ..................................................................
C
      SUBROUTINE GAUSS(IX,S,AM,V)
      A=0.0
      DO 50 I=1,12
      CALL RANDU(IX,IY,Y)
      IX=IY
   50 A=A+Y
      V=(A-6.0)*S+AM
      RETURN
      END
```

† R. W. Hamming, Numerical Methods for Scientists and Engineers, McGraw-Hill, N.Y., 1962, pages 34 and 389.

Distribution Functions

Subroutine NDTR

This subroutine computes $y = P(x) = \text{Prob}\ (X \le x)$, where X is a random variable distributed normally with mean zero and variance one.

$$P(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-u^2/2)du$$

The following approximation† is used:

$$P(x) = 1 - f(x) \sum_{i=1}^{5} a_i w^i;\ x \ge 0$$

where:

$$w = 1/(1 + px)$$

$$f(x) = \exp(-x^2/2)/\sqrt{2\pi}$$

$$P = 0.2316419$$

$$a_1 = 0.3193815$$

$$a_2 = -0.3565638$$

$$a_3 = 1.781478$$

$$a_4 = -1.821256$$

$$a_5 = 1.330274$$

The maximum error is $7(10^{-7})$; f(x) is also presented in output.

---

†C. Hastings, Approximations for Digital Computers. Princeton University Press, Princeton, N.J., 1955.
M. Abramowitz and I. A. Stegun, Handbook of Mathematical Functions. Dover Publications, Inc., N.Y., equation 26.2.17.

```
C.....................................................................
C
C       SUBROUTINE NDTR
C
C       PURPOSE
C          COMPUTES Y = P(X) = PROBABILITY THAT THE RANDOM VARIABLE  U,
C          DISTRIBUTED NORMALLY(0,1), IS LESS THAN OR EQUAL TO X.
C          F(X), THE ORDINATE OF THE NORMAL DENSITY AT X, IS ALSO
C          COMPUTED.
C
C       USAGE
C          CALL NDTR(X,P,D)
C
C       DESCRIPTION OF PARAMETERS
C          X--INPUT SCALAR FOR WHICH P(X) IS COMPUTED.
C          P--OUTPUT PROBABILITY.
C          D--OUTPUT DENSITY.
C
C       REMARKS
C          MAXIMUM ERROR IS 0.0000007.
C
C       SUBROUTINES AND SUBPROGRAMS REQUIRED
C          NONE
C
C       METHOD
C          BASED ON APPROXIMATIONS IN C. HASTINGS, APPROXIMATIONS FOR
C          DIGITAL COMPUTERS, PRINCETON UNIV. PRESS, PRINCETON, N.J.,
C          1955.  SEE EQUATION 26.2.17, HANDBOOK OF MATHEMATICAL
C          FUNCTIONS, ABRAMOWITZ AND STEGUN, DOVER PUBLICATIONS, INC.,
C          NEW YORK.
C
C.....................................................................
C
      SUBROUTINE NDTR(X,P,D)
C
      AX=ABS(X)
      T=1.0/(1.0+.2316419*AX)
      D=0.3989423*EXP(-X*X/2.0)
      P = 1.0 - D*T*((((1.330274*T - 1.821256)*T + 1.781478)*T -
     1  0.3565638)*T + 0.3193815)
      IF(X)1,2,2
    1 P=1.0-P
    2 RETURN
      END
```

Subroutine BDTR

This subroutine computes $P = I_x(m,n) = \text{Prob}\ (X \le x)$ where X is a random variable following the beta distribution with degrees of freedom (continuous parameters) m and n. For computation to take place, $0 \le x \le 1$, $0.5 \le m \le 10^{+5}$, and $0.5 \le n \le 10^{+5}$. D, the ordinate of the beta density at x, is also presented in the output.

For $0 \le x \le 1$, $I_x(m,n)$ may be written as:

$$I_x(m,n) = \int_0^x f(m,n,y)\,dy$$

where:

$$f(m,n,y) = \frac{1}{B(m,n)}\, y^{m-1}(1-y)^{n-1}$$

$$B(m,n) = \frac{\Gamma(m)\,\Gamma(n)}{\Gamma(m+n)};\ D = f(m,n,x) \qquad (1)$$

$I_x(m,n)$ can be reduced to a binomial partial sum which can be evaluated by means of a continued fraction expansion.

Let $N = m+n-1$ and $r = m-1$. Then:

$$I_x(m,n) = I_x(r+1,\ N-r)$$

$$I_x(r+1,\ N-r) = \sum_{s=r+1}^{N} \binom{N}{s} x^s (1-x)^{N-s}$$

$$= \binom{N}{r+1} x^{r+1} (1-x)^{N-r-1} S \qquad (2)$$

where $0 \le s \le N$

S is a continued fraction, with 80 terms being sufficient for the desired accuracy.

$$S = \frac{1}{1-}\ \frac{c_1}{1+}\ \frac{d_1}{1-}\ \frac{c_2}{1+}\ \frac{d_2}{1-} \cdots \frac{c_{80}}{1+}\ \frac{d_{80}}{1} \qquad (3)$$

$$c_i = \frac{(N-i-r)\,(r+i)}{(r+2i-1)\,(r+2i)}\ \frac{x}{1-x} \qquad (4)$$

$$d_i = \frac{i\,(N+i)}{(r+2i+1)\,(r+2i)}\ \frac{x}{1-x} \qquad (5)$$

The above continued fraction expansion of $I_x(m,n)$ holds for positive m and n (integers or nonintegers), $N \geq 0$ $(m+n \geq 1)$, and $r \geq 0$ $(m \geq 1)$. In order to fulfill these last two conditions, if $m < 1$, the following transformation must be made before computation of $I_x(m,n)$ can take place:

$$I_x(m,n) = \frac{\Gamma(m+n)}{\Gamma(m+1)\Gamma(n)} x^m (1-x)^n + I_x(m+1,n) \tag{6}$$

The quantities on the right-hand side of equation (6) are those which are computed.

It is known that $I_x(m,n) = I_{1-x}(n,m)$. Thus, either of the two parameter sets indicated by this equation may be used in computing the beta integral. The parameter set selection is made by applying the following empirically derived rule:

> Let p and q be the degrees of freedom corresponding to z, where $z = x$ if $x \leq .5$ or $(1-x)$ otherwise. If the quantity $[(p-1) - (p+q-1) z^{6/5} + 2]$ is positive, use the parameter set corresponding to z. Otherwise, use the parameter set corresponding to (1-z).

If $0 \leq x \leq 10^{-8}$ or $0 \leq 1-x \leq 10^{-8}$, the approximation is made that x = 0 or 1 respectively. P and D are then set according to the following table:

$0 \leq x \leq 10^{-8}$		$0 \leq 1-x \leq 10^{-8}$	
P = 0		P = 1	
If:	then:	If:	then:
$A < 1$	$D = 10^{75}$	$B < 1$	$D = 10^{75}$
$A = 1$	$D = 1/B(m,n)$	$B = 1$	$D = 1/B(m,n)$
$A > 1$	$D = 0$	$B > 1$	$D = 0$

If either or both m and n are within 10^{-8} of 1, the beta integral is solved explicitly for m = 1, n = 1, or m = n = 1:

If:	then:
$A = 1,\ B = 1$	$P = x$
$A = 1,\ B \neq 1$	$P = 1 - (1-x)^n$
$A \neq 1,\ B = 1$	$P = x^m$

If m or n is greater than 1000, the chi-square approximation is used:

$z_1 = 2m(1-x)/x$ is distributed as χ^2 with 2n degrees of freedom and $P = 1 - P_{\chi 2}(z_1)$ for m > 1000.

$z_2 = 2nx/(1-x)$ is distributed as χ^2 with 2 m degrees of freedom and $P = P_{\chi 2}(z_2)$ for n > 1000. If both m and n are greater than 1000, the approximation corresponding to z_1 is used.

The values of P very near zero or one may be somewhat imprecise. To eliminate possible misinterpretation of results, if $0 \leq P \leq 10^{-8}$ or $0 \leq 1-P \leq 10^{-8}$, P is set to 0 or 1 respectively.

For reference see:

(1) R. E. Bargmann and S. P. Ghosh, Statistical Description Programs For a Computer Language. IBM Research Report RC-1094, 1963.

(2) M. Abramowitz and I. A. Stegun, Handbook of Mathematical Functions. U. S. Department of Commerce, National Bureau of Standards Applied Mathematics Series, 1966.

```
C                                                                       BDTR  10
C     ..................................................................BDTR  20
C                                                                       BDTR  30
C        SUBROUTINE BDTR                                                BDTR  40
C                                                                       BDTR  50
C        PURPOSE                                                        BDTR  60
C           COMPUTES P(X) = PROBABILITY THAT THE RANDOM VARIABLE U,     BDTR  70
C           DISTRIBUTED ACCORDING TO THE BETA DISTRIBUTION WITH         BDTR  80
C           PARAMETERS A AND B, IS LESS THAN OR EQUAL TO X.  F(A,B,X),  BDTR  90
C           THE ORDINATE OF THE BETA DENSITY AT X, IS ALSO COMPUTED.    BDTR 100
C                                                                       BDTR 110
C        USAGE                                                          BDTR 120
C           CALL BDTR(X,A,B,P,D,IER)                                    BDTR 130
C                                                                       BDTR 140
C        DESCRIPTION OF PARAMETERS                                      BDTR 150
C           X   - INPUT SCALAR FOR WHICH P(X) IS COMPUTED.              BDTR 160
C           A   - BETA DISTRIBUTION PARAMETER (CONTINUOUS).             BDTR 170
C           B   - BETA DISTRIBUTION PARAMETER (CONTINUOUS).             BDTR 180
C           P   - OUTPUT PROBABILITY.                                   BDTR 190
C           D   - OUTPUT DENSITY.                                       BDTR 200
C           IER - RESULTANT ERROR CODE WHERE                            BDTR 210
C                 IER= 0 --- NO ERROR                                   BDTR 220
C                 IER=-1,+1  CDTF HAS BEEN CALLED AND AN ERROR HAS      BDTR 230
C                            OCCURRED.  SEE CDTR.                       BDTR 240
C                 IER=-2 --- AN INPUT PARAMETER IS INVALID.  X IS LESS  BDTR 250
C                            THAN 0.0 OR GREATER THAN 1.0, OR EITHER A OR BDTR 260
C                            B IS LESS THAN 0.5 OR GREATER THAN 10**(+5). BDTR 270
C                            P AND D ARE SET TO -1.E75.                 BDTR 280
C                 IER=+2 --- INVALID OUTPUT.  P IS LESS THAN ZERO OR    BDTR 290
C                            GREATER THAN ONE.  P IS SET TO 1.E75.      BDTR 300
C                                                                       BDTR 310
C        REMARKS                                                        BDTR 320
C           SEE MATHEMATICAL DESCRIPTION.                               BDTR 330
C                                                                       BDTR 340
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  BDTR 350
C           DLGAM                                                       BDTR 360
C           NDTR                                                        BDTR 370
C           CDTF                                                        BDTR 380
C                                                                       BDTR 390
C        METHOD                                                         BDTR 400
C           REFER TO R.E. BARGMANN AND S.P. GHOSH, STATISTICAL          BDTR 410
C           DISTRIBUTION PROGRAMS FOR A COMPUTER LANGUAGE,              BDTR 420
C           IBM RESEARCH REPORT RC-1094, 1963.                          BDTR 430
C                                                                       BDTR 440
C     ..................................................................BDTR 450
C                                                                       BDTR 460
      SUBROUTINE BDTR(X,A,B,P,D,IER)                                    BDTR 470
      DOUBLE PRECISION XX,DLXX,DL1X,AA,BB,G1,G2,G3,G4,DD,PP,XO,FF,FN,   BDTR 480
     1XI,SS,CC,RR,DLBETA                                                BDTR 490
C                                                                       BDTR 500
C        TEST FOR VALID INPUT DATA                                      BDTR 510
C                                                                       BDTR 520
      IF(A-(.5-1.E-5)) 640,10,10                                        BDTR 530
   10 IF(B-(.5-1.E-5)) 640,20,20                                        BDTR 540
   20 IF(A-1.E+5) 30,30,640                                             BDTR 550
   30 IF(B-1.E+5) 40,40,640                                             BDTR 560
   40 IF(X) 640,50,50                                                   BDTR 570
   50 IF(1.-X) 640,60,60                                                BDTR 580
C                                                                       BDTR 590
```

```
C        COMPUTE LOG(BETA(A,B))
C
   60 AA=DBLE(A)
      BB=DBLE(B)
      CALL DLGAM(AA,G1,IOK)
      CALL DLGAM(BB,G2,IOK)
      CALL DLGAM(AA+BB,G3,IOK)
      DLBETA=G1+G2-G3
C
C        TEST FOR X NEAR 0.0 OR 1.0
C
      IF(X-1.E-8) 80,80,70
   70 IF((1.-X)-1.E-8) 130,130,140
   80 P=0.0
      IF(A-1.) 90,100,120
   90 D=1.E+75
      GO TO 660
  100 DD=-DLBETA
      IF(DD+1.68D02)  120,120,110
  110 DD=DEXP(DD)
      D=SNGL(DD)
      GO TO 660
  120 D=0.0
      GO TO 660
  130 P=1.0
      IF(B-1.) 90,100,120
C
C        SET PROGRAM PARAMETERS
C
  140 XX=DBLE(X)
      DLXX=DLOG(XX)
      DL1X=DLOG(1.D0-XX)
      XO=XX/(1.D0-XX)
      ID=0
C
C        COMPUTE ORDINATE
C
      DD=(AA-1.D0)*DLXX+(BB-1.D0)*DL1X-DLBETA
      IF(DD-1.68D02) 150,150,160
  150 IF(DD+1.68D02) 170,170,180
  160 D=1.E75
      GO TO 190
  170 D=0.0
      GO TO 190
  180 DD=DEXP(DD)
      D=SNGL(DD)
C
C        A OR B OR BOTH WITHIN 1.E-8 OF 1.0
C
  190 IF(ABS(A-1.)-1.E-8)  200,200,210
  200 IF(ABS(B-1.)-1.E-8)  220,220,230
  210 IF(ABS(B-1.)-1.E-8)  260,260,290
  220 P=X
      GO TO 660
  230 PP=BB*DL1X
      IF(PP+1.68D02) 240,240,250
  240 P=1.0
      GO TO 660
  250 PP=DEXP(PP)
      PP=1.D0-PP
      P=SNGL(PP)
      GO TO 600
  260 PP=AA*DLXX
      IF(PP+1.68D02) 270,270,280
  270 P=0.0
      GO TO 660
  280 PP=DEXP(PP)
      P=SNGL(PP)
      GO TO 600
C
C        TEST FOR A OR B GREATER THAN 1000.0
C
  290 IF(A-1000.) 300,300,310
  300 IF(B-1000.) 330,330,320
  310 XX=2.D0*AA/XO
      XS=SNGL(XX)
      AA=2.D0*BB
      DF=SNGL(AA)
      CALL CDTR(XS,DF,P,DUMMY,IER)
      P=1.0-P
      GO TO 670
  320 XX=2.D0*BB*XO
      XS=SNGL(XX)
      AA=2.D0*AA
      DF=SNGL(AA)
      CALL CDTR(XS,DF,P,DUMMY,IER)
      GO TO 670
C
C        SELECT PARAMETERS FOR CONTINUED FRACTION COMPUTATION
C
  330 IF(X-.5) 340,340,380
  340 IF(AA-1.D0) 350,350,360
  350 RR=AA+1.D0
      GO TO 370
  360 RR=AA
  370 DD=DLXX/5.D0
      DD=DEXP(DD)
      DD=(RR-1.D0)-(RR+BB-1.D0)*XX*DD +2.D0
      IF(DD) 420,420,430
  380 IF(BB-1.D0) 390,390,400
  390 RR=BB+1.D0
      GO TO 410
  400 RR=BB
  410 DD=DL1X/5.D0
      DD=DEXP(DD)
      DD=(RR-1.D0)-(AA+RR-1.D0)*(1.D0-XX)*DD +2.D0
      IF(DD) 430,430,420
  420 ID=1
      FF=DL1X
      DL1X=DLXX
      DLXX=FF
      XO=1.D0/XO
      FF=AA
      AA=BB
      BB=FF
      G2=G1
C
C        TEST FOR A LESS THAN 1.0
C
  430 FF=0.D0
      IF(AA-1.D0) 440,440,470
  440 CALL DLGAM(AA+1.D0,G4,IOK)
      DD=AA*DLXX+BB*DL1X+G3-G2-G4
      IF(DD+1.68D02) 460,460,450
  450 FF=FF+DEXP(DD)
  460 AA=AA+1.D0
C
C        COMPUTE P USING CONTINUED FRACTION EXPANSION
C
  470 FN=AA+BB-1.D0
      RR=AA-1.D0
      II=80
      XI=DFLOAT(II)
      SS=((BB-XI)*(RR+XI))/((RR+2.D0*XI-1.D0)*(RR+2.D0*XI))
      SS=SS*XO
      DO 480 I=1,79
      II=80-I
      XI=DFLOAT(II)
      DD=(XI*(FN+XI))/((RR+2.D0*XI+1.D0)*(RR+2.D0*XI))
      DD=DD*XO
      CC=((BB-XI)*(RR+XI))/((RR+2.D0*XI-1.D0)*(RR+2.D0*XI))
      CC=CC*XO
      SS=CC/(1.D0+DD/(1.D0-SS))
  480 CONTINUE
      SS=1.D0/(1.D0-SS)
      IF(SS) 650,650,490
  490 CALL DLGAM(AA+BB,G1,IOK)
      CALL DLGAM(AA+1.D0,G4,IOK)
      CC=G1-G2-G4+AA*DLXX+(BB-1.D0)*DL1X
      PP=CC+DLOG(SS)
      IF(PP+1.68D02) 500,500,510
  500 PP=FF
      GO TO 520
  510 PP=DEXP(PP)+FF
  520 IF(ID) 540,540,530
  530 PP=1.D0-PP
  540 P=SNGL(PP)
C
C        SET ERROR INDICATOR
C
      IF(P) 550,570,570
  550 IF(ABS(P)-1.E-7) 560,560,650
  560 P=0.0
      GO TO 660
  570 IF(1.-P) 580,600,600
  580 IF(ABS(1.-P)-1.E-7) 590,590,650
  590 P=1.0
      GO TO 660
  600 IF(P-1.E-8) 610,610,620
  610 P=0.0
      GO TO 660
  620 IF((1.0-P)-1.E-8) 630,630,660
  630 P=1.0
      GO TO 660
  640 IER=-2
      D=-1.E75
      P=-1.E75
      GO TO 670
  650 IER=+2
      P= 1.E75
      GO TO 670
  660 IER=0
  670 RETURN
      END
```

Subroutine CDTR

This subroutine computes $P = P(x) = \text{Prob.}\ (X \le x)$, where X is a random variable following the χ^2 distribution with continuous parameter m. The value of x must be greater than or equal to zero and $0.5 \le m \le 2\ (10^5)$ for computation to take place. D, the ordinate of the χ^2 density at x, is also presented in the output.

For $x \ge 0$, P(x) may be written as:

$$P(x) = \int_0^x f(m,y)\,dy \qquad (1)$$

where $f(m,y) = y^{(m-2)/2}\ e^{-y/2} / (2^{m/2}\,\Gamma(\frac{m}{2}))$

and $D = f(m,x)$

To evaluate the integral, $\theta = \frac{m}{2} - [\frac{m}{2}]$ is first defined where $[\frac{m}{2}]$ denotes the largest integer less than or equal to $\frac{m}{2}$. θ is thus the fractional part of $\frac{m}{2}$.

Substituting this expression into the integral and performing the proper reductions, the following results are obtained.

If:	then:
$0 < m < 2$	$P(x) = T1 + T2$
$2 \le m < 4$	$P(x) = T1$
$m \ge 4$	$P(x) = T1 - 2T3$

where:

$$T1 = \int_0^x \frac{y^{\theta}\, e^{-y/2}\, dy}{2^{1+\theta}\,\Gamma(1+\theta)}$$

$$T2 = f(2+2\theta, x)$$

$$T3 = \sum_{i=2}^{[\frac{m}{2}]} f(2i+2\theta, x)$$

T2 and T3 may be evaluated directly using logs and antilogs.

If $\theta = 0$ ($\frac{m}{2}$ is an integer), T1 is easily evaluated as:

$$T1 = 1 - e^{-x/2}$$

If $\theta > 0$, T1 can be expanded in the following infinite series:

$$T1 = \frac{Z^{\theta}}{\Gamma(1+\theta)}\left(\frac{Z}{1+\theta} - \frac{Z^2}{2+\theta} + \frac{Z^3}{2!\,(3+\theta)} - \frac{Z^4}{3!\,(4+\theta)} \cdots\right) \qquad (2)$$

where $Z = \frac{x}{2}$

This series is used in the range $10^{-8} < x \le 10$, and not more than 30 terms are necessary to ensure convergence within error bounds of 10^{-9}.

For $x > 10$, 1-T1 is evaluated by the Euler-McLaurin formula up to third-derivative terms (see reference (2), equation 23.1.30). One finds:

$$1 - T1 = \int_0^N h(u)\,du \qquad (3)$$

where $h(u) = \frac{1}{\Gamma(1+\theta)} \left(\frac{2u}{Nx}\right)^{-(1+\theta)} u^{-1}\ e^{-Nx/2u}$

and $\int_0^N h(u)du = \sum_{u=o}^{N-1} h(u) + \frac{1}{2}\,h(N) - \frac{1}{12}\,h'(N)$

$+ \frac{1}{720}\,h'''(N)$ (Note: $h' = h'''$

$= 0$ at 0)

In order to achieve accuracy consistent with that obtained by the method of equation (2), N = 26 is used in equation (3).

If $0 \le x \le 10^{-8}$, the approximation $x = 0$ is made. P is set to 0; and D is set to 10^{75}, 0.5, or 0, corresponding to m less than 2, equal to 2, or greater than 2 respectively.

If $m > 1000$, Wilson and Hilferty's approximation is used. $(\chi^2/m)^{1/3}$ is approximately normally distributed with mean $1 - 2/9m$ and variance $2/9m$ (see reference (2), equation 26.4.14). If $m \le 1000$ and $x > 2000$, or $m > 1000$ and $x > 10^6$, P is set to 1.

Since T1 may have an error of about 10^{-9}, values of P(x) very near zero or one may be somewhat imprecise. To eliminate possible misinterpretation of results, if $0 \le P(x) \le 10^{-8}$ or $0 \le 1-P(x) \le 10^{-8}$, P(x) is set to 0 or 1 respectively.

The χ^2 distribution is a member of the gamma family of probability distributions. The general form for distributions of this class is:

$$P_G(x) = \int_0^x G(n, A, \psi; u)\, du$$

where $G(n, a, \psi; u) = (u-a)^{n-1} e^{-(u-a)/\psi} / (\psi^n \Gamma(n))$

This subroutine may, therefore, also be used to compute the probability integral from zero to x and the corresponding ordinate at x for any member of this gamma family by setting:

$$x = 2(u-a)/\psi \text{ and } m = 2n$$

Then P(x) will be the desired probability and $2f(m,x)/\psi$ will be the desired ordinate.

For reference see:

(1) R. E. Bargmann and S. P. Ghosh, Statistical Distribution Programs For a Computer Language. IBM Research Report RC-1094, 1963

(2) M. Abramowitz and I. A. Stegun, Handbook of Mathematical Functions. U. S. Department of Commerce, National Bureau of Standards Applied Mathematics Series, 1966.

```
C                                                                       CDTR  10
C     ...................................................................CDTR  20
C                                                                       CDTR  30
C        SUBROUTINE CDTR                                                CDTR  40
C                                                                       CDTR  50
C        PURPOSE                                                        CDTR  60
C           COMPUTES P(X) = PROBABILITY THAT THE RANDOM VARIABLE U,     CDTR  70
C           DISTRIBUTED ACCORDING TO THE CHI-SQUARE DISTRIBUTION WITH G CDTR  80
C           DEGREES OF FREEDOM, IS LESS THAN OR EQUAL TO X.  F(G,X), THECDTR  90
C           ORDINATE OF THE CHI-SQUARE DENSITY AT X, IS ALSO COMPUTED.  CDTR 100
C                                                                       CDTR 110
C        USAGE                                                          CDTR 120
C           CALL CDTR(X,G,P,D,IER)                                      CDTR 130
C                                                                       CDTR 140
C        DESCRIPTION OF PARAMETERS                                      CDTR 150
C           X   - INPUT SCALAR FOR WHICH P(X) IS COMPUTED.              CDTR 160
C           G   - NUMBER OF DEGREES OF FREEDOM OF THE CHI-SQUARE        CDTR 170
C                 DISTRIBUTION.  G IS A CONTINUOUS PARAMETER.           CDTR 180
C           P   - OUTPUT PROBABILITY.                                   CDTR 190
C           D   - OUTPUT DENSITY.                                       CDTR 200
C           IER - RESULTANT ERROR CODE WHERE                            CDTR 210
C                 IER= 0 --- NO ERROR                                   CDTR 220
C                 IER=-1 --- AN INPUT PARAMETER IS INVALID.  X IS LESS  CDTR 230
C                            THAN 0.0, OR G IS LESS THAN 0.5 OR GREATER CDTR 240
C                            THAN 2*10**(+5).  P AND D ARE SET TO -1.E75. CDTR 250
C                 IER=+1 --- INVALID OUTPUT.  P IS LESS THAN ZERO OR    CDTR 260
C                            GREATER THAN ONE, OR SERIES FOR T1 (SEE    CDTR 270
C                            MATHEMATICAL DESCRIPTION) HAS FAILED TO    CDTR 280
C                            CONVERGE.  P IS SET TO 1.E75.              CDTR 290
C                                                                       CDTR 300
C        REMARKS                                                        CDTR 310
C           SEE MATHEMATICAL DESCRIPTION.                               CDTR 320
C                                                                       CDTR 330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CDTR 340
C           DLGAM                                                       CDTR 350
C           NDTR                                                        CDTR 360
C                                                                       CDTR 370
C        METHOD                                                         CDTR 380
C           REFER TO R.E. BARGMANN AND S.P. GHOSH, STATISTICAL          CDTR 390
C           DISTRIBUTION PROGRAMS FOR A COMPUTER LANGUAGE,              CDTR 400
C           IBM RESEARCH REPORT RC-1094, 1963.                          CDTR 410
C                                                                       CDTR 420
C     ...................................................................CDTR 430
C                                                                       CDTR 440
      SUBROUTINE CDTR(X,G,P,D,IER)                                      CDTR 450
      DOUBLE PRECISION XX,DLXX,X2,DLX2,GG,G2,DLT3,THETA,THP1,           CDTR 460
     1GLG2,DD,T11,SER,CC,X1,FAC,TLOG,TERM,GTH,A2,A,B,C,DT2,DT3,THPI     CDTR 470
C                                                                       CDTR 480
C        TEST FOR VALID INPUT DATA                                      CDTR 490
C                                                                       CDTR 500
      IF(G-(.5-1.E-5)) 590,10,10                                        CDTR 510
   10 IF(G-2.E+5) 20,20,590                                             CDTR 520
   20 IF(X) 590,30,30                                                   CDTR 530
C                                                                       CDTR 540
C        TEST FOR X NEAR 0.0                                            CDTR 550
C                                                                       CDTR 560
   30 IF(X-1.E-8) 40,40,80                                              CDTR 570
   40 P=0.0                                                             CDTR 580
      IF(G-2.) 50,60,70                                                 CDTR 590
   50 D=1.E75                                                           CDTR 600
      GO TO 610                                                         CDTR 610
   60 D=0.5                                                             CDTR 620
      GO TO 610                                                         CDTR 630
   70 D=0.0                                                             CDTR 640
      GO TO 610                                                         CDTR 650
C                                                                       CDTR 660
C        TEST FOR X GREATER THAN 1.E+6                                  CDTR 670
C                                                                       CDTR 680
   80 IF(X-1.E+6) 100,100,90                                            CDTR 690
   90 D=0.0                                                             CDTR 700
      P=1.0                                                             CDTR 710
      GO TO 610                                                         CDTR 720
C                                                                       CDTR 730
C        SET PROGRAM PARAMETERS                                         CDTR 740
C                                                                       CDTR 750
  100 XX=DBLE(X)                                                        CDTR 760
      DLXX=DLOG(XX)                                                     CDTR 770
      X2=XX/2.D0                                                        CDTR 780
      DLX2=DLOG(X2)                                                     CDTR 790
      GG=DBLE(G)                                                        CDTR 800
      G2=GG/2.D0                                                        CDTR 810
C                                                                       CDTR 820
C        COMPUTE ORDINATE                                               CDTR 830
C                                                                       CDTR 840
      CALL DLGAM(G2,GLG2,IOK)                                           CDTR 850
      DD=(G2-1.D0)*DLXX-X2-G2*.6931471805599453 -GLG2                   CDTR 860
      IF(DD-1.68D02) 110,110,120                                        CDTR 870
  110 IF(DD+1.68D02) 130,130,140                                        CDTR 880
  120 D=1.E75                                                           CDTR 890
      GO TO 150                                                         CDTR 900
  130 D=0.0                                                             CDTR 910
      GO TO 150                                                         CDTR 920
  140 DD=DEXP(DD)                                                       CDTR 930
      D=SNGL(DD)                                                        CDTR 940
C                                                                       CDTR 950
C        TEST FOR G GREATER THAN 1000.0                                 CDTR 960
C        TEST FOR X GREATER THAN 2000.0                                 CDTR 970
C                                                                       CDTR 980
  150 IF(G-1000.) 160,160,180                                           CDTR 990
  160 IF(X-2000.) 190,190,170                                           CDTR1000
  170 P=1.0                                                             CDTR1010
      GO TO 610                                                         CDTR1020
  180 A=DLOG(XX/GG)/3.D0                                                CDTR1030
      A=DEXP(A)                                                         CDTR1040
      B=2.D0/(9.D0*GG)                                                  CDTR1050
      C=(A-1.D0+B)/DSQRT(B)                                             CDTR1060
      SC=SNGL(C)                                                        CDTR1070
      CALL NDTR(SC,P,DUMMY)                                             CDTR1080
      GO TO 490                                                         CDTR1090
C                                                                       CDTR1100
C        COMPUTE THETA                                                  CDTR1110
C                                                                       CDTR1120
  190 K= IDINT(G2)                                                      CDTR1130
      THETA=G2-DFLOAT(K)                                                CDTR1140
      IF(THETA-1.D-8) 200,200,210                                       CDTR1150
  200 THETA=0.D0                                                        CDTR1160
  210 THP1=THETA+1.D0                                                   CDTR1170
C                                                                       CDTR1180
C        SELECT METHOD OF COMPUTING T1                                  CDTR1190
C                                                                       CDTR1200
      IF(THETA) 230,230,220                                             CDTR1210
  220 IF(XX-10.D0) 260,260,320                                          CDTR1220
C                                                                       CDTR1230
C        COMPUTE T1 FOR THETA EQUALS 0.0                                CDTR1240
C                                                                       CDTR1250
  230 IF(X2-1.68D02) 250,240,240                                        CDTR1260
  240 T1=1.0                                                            CDTR1270
      GO TO 400                                                         CDTR1280
  250 T11=1.D0-DEXP(-X2)                                                CDTR1290
      T1=SNGL(T11)                                                      CDTR1300
      GO TO 400                                                         CDTR1310
C                                                                       CDTR1320
C        COMPUTE T1 FOR THETA GREATER THAN 0.0 AND                      CDTR1330
C        X LESS THAN OR EQUAL TO 10.0                                   CDTR1340
C                                                                       CDTR1350
  260 SER=X2*(1.D0/THP1 -X2/(THP1+1.D0))                                CDTR1360
      J=+1                                                              CDTR1370
      CC=DFLOAT(J)                                                      CDTR1380
      DO 270 IT1=3,30                                                   CDTR1390
      XI=DFLOAT(IT1)                                                    CDTR1400
      CALL DLGAM(XI,FAC,IOK)                                            CDTR1410
      TLOG= XI*DLX2-FAC-DLOG(XI+THETA)                                  CDTR1420
      TERM=DEXP(TLOG)                                                   CDTR1430
      TERM=DSIGN(TERM,CC)                                               CDTR1440
      SER=SER+TERM                                                      CDTR1450
      CC=-CC                                                            CDTR1460
      IF(DABS(TERM)-1.D-9) 280,270,270                                  CDTR1470
  270 CONTINUE                                                          CDTR1480
      GO TO 600                                                         CDTR1490
  280 IF(SER) 600,600,290                                               CDTR1500
  290 CALL DLGAM(THP1,GTH,IOK)                                          CDTR1510
      TLOG=THETA*DLX2+DLOG(SER)-GTH                                     CDTR1520
      IF(TLOG+1.68D02) 300,300,310                                      CDTR1530
  300 T1=0.0                                                            CDTR1540
      GO TO 400                                                         CDTR1550
  310 T11=DEXP(TLOG)                                                    CDTR1560
      T1=SNGL(T11)                                                      CDTR1570
      GO TO 400                                                         CDTR1580
C                                                                       CDTR1590
C        COMPUTE T1 FOR THETA GREATER THAN 0.0 AND                      CDTR1600
C        X GREATER THAN 10.0 AND LESS THAN 2000.0                       CDTR1610
C                                                                       CDTR1620
  320 A2=0.D0                                                           CDTR1630
      DO 340 I=1,25                                                     CDTR1640
      XI=DFLOAT(I)                                                      CDTR1650
      CALL DLGAM(THP1,GTH,IOK)                                          CDTR1660
      T11=-(13.D0*XX)/XI +THP1*DLOG(13.D0*XX/XI) -GTH-DLOG(XI)          CDTR1670
      IF(T11+1.68D02) 340,340,330                                       CDTR1680
  330 T11=DEXP(T11)                                                     CDTR1690
      A2=A2+T11                                                         CDTR1700
  340 CONTINUE                                                          CDTR1710
      A=1.01282051+THETA/156.D0-XX/312.D0                               CDTR1720
      B=DABS(A)                                                         CDTR1730
      C= -X2+THP1*DLX2+DLOG(B)-GTH-3.951243718581427                    CDTR1740
      IF(C+1.68D02) 370,370,350                                         CDTR1750
  350 IF (A) 360,370,380                                                CDTR1760
  360 C=-DEXP(C)                                                        CDTR1770
      GO TO 390                                                         CDTR1780
  370 C=0.D0                                                            CDTR1790
      GO TO 390                                                         CDTR1800
  380 C=DEXP(C)                                                         CDTR1810
  390 C=A2+C                                                            CDTR1820
      T11=1.D0-C                                                        CDTR1830
      T1=SNGL(T11)                                                      CDTR1840
C                                                                       CDTR1850
C        SELECT PROPER EXPRESSION FOR P                                 CDTR1860
C                                                                       CDTR1870
  400 IF(G-2.) 420,410,410                                              CDTR1880
  410 IF(G-4.) 450,460,460                                              CDTR1890
C                                                                       CDTR1900
C        COMPUTE P FOR G GREATER THAN ZERO AND LESS THAN 2.0            CDTR1910
C                                                                       CDTR1920
  420 CALL DLGAM(THP1,GTH,IOK)                                          CDTR1930
      DT2=THETA*DLXX-X2-THP1*.6931471805599453 -GTH                     CDTR1940
```

```
      IF(DT2+1.68D02) 430,430,440
  430 P=T1
      GO TO 490
  440 DT2=DEXP(DT2)
      T2=SNGL(DT2)
      P=T1+T2+T2
      GO TO 490
C
C        COMPUTE P FOR G GREATER THAN OR EQUAL TO 2.0
C        AND LESS THAN 4.0
C
  450 P=T1
      GO TO 490
C
C        COMPUTE P FOR G GREATER THAN OR EQUAL TO 4.0
C        AND LESS THAN OR EQUAL TO 1000.0
C
  460 DT3=0.D0
      DO 480 I3=2,K
      THPI=DFLOAT(I3)+THETA
      CALL DLGAM(THPI,GTH,IOK)
      DLT3=THPI*DLX2-DLXX-X2-GTH
      IF(DLT3+1.68D02) 480,480,470
  470 DT3=DT3+DEXP(DLT3)
  480 CONTINUE
      T3=SNGL(DT3)
      P=T1-T3-T3
C
C        SET ERROR INDICATOR
C
  490 IF(P) 500,520,520
  500 IF(ABS(P)-1.E-7) 510,510,600
  510 P=0.0
      GO TO 610
  520 IF(1.-P) 530,550,550
  530 IF(ABS(1.-P)-1.E-7) 540,540,600
  540 P=1.0
      GO TO 610
  550 IF(P-1.E-8) 560,560,570
  560 P=0.0
      GO TO 610
  570 IF((1.0-P)-1.E-8) 580,580,610
  580 P=1.0
      GO TO 610
  590 IER=-1
      D=-1.E75
      P=-1.E75
      GO TO 620
  600 IER=+1
      P= 1.E75
      GO TO 620
  610 IER=0
  620 RETURN
      END
```

Subroutine NDTRI

This subroutine computes $x = p^{-1}(y)$ such that $y = P(x) = \text{Prob}\ (X \le x)$ where X is a random variable distributed normally with mean zero and variance one.
That is, given P(x), the following is solved for x:

$$P(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp(-u^2/2)du$$

The following approximation† is used:

$$x = w - \sum_{i=0}^{2} a_i w^i / \sum_{i=0}^{3} b_i w^i \qquad (1)$$

where:

$$w = \sqrt{\ln(1/p^2)}\ (0 < p \le .5) \qquad (2)$$

$a_0 = 2.515517$

$a_1 = 0.802853$

$a_2 = 0.010328$

$b_1 = 1.432788$

$b_2 = 0.189269$

$b_3 = 0.001308$

If P(x) is greater than 0.5, 1 - P(x) is used as p in (2) above, and the result of (1), x, is negated.
The maximum error is 0.00045; f(x) is also calculated.

---

†C. Hastings, Approximations for Digital Computers. Princeton University Press, Princeton, N.J., 1955
M. Abramowitz and I. A. Stegun, Handbook of Mathematical Functions. Dover Publications, Inc., N.Y., equation 26.2.23.

```
                                                                        NTRI  10
*************************************************************************NTRI  20
                                                                        NTRI  30
      SUBROUTINE NCTRI                                                  NTRI  40
                                                                        NTRI  50
      PURPOSE                                                           NTRI  60
        COMPUTES X = P**(-1)(Y), THE ARGUMENT X SUCH THAT Y= P(X) =     NTRI  70
        THE PROBABILITY THAT THE RANDOM VARIABLE U, DISTRIBUTED         NTRI  80
        NORMALLY(0,1), IS LESS THAN OR EQUAL TO X.  F(X), THE           NTRI  90
        ORDINATE OF THE NORMAL DENSITY, AT X, IS ALSO COMPUTED.         NTRI 100
                                                                        NTRI 110
      USAGE                                                             NTRI 120
        CALL NCTRI(P,X,C,IER)                                           NTRI 130
                                                                        NTRI 140
      DESCRIPTION OF PARAMETERS                                         NTRI 150
        P   - INPUT PROBABILITY.                                        NTRI 160
        X   - OUTPUT ARGUMENT SUCH THAT P = Y = THE PROBABILITY THAT    NTRI 170
                U, THE RANDOM VARIABLE, IS LESS THAN OR EQUAL TO X.     NTRI 180
        C   - OUTPUT DENSITY, F(X).                                     NTRI 190
        IER - OUTPUT ERROR CODE                                         NTRI 200
              = -1 IF P IS NOT IN THE INTERVAL (0,1), INCLUSIVE.        NTRI 210
                X=C=.99999E+74 IN THIS CASE                             NTRI 211
              = 0 IF THERE IS NO ERROR.  SEE REMARKS, BELOW.            NTRI 220
                                                                        NTRI 230
      REMARKS                                                           NTRI 240
        MAXIMUM ERROR IS 0.00045.                                       NTRI 250
        IF P = 0, X IS SET TO -(10)**74.  C IS SET TO 0.                NTRI 260
        IF P = 1, X IS SET TO  (10)**74.  C IS SET TO 0.                NTRI 270
                                                                        NTRI 280
      SUBROUTINES AND SUBPROGRAMS REQUIRED                              NTRI 290
        NONE                                                            NTRI 300
                                                                        NTRI 310
      METHOD                                                            NTRI 320
        BASED ON APPROXIMATIONS IN C. HASTINGS, APPROXIMATIONS FOR      NTRI 330
        DIGITAL COMPUTERS, PRINCETON UNIV. PRESS, PRINCETON, N.J.,      NTRI 340
        1955.  SEE EQUATION 26.2.23, HANDBOOK OF MATHEMATICAL           NTRI 350
        FUNCTIONS, ABRAMOWITZ AND STEGUN, DOVER PUBLICATIONS, INC.,     NTRI 360
        NEW YORK.                                                       NTRI 370
                                                                        NTRI 380
*************************************************************************NTRI 390
                                                                        NTRI 400
      SUBROUTINE NCTRI(P,X,C,IE)                                        NTRI 410
                                                                        NTRI 420
      IE=0                                                              NTRI 430
      X=.99999E+74                                                      NTRI 431
      C=X                                                               NTRI 432
      IF(P)1,4,2                                                        NTRI 440
    1 IE=-1                                                             NTRI 450
      GO TO 12                                                          NTRI 460
    2 IF (P-1.0)7,5,1                                                   NTRI 470
    4 X=-.99999E+74                                                     NTRI 480
    5 C=0.0                                                             NTRI 490
      GO TO 12                                                          NTRI 500
                                                                        NTRI 510
                                                                        NTRI 520
    7 C=P                                                               NTRI 530
      IF(C-0.5)9,9,8                                                    NTRI 540
    8 C=1.0-C                                                           NTRI 550
    9 T2=ALOG(1.0/(C*C))                                                NTRI 560
      T=SQRT(T2)                                                        NTRI 570
      X=T-(2.515517+0.802853*T+0.010328*T2)/(1.0+1.432788*T+0.189269*T2 NTRI 580
     1  +0.001308*T*T2)                                                 NTRI 590
      IF(P-0.5)10,10,11                                                 NTRI 600
   10 X=-X                                                              NTRI 610
   11 C=0.39894228*EXP(-X*X/2.0)                                        NTRI 620
   12 RETURN                                                            NTRI 630
      END                                                               NTRI 640
```

Elementary Statistics and Miscellany

Subroutine MOMEN

This subroutine computes four moments for grouped data $F_1, F_2, \ldots, F_n$ on equal class intervals. The number of class intervals is computed as follows:

$$n = (UBO_3 - UBO_1)/UBO_2 \qquad (1)$$

where UBO_1 = given lower bound

UBO_2 = given class interval

UBO_3 = given upper bound

and the total frequency as follows:

$$T = \sum_{i=1}^{n} F_i \qquad (2)$$

where F_i = frequency count in i^{th} interval.

Then, the following are computed.

First moment (mean):

$$ANS_1 = \frac{\sum_{i=1}^{n} F_i \left[UBO_1 + (i-0.5)\, UBO_2 \right]}{T} \qquad (3)$$

j^{th} moment:

$$ANS_j = \frac{\sum_{i=1}^{n} F_i \left[UBO_1 + (i-0.5)\, UBO_2 - ANS_1 \right]^j}{T} \qquad (4)$$

$j = 2, 3, 4$

These moments are biased and not corrected for grouping.

```
C                                                                       MOME  10
C     ...................................................................MOME  20
C                                                                       MOME  30
C        SUBROUTINE MOMEN                                               MOME  40
C                                                                       MOME  50
C        PURPOSE                                                        MOME  60
C           TO FIND THE THE FIRST FOUR MOMENTS FOR GROUPED DATA ON      MOME  70
C           EQUAL CLASS INTERVALS.                                      MOME  80
C                                                                       MOME  90
C        USAGE                                                          MOME 100
C           CALL MOMEN (F,LBO,NOP,ANS)                                  MOME 110
C                                                                       MOME 120
C        DESCRIPTION OF PARAMETERS                                      MOME 130
C           F   - GROUPED DATA (FREQUENCIES).  GIVEN AS A VECTOR OF     MOME 140
C                 LENGTH (LBO(3)-UBO(1))/UBO(2)                         MOME 150
C           LBO - 3 CELL VECTOR. LBO(1) IS LOWER BOUND AND LBO(3) UPPER MOME 160
C                 BOUND ON DATA.  UBO(2) IS CLASS INTERVAL.  NOTE THAT  MOME 170
C                 UBO(3) MUST BE GREATER THAN UBO(1).                   MOME 180
C           NOP - OPTION PARAMETER.  IF NOP = 1, ANS(1) = MEAN.  IF     MOME 190
C                 NOP = 2, ANS(2) = SECOND MOMENT.  IF NOP = 3, ANS(3) =MOME 200
C                 THIRD MOMENT.  IF NOP = 4, ANS(4) = FOURTH MOMENT.    MOME 210
C                 IF NOP = 5, ALL FOUR MOMENTS ARE FILLED IN.           MOME 220
C           ANS - OUTPUT VECTOR OF LENGTH 4 INTO WHICH MOMENTS ARE PUT. MOME 230
C                                                                       MOME 240
C        REMARKS                                                        MOME 250
C           NOTE THAT THE FIRST MOMENT IS NOT CENTRAL BUT THE VALUE OF  MOME 260
C           THE MEAN ITSELF.  THE MEAN IS ALWAYS CALCULATED.  MOMENTS   MOME 270
C           ARE BIASED AND NOT CORRECTED FOR GROUPING.                  MOME 280
C                                                                       MOME 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MOME 300
C           NONE                                                        MOME 310
C                                                                       MOME 320
C        METHOD                                                         MOME 330
C           REFER TO M. G. KENDALL, 'THE ADVANCED THEORY OF STATISTICS',MOME 340
C           V.1, HAFNER PUBLISHING COMPANY, 1958, CHAPTER 3.            MOME 350
C                                                                       MOME 360
C     ...................................................................MOME 370
C                                                                       MOME 380
      SUBROUTINE MOMEN (F,LBO,NOP,ANS)                                  MOME 390
      DIMENSION F(1),LBO(1),ANS(1)                                      MOME 400
C                                                                       MOME 410
      DO 100 I=1,4                                                      MOME 420
  100 ANS(I)=0.0                                                        MOME 430
C                                                                       MOME 440
C     CALCULATE THE NUMBER OF CLASS INTERVALS                           MOME 450
C                                                                       MOME 460
      N=(LBO(3)-LBO(1))/LBO(2)+0.5                                      MOME 470
C                                                                       MOME 480
C     CALCULATE TOTAL FREQUENCY                                         MOME 490
C                                                                       MOME 500
      T=0.0                                                             MOME 510
      DO 110 I=1,N                                                      MOME 520
  110 T=T+F(I)                                                          MOME 530
C                                                                       MOME 540
      IF(NOP-5) 130, 120, 115                                           MOME 550
  115 NOP=5                                                             MOME 560
  120 JUMP=1                                                            MOME 570
      GO TO 150                                                         MOME 580
  130 JUMP=2                                                            MOME 590
C                                                                       MOME 600
C        FIRST MOMENT                                                   MOME 610
C                                                                       MOME 620
  150 DO 160 I=1,N                                                      MOME 630
      FI=I                                                              MOME 640
  160 ANS(1)=ANS(1)+F(I)*(LBO(1)+(FI-0.5)*UBO(2))                       MOME 650
      ANS(1)=ANS(1)/T                                                   MOME 660
C                                                                       MOME 670
      GO TO (250,200,250,300,200), NOP                                  MOME 680
C                                                                       MOME 690
C        SECOND MOMENT                                                  MOME 700
C                                                                       MOME 710
  200 DO 210 I=1,N                                                      MOME 720
      FI=I                                                              MOME 730
  210 ANS(2)=ANS(2)+F(I)*(LBO(1)+(FI-0.5)*UBO(2)-ANS(1))**2             MOME 740
      ANS(2)=ANS(2)/T                                                   MOME 750
      GO TO (250,350), JUMP                                             MOME 760
C                                                                       MOME 770
C        THIRD MOMENT                                                   MOME 780
C                                                                       MOME 790
  250 DO 260 I=1,N                                                      MOME 800
      FI=I                                                              MOME 810
  260 ANS(3)=ANS(3)+F(I)*(LBO(1)+(FI-0.5)*UBO(2)-ANS(1))**3             MOME 820
      ANS(3)=ANS(3)/T                                                   MOME 830
      GO TO (300,350), JUMP                                             MOME 840
C                                                                       MOME 850
C        FOURTH MOMENT                                                  MOME 860
C                                                                       MOME 870
  300 DO 310 I=1,N                                                      MOME 880
      FI=I                                                              MOME 890
  310 ANS(4)=ANS(4)+F(I)*(LBO(1)+(FI-0.5)*UBO(2)-ANS(1))**4             MOME 900
      ANS(4)=ANS(4)/T                                                   MOME 910
  350 RETURN                                                            MOME 920
      END                                                               MOME 930
```

Subroutine TTEST

This subroutine computes certain t-statistics on the means of populations under various hypotheses.

The sample means of $A_1, A_2, \ldots, A_{NA}$, and $B_1, B_2, \ldots, B_{NB}$ are normally found by the following formulas:

$$\overline{A} = \frac{\sum_{i=1}^{NA} A_i}{NA}; \qquad \overline{B} = \frac{\sum_{i=1}^{NB} B_i}{NB} \tag{1}$$

and the corresponding sample variances by:

$$SA^2 = \frac{\sum_{i=1}^{NA} (A_i - \overline{A})^2}{NA - 1}; \qquad SB^2 = \frac{\sum_{i=1}^{NB} (B_i - \overline{B})^2}{NB - 1} \tag{2}$$

The quantities μ and σ^2 stand respectively for population mean and variance in the following hypotheses.

Hypothesis: $\mu_B = A$; A = a given value (Option 1)

Let $\overline{B}$ = estimate of μ_B and set NA = 1. (A is stored in location A_1).

The subroutine computes:

$$ANS = \frac{\overline{B} - A}{SB} \cdot \sqrt{NB} \quad \text{(t-statistics)} \tag{3}$$

$$NDF = NB - 1 \quad \text{(degrees of freedom)} \tag{4}$$

Hypothesis: $\mu_A = \mu_B \left(\sigma_A^2 = \sigma_B^2\right)$ (Option 2)

The subroutine computes:

$$ANS = \frac{\overline{B} - \overline{A}}{S} \cdot \frac{1}{\sqrt{\frac{1}{NA} + \frac{1}{NB}}} \quad \text{(t-statistics)} \tag{5}$$

$$NDF = NA + NB - 2 \quad \text{(degrees of freedom)} \tag{6}$$

$$\text{where } S = \sqrt{\frac{(NA-1)\,SA^2 + (NB-1)\,SB^2}{NA + NB - 2}} \tag{7}$$

Hypothesis: $\mu_A = \mu_B \left(\sigma_A^2 \neq \sigma_B^2\right)$ (Option 3)

The subroutine computes:

$$ANS = \frac{\overline{B} - \overline{A}}{\sqrt{\frac{SA^2}{NA} + \frac{SB^2}{NB}}} \quad \text{(t-statistics)} \tag{8}$$

$$NDF = \frac{\left(\frac{SA^2}{NA} + \frac{SB^2}{NB}\right)^2}{\left(\frac{SA^2}{NA}\right)^2 \Big/ (NA+1) + \left(\frac{SB^2}{NB}\right)^2 \Big/ (NB+1)} - 2 \tag{9}$$

(degrees of freedom)

Note: The program returns a rounded NDF, not a truncated NDF.

Hypothesis: $\mu_A = \mu_B \left(\text{no assumption on } \sigma^2\right)$ (Option 4)

The subroutine computes:

$$ANS = \frac{\overline{D}}{SD} \cdot \sqrt{NB} \quad \text{(t-statistics)} \tag{10}$$

$$NDF = NB - 1 \quad \text{(degrees of freedom)} \tag{11}$$

$$\text{where } \overline{D} = \overline{B} - \overline{A} \tag{12}$$

$$SD = \sqrt{\frac{\sum_{i=1}^{NB} (B_i - A_i - \overline{D})^2}{NB - 1}} \tag{13}$$

NA = NB

```
C                                                                       TTES  10
C     ...................................................................TTES  20
C                                                                       TTES  30
C        SUBROUTINE TTEST                                               TTES  40
C                                                                       TTES  50
C        PURPOSE                                                        TTES  60
C           TO FIND CERTAIN T-STATISTICS ON THE MEANS OF POPULATIONS.   TTES  70
C                                                                       TTES  80
C        USAGE                                                          TTES  90
C           CALL TTEST (A,NA,B,NB,NOP,NDF,ANS)                          TTES 100
C                                                                       TTES 110
C        DESCRIPTION OF PARAMETERS                                      TTES 120
C           A   - INPUT VECTOR OF LENGTH NA CONTAINING DATA.            TTES 130
C           NA  - NUMBER OF OBSERVATIONS IN A.                          TTES 140
C           B   - INPUT VECTOR OF LENGTH NB CONTAINING DATA.            TTES 150
C           NB  - NUMBER OF OBSERVATIONS IN B.                          TTES 160
C           NOP - OPTIONS FOR VARIOUS HYPOTHESES..                      TTES 170
C                 NOP=1--- THAT POPULATION MEAN OF B = GIVEN VALUE A.   TTES 180
C                          (SET NA=1)                                   TTES 190
C                 NOP=2--- THAT POPULATION MEAN OF B = POPULATION MEAN  TTES 200
C                          OF A, GIVEN THAT THE VARIANCE OF B = THE     TTES 210
C                          VARIANCE OF A.                               TTES 220
C                 NOP=3--- THAT POPULATION MEAN OF B = POPULATION MEAN  TTES 230
C                          OF A, GIVEN THAT THE VARIANCE OF B IS NOT    TTES 240
C                          EQUAL TO THE VARIANCE OF A.                  TTES 250
C                 NOP=4--- THAT POPULATION MEAN OF B = POPULATION MEAN  TTES 260
C                          OF A, GIVEN NO INFORMATION ABOUT VARIANCES OFTTES 270
C                          A AND B.  (SET NA=NB)                        TTES 280
C           NDF - OUTPUT VARIABLE CONTAINING DEGREES OF FREEDOM ASSOCI- TTES 290
C                 ATED WITH T-STATISTIC CALCULATED.                     TTES 300
C           ANS - T-STATISTIC FOR GIVEN HYPOTHESIS.                     TTES 310
C                                                                       TTES 320
C        REMARKS                                                        TTES 330
C           NA AND NB MUST BE GREATER THAN 1, EXCEPT THAT NA=1 IN       TTES 340
C           OPTION 1. NA AND NB MUST BE THE SAME IN OPTION 4.           TTES 350
C           IF NOP IS OTHER THAN 1, 2, 3 OR 4, DEGREES OF FREEDOM AND   TTES 360
C           T-STATISTIC WILL NOT BE CALCULATED.  NDF AND ANS WILL BE    TTES 370
C           SET TO ZERO.                                                TTES 380
C                                                                       TTES 390
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TTES 400
C           NONE                                                        TTES 410
C                                                                       TTES 420
C        METHOD                                                         TTES 430
C           REFER TO DIXIE, BERNARD, 'STATISTICS IN RESEARCH', IOWA     TTES 440
C           STATE COLLEGE PRESS, 1954, CHAPTER 5.                       TTES 450
C                                                                       TTES 460
C     ...................................................................TTES 470
C                                                                       TTES 480
      SUBROUTINE TTEST (A,NA,B,NB,NOP,NDF,ANS)                          TTES 490
      DIMENSION A(1),B(1)                                               TTES 500
C                                                                       TTES 510
C     INITIALIZATION                                                    TTES 520
C                                                                       TTES 530
      NDF=0                                                             TTES 540
      ANS=0.0                                                           TTES 550
C                                                                       TTES 560
C     CALCULATE THE MEAN OF A                                           TTES 570
C                                                                       TTES 580
      AMEAN=0.0                                                         TTES 590
      DO 110 I=1,NA                                                     TTES 600
```

```
  110 AMEAN=AMEAN+A(I)                                              TTES 610
      FNA=NA                                                        TTES 620
      AMEAN=AMEAN/FNA                                               TTES 630
C                                                                   TTES 640
C     CALCULATE THE MEAN OF B                                       TTES 650
C                                                                   TTES 660
  115 BMEAN=0.0                                                     TTES 670
      DO 120 I=1,NB                                                 TTES 680
  120 BMEAN=BMEAN+B(I)                                              TTES 690
      FNB=NB                                                        TTES 700
      BMEAN=BMEAN/FNB                                               TTES 710
C                                                                   TTES 720
      IF(NOP-4) 122, 180, 200                                       TTES 730
  122 IF(NOP-1) 200, 135, 125                                       TTES 740
C                                                                   TTES 750
C     CALCULATE THE VARIANCE OF A                                   TTES 760
C                                                                   TTES 770
  125 SA2=0.0                                                       TTES 780
      DO 130 I=1,NA                                                 TTES 790
  130 SA2=SA2+(A(I)-AMEAN)**2                                       TTES 800
      SA2=SA2/(FNA-1.0)                                             TTES 810
C                                                                   TTES 820
C     CALCULATE THE VARIANCE OF B                                   TTES 830
C                                                                   TTES 840
  135 SB2=0.0                                                       TTES 850
      DO 140 I=1,NB                                                 TTES 860
  140 SB2=SB2+(B(I)-BMEAN)**2                                       TTES 870
      SB2=SB2/(FNB-1.0)                                             TTES 880
C                                                                   TTES 890
      GO TO (150,160,170), NOP                                      TTES 900
C                                                                   TTES 910
C        OPTION 1                                                   TTES 920
C                                                                   TTES 930
  150 ANS=((BMEAN-AMEAN)/SQRT(SB2))*SQRT(FNB)                       TTES 940
      NDF=NB-1                                                      TTES 950
      GO TO 200                                                     TTES 960
C                                                                   TTES 970
C        OPTION 2                                                   TTES 980
C                                                                   TTES 990
  160 NDF=NA+NB-2                                                   TTES1000
      FNDF=NDF                                                      TTES1010
      S=SQRT(((FNA-1.0)*SA2+(FNB-1.0)*SB2)/FNDF)                    TTES1020
      ANS=((BMEAN-AMEAN)/S)*(1.0/SQRT(1.0/FNA+1.0/FNB))             TTES1030
      GO TO 200                                                     TTES1040
C                                                                   TTES1050
C        OPTION 3                                                   TTES1060
C                                                                   TTES1070
  170 ANS=(BMEAN-AMEAN)/SQRT(SA2/FNA+SB2/FNB)                       TTES1080
      A1=(SA2/FNA+SB2/FNB)**2                                       TTES1090
      A2=(SA2/FNA)**2/(FNA+1.0)+(SB2/FNB)**2/(FNB+1.0)              TTES1100
      NDF=A1/A2-2.0+0.5                                             TTES1110
      GO TO 200                                                     TTES1120
C                                                                   TTES1130
C        OPTION 4                                                   TTES1140
C                                                                   TTES1150
  180 SD=0.0                                                        TTES1160
      D=BMEAN-AMEAN                                                 TTES1170
      DO 190 I=1,NB                                                 TTES1180
  190 SD=SD+(B(I)-A(I)-D)**2                                        TTES1190
      SD=SQRT(SD/(FNB-1.0))                                         TTES1200
      ANS=(D/SD)*SQRT(FNB)                                          TTES1210
      NDF=NB-1                                                      TTES1220
C                                                                   TTES1230
  200 RETURN                                                        TTES1240
      END                                                           TTES1250
```

Subroutine BISER

This subroutine computes a biserial coefficient of correlation between two continuous variables when one has been artificially dichotomized. The computational steps are as follows:

1. Compute the mean of the continuously measured variable, A:

$$\bar{A} = \frac{\sum_{i=1}^{N} A_i}{N} \tag{1}$$

where N = number of observations in the sample.

2. Compute the standard deviation of the continuously measured variable, A:

$$s = \sqrt{\frac{\sum_{i=1}^{N} A_i^2 - \frac{\left(\sum_{i=1}^{N} A_i\right)^2}{N}}{N-1}} \tag{2}$$

3. Obtain the following information for the dichotomous variable, B:

p = proportion of the cases in the higher group

q = proportion of the cases in the lower group

y = ordinate of the normal distribution curve at the point of division between p and q proportions

4. Obtain $\bar{A}_p$ = mean of A values whose corresponding B values are in the higher group.

5. Then, compute the biserial coefficient of correlation, r_b, by the use of the following formula:

$$r_b = \frac{\bar{A}_p - \bar{A}}{s} \cdot \frac{p}{y} \tag{3}$$

6. The standard error of r_b is estimated by the formula:

$$s_{r_b} = \frac{\sqrt{\frac{pq}{y} - r_b^2}}{\sqrt{N}} \tag{4}$$

For reference see:

(1) J. P. Guilford, Fundamental Statistics in Psychology and Education. McGraw-Hill, New York, 1956, chapter 13.

(2) J. W. Dunlop, "Note on Computation of Biserial Correlations in Item Evaluation", Psychometrika, 1936, I, pp. 51-60.

```
C
C     ..................................................................
C
C        SUBROUTINE BISER
C
C        PURPOSE
C           TO COMPUTE THE BISERIAL CORRELATION COEFFICIENT BETWEEN TWO
C           CONTINUOUS VARIABLES WHEN ONE OF THEM HAS BEEN ARTIFICIALLY
C           DICHOTOMIZED.
C
C        USAGE
C           CALL BISER (N,A,B,HI,ANS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           N   - NUMBER OF OBSERVATIONS
C           A   - INPUT VECTOR OF LENGTH N CONTAINING THE CONTINUOUS
C                 VARIABLE
C           B   - INPUT VECTOR OF LENGTH N CONTAINING THE DICHOTOMIZED
C                 VARIABLE
C           HI  - INPUT - NUMERICAL CODE TO INDICATE THE HIGHER CATEGORY
C                 OF THE DICHOTOMIZED VARIABLE.  ANY VALUE IN VECTOR B
C                 EQUAL TO OR GREATER THAN HI WILL BE CLASSIFIED INTO
C                 THE HIGHER CATEGORY.
C           ANS - OUTPUT VECTOR OF LENGTH 8 CONTAINING THE FOLLOWING
C                 ANS(1) - MEAN OF VARIABLE A
C                 ANS(2) - STANDARD DEVIATION OF VARIABLE A
C                 ANS(3) - PROPORTION OF THE CASES IN THE HIGHER
C                          CATEGORY OF VARIABLE B
C                 ANS(4) - PROPORTION OF THE CASES IN THE LOWER
C                          CATEGORY OF VARIABLE B
C                 ANS(5) - MEAN OF VARIABLE A FOR THOSE CASES FALLING
C                          INTO THE HIGHER CATEGORY OF VARIABLE B
C                 ANS(6) - MEAN OF VARIABLE A FOR THOSE CASES FALLING
C                          INTO THE LOWER CATEGORY OF VARIABLE B
C                 ANS(7) - BISERIAL CORRELATION COEFFICIENT
C                 ANS(8) - STANDARD ERROR OF BISERIAL CORRELATION
C                          COEFFICIENT
C           IER -  1, IF NO CASES ARE IN THE LOWER CATEGORY OF VARIABLE
C                  B.
C                 -1, IF ALL CASES ARE IN THE LOWER CATEGORY OF
C                  VARIABLE B.
C                  0, OTHERWISE.
C                  IF IER IS NON-ZERO, ANS(I)=10**75,I=5,....,8.
C
C        REMARKS
C           THE VALUES OF THE DICHOTOMIZED VARIABLE, B, MUST BE IN
C           NUMERIC FORM.  THEY CANNOR BE SPECIFIED BY MEANS OF
C           ALPHABETIC OR SPECIAL CHARACTERS.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NDTRI
C
C        METHOD
C           REFER TO P. HORST, 'PSYCHOLOGICAL MEASUREMENT AND
C           PREDICTION', P.95-96 (WADSWORTH, 1966).
C
C     ..................................................................
C
      SUBROUTINE BISER (N,A,B,HI,ANS,IER)
C
      DIMENSION A(1),B(1),ANS(1)
C
C        COMPUTE MEAN AND STANDARD DEVIATION OF VARIABLE A
C
      IER=0
      SUM=0.0
      SUM2=0.0
      DO 10 I=1,N
      SUM=SUM+A(I)
   10 SUM2=SUM2+A(I)*A(I)
      FN=N
      ANS(1)=SUM/FN
      ANS(2)=(SUM2-ANS(1)*SUM)/(FN-1.0)
      ANS(2)= SQRT(ANS(2))
C
C        FIND PROPORTIONS OF CASES IN THE HIGHER AND LOWER CATEGORIES
C
      P=0.0
      SUM=0.0
      SUM2=0.0
      DO 30 I=1,N
      IF(B(I)-HI) 20, 25, 25
   20 SUM2=SUM2+A(I)
      GO TO 30
   25 P=P+1.0
      SUM=SUM+A(I)
   30 CONTINUE
      ANS(4)=1.0
      ANS(3)=0.0
      Q=FN-P
      IF (P) 35,35,40
   35 IER=-1
      GO TO 50
   40 ANS(5)=SUM/P
      IF (Q) 45,45,60
   45 IER=1
      ANS(4)=0.0
      ANS(3)=1.0
   50 DO 55 I=5,8
   55 ANS(I)=1.E75
      GO TO 65
   60 ANS(6)=SUM2/Q
      P=P/FN
      Q=1.0-P
C
C        FIND ORDINATE OF THE NORMAL DISTRIBUTION CURVE AT THE POINT OF
C        DIVISION BETWEEN SEGMENTS CONTAINING P AND Q PROPORTIONS
C
      CALL NDTRI (Q,X,Y,ER)
C
C        COMPUTE THE BISERIAL COEFFICIENT OF CORRELATION
C
      R=((ANS(5)-ANS(1))/ANS(2))*(P/Y)
C
C        COMPUTE THE STANDARD ERROR OF R
C
      ANS(8)=( SQRT(P*Q)/Y-R*R)/SQRT(FN)
C
C        STORE RESULTS
C
      ANS(3)=P
      ANS(4)=Q
      ANS(7)=R
C
   65 RETURN
      END
```

Subroutine PHI

This subroutine computes a phi coefficient between two variables which are dichotomous. The computational steps are as follows:

1. Two dichotomous variables are summarized in a 2 x 2 table as shown below:

		Variable 1: High	Variable 1: Low
Variable 2	High	A	B
	Low	C	D

A, B, C, and D stand for frequencies.

2. Compute the phi coefficient by the formula:

$$\phi = \frac{AD - BC}{\sqrt{(A+B)\,(C+D)\,(A+C)\,(B+D)}} \qquad (1)$$

3. Compute χ^2 as a function of phi coefficient as follows:

$$\chi^2 = N\,\phi^2 \qquad (2)$$

where N = number of observations.

χ^2, with one degree of freedom, may be tested at a required level of significance. If it is significant, the obtained phi coefficient is also significant.

4. Compute the maximal phi coefficient that can be attainable in a given problem by the formula:

$$\phi_{max} = \sqrt{\left(\frac{p_j}{q_j}\right)\left(\frac{q_i}{p_i}\right)} \qquad (3)$$

where p_i is the largest marginal proportion in a 2 x 2 contingency table, and p_j is the corresponding marginal proportion in the other variable.

For reference see:

(1) J. P. Guilford, Fundamental Statistics in Psychology and Education. McGraw-Hill, New York, 1956, chapter 13.

(2) G. U. Yule, "On the Methods of Measuring the Association Between Two Attributes", Journal of Royal Statistical Association, 1912, 75, pp. 576-642.

```
C                                                                       PHI   10
C     ..................................................................PHI   20
C                                                                       PHI   30
C        SUBROUTINE PHI                                                 PHI   40
C                                                                       PHI   50
C        PURPOSE                                                        PHI   60
C           TO COMPUTE THE PHI COEFFICIENT BETWEEN TWO VARIABLES  WHICH PHI   70
C           ARE DICHOTOMOUS.                                            PHI   80
C                                                                       PHI   90
C        USAGE                                                          PHI  100
C           CALL PHI (N,U,V,HU,HV,P,CH,XP,IE)                           PHI  110
C                                                                       PHI  120
C        DESCRIPTION OF PARAMETERS                                      PHI  130
C           N  - NUMBER OF OBSERVATIONS                                 PHI  140
C           U  - INPUT VECTOR OF LENGTH N CONTAINING THE FIRST DICHOTO- PHI  150
C                MOUS VARIABLE                                          PHI  160
C           V  - INPUT VECTOR OF LENGTH N CONTAINING THE SECOND DICHOTO-PHI  170
C                MOUS VARIABLE                                          PHI  180
C           HU - INPUT NUMERICAL CODE WHICH INDICATES THE HIGHER        PHI  190
C                CATEGORY OF THE FIRST VARIABLE.  ANY OBSERVATION IN    PHI  200
C                VECTOR U WHICH HAS A VALUE EQUAL TO OR GREATER THAN HU PHI  210
C                WILL BE CLASSIFIED IN THE HIGHER CATEGORY.             PHI  220
C           HV - INPUT NUMERICAL CODE FOR VECTOR V, SIMILAR TO HU       PHI  230
C           P  - PHI COEFFICIENT COMPUTED                               PHI  240
C           CH - CHI-SQUARE COMPUTED AS A FUNCTION OF PHI COEFFICIENT   PHI  250
C                (DEGREES OF FREEDOM FOR CHI-SQUARE = 1)                PHI  260
C           XP - COMPUTED VALUE OF THE MAXIMAL PHI COEFFICIENT THAT     PHI  270
C                CAN BE ATTAINED IN THE PROBLEM                         PHI  280
C           IE - IF IE IS NON-ZERO, SOME CELL IN THE 2 BY 2 TABLE IS    PHI  290
C                NULL.  IF SO, P, CH, AND XP ARE SET TO 10**75.         PHI  300
C                                                                       PHI  310
C        REMARKS                                                        PHI  320
C           VARIABLES U AND V MUST BE SPECIFIED NUMERIC.                PHI  330
C           THE PHI COEFFICIENT IS A SPECIAL CASE OF THE                PHI  340
C           PEARSON PRODUCT-MOMENT CORRELATION WHEN BOTH VARIABLES ARE  PHI  350
C           BINARY.                                                     PHI  360
C                                                                       PHI  370
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PHI  380
C           NONE                                                        PHI  390
C                                                                       PHI  400
C        METHOD                                                         PHI  410
C           REFER TO P. HORST, 'PYSCHOLOGICAL MEASUREMENT AND           PHI  420
C           PREDICTION', P. 94 (WADSWORTH, 1966).                       PHI  430
C                                                                       PHI  440
C     ..................................................................PHI  450
C                                                                       PHI  460
      SUBROUTINE PHI (N,U,V,HU,HV,P,CH,XP,IE)                           PHI  470
C                                                                       PHI  480
      DIMENSION U(1),V(1)                                               PHI  490
C                                                                       PHI  500
C        CONSTRUCT A 2X2 CONTINGENCY TABLE                              PHI  510
C                                                                       PHI  520
      IE=0                                                              PHI  530
      A=0.0                                                             PHI  540
      B=0.0                                                             PHI  550
      C=0.0                                                             PHI  560
      D=0.0                                                             PHI  570
C                                                                       PHI  580
      DO 40 I=1,N                                                       PHI  590
      IF(U(I)-HU) 10,25,25                                              PHI  600
   10 IF(V(I)-HV) 15,20,20                                              PHI  610
   15 D=D+1.0                                                           PHI  620
      GO TO 40                                                          PHI  630
   20 B=B+1.0                                                           PHI  640
      GO TO 40                                                          PHI  650
   25 IF(V(I)-HV) 30,35,35                                              PHI  660
   30 C=C+1.0                                                           PHI  670
      GO TO 40                                                          PHI  680
   35 A=A+1.0                                                           PHI  690
   40 CONTINUE                                                          PHI  700
      IF(A) 100,100,41                                                  PHI  710
   41 IF(B) 100,100,42                                                  PHI  720
   42 IF(C) 100,100,43                                                  PHI  730
   43 IF(D) 100,100,44                                                  PHI  740
C                                                                       PHI  750
C        COMPUTE THE PHI COEFFICIENT                                    PHI  760
C                                                                       PHI  770
   44 P=(A*D-B*C)/ SQRT((A+B)*(C+D)*(A+C)*(B+D))                        PHI  780
C                                                                       PHI  790
C        COMPUTE CHI-SQURE                                              PHI  800
C                                                                       PHI  810
      T=N                                                               PHI  820
      CH=T*P*P                                                          PHI  830
C                                                                       PHI  840
C        COMPUTE THE MAXIMAL PHI COEFFICIENT                            PHI  850
C                                                                       PHI  860
      P1=(A+C)/T                                                        PHI  870
      P2=(B+D)/T                                                        PHI  880
      P3=(A+B)/T                                                        PHI  890
      P4=(C+D)/T                                                        PHI  900
      IF(P1-P2) 75, 45, 45                                              PHI  910
   45 IF(P3-P4) 65, 50, 50                                              PHI  920
   50 IF(P1-P3) 60, 55, 55                                              PHI  930
   55 XP=SQRT((P3/P4)*(P2/P1))                                          PHI  940
      GO TO 95                                                          PHI  950
   60 XP=SQRT((P1/P2)*(P4/P3))                                          PHI  960
      GO TO 95                                                          PHI  970
   65 IF(P1-P4) 70, 55, 55                                              PHI  980
   70 XP=SQRT((P2/P1)*(P3/P4))                                          PHI  990
      GO TO 95                                                          PHI 1000
   75 IF(P3-P4) 90, 80, 80                                              PHI 1010
   80 IF(P2-P3) 60, 85, 85                                              PHI 1020
   85 XP=SQRT((P4/P3)*(P1/P2))                                          PHI 1030
      GO TO 95                                                          PHI 1040
   90 IF(P2-P4) 70, 85, 85                                              PHI 1050
C                                                                       PHI 1060
   95 RETURN                                                            PHI 1070
  100 IE=1                                                              PHI 1080
      P=1.E75                                                           PHI 1090
      CH=1.E75                                                          PHI 1100
      XP=1.E75                                                          PHI 1110
      GO TO 95                                                          PHI 1120
      END                                                               PHI 1130
```

Subroutine POINT

This subroutine computes a point-biserial correlation coefficient between two variables, one binary (dichotomized) and one continuous. The computational steps are as follows:

1. Compute the mean of the continuously measured variable, A:

$$\bar{A} = \frac{\sum_{i=1}^{N} A_i}{N} \tag{1}$$

where N = number of observations in the sample.

2. Compute the standard deviation of the continuously measured variable, A:

$$s = \sqrt{\frac{\sum_{i=1}^{N} A_i^2 - \frac{\left(\sum_{i=1}^{N} A_i\right)^2}{N}}{N-1}} \tag{2}$$

3. Obtain the following information for the dichotomous variable, B:

N_p = number of cases in the higher group

N_q = number of cases in the lower group

4. Obtain $\bar{A}_p$ = mean of A values whose corresponding B values are in the higher group.

5. Then, compute the point-biserial correlation, r_{pb}:

$$r_{pb} = \frac{\bar{A}_p - \bar{A}}{s} \sqrt{\frac{N_p}{N_q}} \tag{3}$$

6. Compute the t ratio used to test the hypothesis of zero correlation:

$$t = r_{pb} \sqrt{\frac{N-2}{1-r_{pb}^2}} \tag{4}$$

7. The degrees of freedom for the t ratio are obtained by:

$$d.f. = N-2 \tag{5}$$

For reference see:

(1) J. P. Guilford, Fundamental Statistics in Psychology and Education. McGraw-Hill, New York, 1956, chapter 13.

(2) N. C. Perry and W. B. Michael, "The Reliability of a Point-Biserial Coefficient of Correlation". Psychometrika, 1954, 16, pp. 313-325.

```
C     ..................................................................
C
C        SUBROUTINE POINT
C
C        PURPOSE
C           TO COMPUTE THE POINT-BISERIAL CORRELATION COEFFICIENT
C           BETWEEN TWO VARIABLES, WHEN ONE OF THE VARIABLES IS A BINARY
C           VARIABLE AND ONE IS CONTINUOUS.  THIS IS A SPECIAL CASE OF
C           THE PEARSON PRODUCT-MOMENT CORRELATION COEFFICIENT.
C
C        USAGE
C           CALL POINT (N,A,B,HI,ANS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           N   - NUMBER OF OBSERVATIONS
C           A   - INPUT VECTOR OF LENGTH N CONTAINING THE CONTINUOUS
C                 VARIABLE
C           B   - INPUT VECTOR OF LENGTH N CONTAINING THE DICHOTOMOUS
C                 (BINARY) VARIABLE
C           HI  - INPUT NUMERICAL CODE TO INDICATE THE HIGHER CATEGORY.
C                 ANY VALUE OF THE BINARY VARIABLE NOT LESS THAN HI WILL
C                 BE CLASSIFIED IN THE HIGHER OF THE TWO CATEGORIES.
C           ANS - OUTPUT VECTOR OF LENGTH 9 CONTAINING THE FOLLOWING
C                 RESULTS
C                    ANS(1)- MEAN OF VARIABLE A
C                    ANS(2)- STANDARD DEVIATION OF VARIABLE A
C                    ANS(3)- NUMBER OF OBSERVATIONS IN THE HIGHER
C                            CATEGORY OF VARIABLE B
C                    ANS(4)- NUMBER OF OBSERVATIONS IN THE LOWER
C                            CATEGORY OF VARIABLE B
C                    ANS(5)- MEAN OF VARIABLE A FOR ONLY THOSE
C                            OBSERVATIONS IN THE HIGHER CATEGORY OF
C                            VARIABLE B
C                    ANS(6)- MEAN OF VARIABLE A FOR ONLY THOSE
C                            OBSERVATIONS IN THE LOWER CATEGORY OF
C                            VARIABLE B
C                    ANS(7)- POINT-BISERIAL CORRELATION COEFFICIENT
C                    ANS(8)- T-TEST FOR THE SIGNIFICANCE OF THE
C                            DIFFERENCE BETWEEN THE MEANS OF VARIABLE A
C                            FOR THE HIGHER AND LOWER CATEGORIES
C                            RESPECTIVELY.
C                    ANS(9)- DEGREES OF FREEDOM FOR THE T-TEST
C           IER- 1, IF ALL ELEMENTS OF B ARE NOT LESS THAN HI.
C                -1, IF ALL ELEMENTS OF B ARE LESS THAN HI.
C                0, OTHERWISE.  IF IER IS NON-ZERO, ANS(I), I=5,....,9,
C                IS SET TO 10**75.
C
C        REMARKS
C           THE SYMBOLS USED TO IDENTFY THE VALUES OF THE TWO CATEGORIES
C           OF VARIABLE B MUST BE NUMERIC.  ALPHABETIC OR SPECIAL
C           CHARACTERS CANNOT BE USED.
C           THE T-TEST(ANS(8)) IS A TEST OF WHETHER THE POINT-BISERIAL
C           COEFFICIENT DIFFERS SIGNIFICANTLY FROM ZERO.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           REFER TO P. HORST, 'PSYCHOLOGICAL MEASUREMENT AND
C           PREDICTION', P. 91 (WADSWORTH, 1966).
C     ..................................................................
C
      SUBROUTINE POINT (N,A,B,HI,ANS,IER)
C
      DIMENSION A(1),B(1),ANS(1)
C
C        COMPUTE MEAN AND STANDARD DEVIATION OF VARIABLE A
C
      IER=0
      SUM=0.0
      SUM2=0.0
      DO 10 I=1,N
      SUM=SUM+A(I)
   10 SUM2=SUM2+A(I)*A(I)
      FN=N
      ANS(1)=SUM/FN
      ANS(2)=(SUM2-ANS(1)*SUM)/(FN-1.0)
      ANS(2)= SQRT(ANS(2))
C
C        FIND NUMBERS OF CASES IN THE HIGHER AND LOWER CATEGORIES
C
      P=0.0
      SUM=0.0
      SUM2=0.0
      DO 30 I=1,N
      IF(B(I)-HI) 20, 25, 25
   20 SUM2=SUM2+A(I)
      GO TO 30
   25 P=P+1.0
      SUM=SUM+A(I)
   30 CONTINUE
C
      Q=FN-P
      ANS(3)=P
      ANS(4)=0
      IF (P) 35,35,40
   35 IER=-1
      GO TO 50
   40 ANS(5)=SUM/P
      IF (Q) 45,45,60
   45 IER=1
   50 DO 55 I=5,9
   55 ANS(I)=1.E75
      GO TO 65
   60 ANS(6)=SUM2/Q
C
C        COMPUTE THE POINT-BISERIAL CORRELATION
C
      R=((ANS(5)-ANS(1))/ANS(2))* SQRT(P/Q)
      ANS(7)=R
C
C        COMPUTE T RATIO USED TO TEST THE HYPOTHESIS OF ZERO CORRELATION
C
      T=R* SQRT((FN-2.0)/(1.0-R*R))
      ANS(8)=T
C
C        COMPUTE DEGREES OF FREEDOM
C
      ANS(9)=FN-2
C
   65 RETURN
      END
```

Subroutine TETRA

This subroutine computes a tetrachoric correlation coefficient between two variables where both of the variables have been reduced artifically to two categories. The computational steps are as follows:

1. Two dichotomous variables are summarized in a 2 x 2 table such as the one shown below.

		Variable 1	
		High	Low
Variable 2	High	A	B
	Low	C	D

A, B, C, and D stand for frequencies. If any frequency is zero (0), the tetrachoric correlation will not be computed.

2. Obtain the proportions of marginal totals:

$$p_1 = (A+C)/N \quad (1)$$
$$q_1 = (B+D)/N \quad (2)$$
$$p_2 = (A+B)/N \quad (3)$$
$$q_2 = (C+D)/N \quad (4)$$

where N = number of observations.

3. Obtain:

y_1 = ordinate of the normal distribution curve at the point of division between p_1 and q_1 proportions

y_2 = same as y_1 using p_2 and q_2 proportions

z_1 = standard normal variable corresponding to y_1

z_2 = standard normal variable corresponding to y_2

4. Then, compute the tetrachoric correlation coefficient by the use of the following formula:

$$\begin{aligned}\frac{AD - BC}{y_1 y_2 N^2} = r_t &+ \frac{1}{2} z_1 z_2 r_t^2 \\ &+ \frac{1}{6}\left(z_1^2 - 1\right)\left(z_2^2 - 1\right) r_t^3 \\ &+ \frac{1}{24} z_1 \left(z_1^2 - 3\right) z_2 \left(z_2^2 - 3\right) r_t^4 \\ &+ \frac{1}{120}\left(z_1^4 - 6z_1^2 + 3\right)\left(z_2^4 - 6z_2^2 + 3\right) r_t^5 \\ &+ \frac{1}{720} z_1 \left(z_1^4 - 10z_1^2 + 15\right) \\ &\quad z_2 \left(z_2^4 - 10z_2^2 + 15\right) r_t^6 \\ &+ \frac{1}{5040}\left(z_1^6 - 15z_1^4 + 45z_1^2 - 15\right) \\ &\quad \left(z_2^6 - 15z_2^4 + 45z_2^2 - 15\right) r_t^7\end{aligned} \quad (5)$$

Note: r_t is found by the subroutine POLRT -- Newton-Raphson method of approximation to the root of the above equation.

5. For the purpose of testing $r_t = 0.0$, the standard error is estimated by the formula:

$$s_{r_t} = \frac{\sqrt{p_1 p_2 q_1 q_2}}{y_1 y_2 \sqrt{N}} \qquad (6)$$

6. In selecting the real root in [-1, 1], if $r_t = a + bi$, and

$$|b| \leq .5(10^{-6})\,|a|$$

r_t is considered to be a real root.

For reference see:

(1) J. P. Guilford, Fundamental Statistics in Psychology and Education. McGraw-Hill, New York, 1956, chapter 13.

(2) W. P. Elderton, Frequency Curves and Correlation, 4th Edition. Cambridge University Press, 1953, chapter 9.

```
C                                                                       TETR  10
C     ...................................................................TETR  20
C                                                                       TETR  30
C        SUBROUTINE TETRA                                               TETR  40
C                                                                       TETR  50
C        PURPOSE                                                        TETR  60
C           COMPUTE A TETRACHORIC CORRELATION COEFFICIENT BETWEEN TWO   TETR  70
C           VARIABLES WHERE DATA IN BOTH VARIABLES HAVE BEEN REDUCED    TETR  80
C           ARTIFICIALLY TO TWO CATEGORIES.                             TETR  90
C                                                                       TETR 100
C        USAGE                                                          TETR 110
C           CALL TETRA (N,U,V,HU,HV,R,RS,IE)                            TETR 120
C                                                                       TETR 130
C        DESCRIPTION OF PARAMETERS                                      TETR 140
C           N  - NUMBER OF OBSERVATIONS                                 TETR 150
C           U  - INPUT VECTOR OF LENGTH N CONTAINING THE FIRST VARIABLE TETR 160
C                REDUCED TO TWO CATEGORIES                              TETR 170
C           V  - INPUT VECTOR OF LENGTH N CONTAINING THE SECOND VARIABLETETR 180
C                REDUCED TO TWO CATEGORIES                              TETR 190
C           HU - INPUT NUMERICAL CODE INDICATING THE HIGHER CATEGORY OF TETR 200
C                THE FIRST VARIABLE.  IF ANY VALUE OF VARIABLE U IS     TETR 210
C                EQUAL TO OR GREATER THAN HU, IT WILL BE CLASSIFIED AS  TETR 220
C                THE HIGHER CATEGORY, OTHERWISE AS THE LOWER CATEGORY.  TETR 230
C           HV - SAME AS HU EXCEPT THAT HV IS FOR THE SECOND VARIABLE.  TETR 240
C           R  - TETRACHORIC CORRELATION COMPUTED                       TETR 250
C           RS - STANDARD ERROR OF TETRACHORIC CORRELATION COMPUTED     TETR 260
C           IE - ERROR CODE                                             TETR 270
C                0 - NO ERROR                                           TETR 280
C                1 - UNABLE TO COMPUTE A TETRACHORIC CORRELATION DUE TO TETR 290
C                    THE FACT THAT AT LEAST ONE CELL SHOWS ZERO FRE-    TETR 300
C                    QUENCY IN THE 2X2 CONTINGENCY TABLE CONSTRUCTED    TETR 310
C                    FROM INPUT DATA.  IN THIS CASE, R AND RS ARE SET   TETR 320
C                    TO 10**75.  (SEE GUILFORD, 1956)                   TETR 330
C                2 - THE ROOT SOLVER GIVES MULTIPLE ROOTS, OR NO ROOTS, TETR 340
C                    R, IN THE INTERVAL (-1,1) INCLUSIVE. R AND RS ARE  TETR 350
C                    SET TO 10**75.                                     TETR 360
C                3 - UNABLE TO COMPUTE A SATISFACTORY VALUE OF TETRA-   TETR 370
C                    CHORIC CORRELATION USING NEWTON-RAPHSON METHOD OF  TETR 380
C                    APPROXIMATION TO THE ROOT OF THE EQUATION.  R AND  TETR 390
C                    RS ARE SET TO 10**75.  SEE SUBROUTINE POLRT ERROR  TETR 400
C                    INDICATORS.                                        TETR 410
C                4 - HIGH ORDER COEFFICIENT OF THE POLYNOMIAL IS ZERO.  TETR 420
C                    SEE SUBROUTINE POLRT ERROR INDICATORS.             TETR 430
C                                                                       TETR 440
C        REMARKS                                                        TETR 450
C           VALUES OF VARIABLES U AND V MUST BE NUMERICAL, AND          TETR 460
C           ALPHABETIC AND SPECIAL CHARACTERS MUST NOT BE USED.         TETR 470
C           FOR A DEPENDABLE RESULT FOR TETRACHORIC CORRELATION,        TETR 480
C           IT IS RECOMMENDED THAT N BE AT LEAST 200 OR GREATER.        TETR 490
C                                                                       TETR 500
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TETR 510
C           NDTRI                                                       TETR 520
C           POLRT--THIS POLYNOMIAL ROOT ROUTINE WAS SELECTED BECAUSE OF TETR 530
C                  ITS SMALL STORAGE REQUIREMENT.  OTHER SSP ROUTINES   TETR 540
C                  WHICH COULD REPLACE POLRT ARE PROD AND PRBM.  THEIR  TETR 550
C                  USE WOULD REQUIRE MODIFICATION OF TETRA.             TETR 560
C                                                                       TETR 570
C        METHOD                                                         TETR 580
C           REFER TO J. P. GUILFORD, 'FUNDAMENTAL STATISTICS IN PSYCHO- TETR 590
C           LOGY AND EDUCATION', MCGRAW-HILL, NEW YORK, 1956, CHAPTER 13TETR 600
C           AND W. P. ELDERTON, 'FREQUENCY CURVES AND CORRELATION' 4-TH TETR 610
C           ED., CAMBRIDGE UNIVERSITY PRESS, 1953, CHAPTER 9.           TETR 620
C                                                                       TETR 630
C     ...................................................................TETR 640
C                                                                       TETR 650
      SUBROUTINE TETRA (N,U,V,HU,HV,R,RS,IE)                            TETR 660
C                                                                       TETR 670
      DIMENSION XCOF(8),COF(8),ROOTR(7),ROOTI(7)                        TETR 680
      DIMENSION U(1),V(1)                                               TETR 690
      DOUBLE PRECISION X31,X32,X312,X322                                TETR 700
C                                                                       TETR 710
C        CONSTRUCT A 2X2 CONTINGENCY TABLE                              TETR 720
C                                                                       TETR 730
      A=0.0                                                             TETR 740
      B=0.0                                                             TETR 750
      C=0.0                                                             TETR 760
      D=0.0                                                             TETR 770
      DO 40 I=1,N                                                       TETR 780
      IF(U(I)-HU) 10, 25, 25                                            TETR 790
   10 IF(V(I)-HV) 15, 20, 20                                            TETR 800
   15 D=D+1.0                                                           TETR 810
      GO TO 40                                                          TETR 820
   20 B=B+1.0                                                           TETR 830
      GO TO 40                                                          TETR 840
   25 IF(V(I)-HV) 30, 35, 35                                            TETR 850
   30 C=C+1.0                                                           TETR 860
      GO TO 40                                                          TETR 870
   35 A=A+1.0                                                           TETR 880
   40 CONTINUE                                                          TETR 890
C                                                                       TETR 900
C        TEST WHETHER ANY CELL IN THE CONTINGENCY TABLE IS ZERO.        TETR 910
C        IF SO, RETURN TO THE CALLING ROUTINE WITH R=0.0 AND IE=1.      TETR 920
C                                                                       TETR 930
      IE=0                                                              TETR 940
      IF(A) 60, 60, 45                                                  TETR 950
   45 IF(B) 60, 60, 50                                                  TETR 960
   50 IF(C) 60, 60, 55                                                  TETR 970
   55 IF(D) 60, 60, 70                                                  TETR 980
   60 IE=1                                                              TETR 990
      GO TO 86                                                          TETR1000
C                                                                       TETR1010
C        COMPUTE P1, Q1, P2, AND Q2                                     TETR1020
C                                                                       TETR1030
   70 FN=N                                                              TETR1040
      P1=(A+C)/FN                                                       TETR1050
      Q1=(B+D)/FN                                                       TETR1060
      P2=(A+B)/FN                                                       TETR1070
      Q2=(C+D)/FN                                                       TETR1080
C                                                                       TETR1090
C        FIND THE STANDARD NORMAL DEVIATES AT Q1 AND Q2, AND THE        TETR1100
C        ORDINATES AT THOSE POINTS                                      TETR1110
C                                                                       TETR1120
      CALL NDTRI (Q1,X1,Y1,FR)                                          TETR1130
      CALL NDTRI (Q2,X2,Y2,ER)                                          TETR1140
C                                                                       TETR1150
C        COMPUTE THE TETRACHORIC CORRELATION COEFFICIENT                TETR1160
C                                                                       TETR1170
      IF(X1) 76, 72, 76                                                 TETR1180
   72 IF(X2) 76, 74, 76                                                 TETR1190
   74 R=0.0                                                             TETR1200
      GO TO 90                                                          TETR1210
   76 XCOF(1)=-((A*D-B*C)/(Y1*Y2*FN*FN))                                TETR1220
      XCOF(2)=1.0                                                       TETR1230
      XCOF(3)=X1*X2/2.0                                                 TETR1240
      XCOF(4)=(X1*X1-1.0)*(X2*X2-1.0)/6.0                               TETR1250
      X31=DBLE(X1)                                                      TETR1260
      X32=DBLE(X2)                                                      TETR1270
      X312=X31**2                                                       TETR1280
      X322=X32**2                                                       TETR1290
      XCOF(5)=SNGL(X31*(X312-3.0D0)*X32*(X322-3.0D0)/24.0D0)            TETR1300
      XCOF(6)=SNGL((X312*(X312-6.0D0)+3.0D0)*(X322*(X322-6.0D0)+3.0D0)  TETR1310
     1        /120.0D0)                                                 TETR1320
      XCOF(7)=SNGL(X31*(X312*(X312-10.0D0)+15.0D0)*X32*(X322*(X322-10.0 TETR1330
     1        D0)+15.0D0)/720.0D0)                                      TETR1340
      XCOF(8)=SNGL((((X312-15.0D0)*X312+45.0D0)*X312-15.0D0)*(((X322-   TETR1350
     1        15.0D0)*X322+45.0D0)*X322-15.0D0)/5040.0D0)               TETR1360
C                                                                       TETR1370
      CALL POLRT (XCOF,COF,7,ROOTR,ROOTI,IER)                           TETR1380
C                                                                       TETR1390
      J=0                                                               TETR1400
      IF(IER) 78, 78, 84                                                TETR1410
   78 DO 82 I=1,7                                                       TETR1420
      IF(ABS(ROOTI(I))-.5*ABS(ROOTR(I))*1.0E-6)79,79,82                 TETR1430
   79 R=ROOTR(I)                                                        TETR1440
      IF(ABS(R)-1.0)81,81,80                                            TETR1450
   80 R=1.E75                                                           TETR1460
      GO TO 82                                                          TETR1470
   81 J=J+1                                                             TETR1480
   82 CONTINUE                                                          TETR1490
      IF(J-1)83,88,83                                                   TETR1500
   83 IE=2                                                              TETR1510
      GO TO 86                                                          TETR1520
C                                                                       TETR1530
C        UNABLE TO COMPUTE R                                            TETR1540
C                                                                       TETR1550
   84 IE=IER                                                            TETR1560
   86 R=1.0E75                                                          TETR1570
      RS=R                                                              TETR1580
      GO TO 100                                                         TETR1590
   88 IF(R-1.0E75)90,83,83                                              TETR1600
C                                                                       TETR1610
C        STANDARD ERROR OF R=0.0                                        TETR1620
C                                                                       TETR1630
   90 RS= SQRT(P1*P2*Q1*Q2)/(Y1*Y2* SQRT(FN))                           TETR1640
C                                                                       TETR1650
  100 RETURN                                                            TETR1660
      END                                                               TETR1670
```

Subroutine SRATE

This subroutine computes the survival rates of subjects alive n periods after diagnosis or treatment. The survival rate for each period is computed successively for each of K periods where K is specified and $K \leq n$. An example of the input to this subroutine is presented in Table 2.

Table 2. Example of Input to Subroutine SRATE

Periods after diagnosis	Alive at beginning of period	Died during period	Lost during period	Withdrawn alive during period
(i)	(L_i)	(D_i)	(U_i)	(W_i)
1	130	40	4	10
2	76	6	2	8
3	60	2	1	5
4	52	1	0	4
5	47	0	0	47

The input information includes:

- Periods after diagnosis. This column shows the time period elapsed from the date of diagnosis or treatment. A period may be any time interval.
- Alive at beginning of period. The first entry in this column indicates the initial number of subjects; in Table 2, $L_1 = 130$. Each entry thereafter is obtained by subtracting from the preceding number of subjects the total of those who are no longer in the program; in Table 2, $L_2 = L_1 - (D_1 + U_1 + W_1) = 130 - (40 + 4 + 10) = 76$.
- Died during period. This column indicates the number of subjects that died during the period.
- Lost during period. This column indicates the number of subjects whose survival status became unknown during the period.
- Withdrawn alive during period. This column indicates the number of living subjects withdrawn from the study during the period.

For the value of $i = 1, 2, \ldots, K$, the subroutine computes the following information:

1. Effective number exposed to risk of dying during period:

$$E_i = L_i - (U_i + W_i)/2 \qquad (1)$$

2. Proportion dying during period:

$$Q_i = \frac{D_i}{E_i} \qquad (2)$$

3. Proportion surviving during period:

$$P_i = 1 - Q_i \qquad (3)$$

4. i^{th} period survival rate:

$$R_i = \prod_{j=1}^{i} P_i \qquad (4)$$

5. Standard error of survival rate:

$$s_i = R_i \sqrt{\sum_{j=1}^{i} \frac{Q_j}{\bar{E}_j - D_j}} \qquad (5)$$

For reference see: S. J. Cutler and F. Ederer, "Maximum Utilization of the Life Table Method in Analyzing Survival", Journal of Chronic Diseases, December, 1958. pp 699-712.

```
C                                                                       SRAT  10
C     ...................................................................SRAT  20
C                                                                       SRAT  30
C        SUBROUTINE SRATE                                               SRAT  40
C                                                                       SRAT  50
C        PURPOSE                                                        SRAT  60
C           TO COMPUTE THE PROPORTION OF SUBJECTS SURVIVING, THE        SRAT  70
C           SURVIVAL RATES AND THE STANDARD ERRORS FOR SUCCESSIVELY     SRAT  80
C           REDUCED TIME PERIODS.  THE SURVIVAL RATE IS COMPUTED FOR    SRAT  90
C           EACH OF K PERIODS, WHERE K IS A CONSTANT TO BE SPECIFIED ANDSRAT 100
C           IS LESS THAN OR EQUAL TO N (WHERE N = TOTAL NUMBER OF       SRAT 110
C           PERIODS).                                                   SRAT 120
C                                                                       SRAT 130
C        USAGE                                                          SRAT 140
C           CALL SRATE (N,K,X,IE)                                       SRAT 150
C                                                                       SRAT 160
C        DESCRIPTION OF PARAMETERS                                      SRAT 170
C           N - THE TOTAL NUMBER OF PERIODS AFTER TREATMENT OR DIAGNOSISSRAT 180
C           K - THE SPECIFIED PERIOD UP TO WHICH SURVIVAL RATES ARE TO  SRAT 190
C               BE CALCULATED                                           SRAT 200
C           X - AN INPUT AND OUTPUT MATRIX (N X 9) CONTAINING THE       SRAT 210
C               FOLLOWING INFORMATION                                   SRAT 220
C               FOR INPUT--STORED IN THE N ROWS OF EACH COLUMN          SRAT 230
C                  COL 1 - NUMBER OF SUBJECTS ALIVE AT THE BEGINNING OF SRAT 240
C                          PERIOD                                       SRAT 250
C                  COL 2 - NUMBER OF SUBJECTS WHICH DIED DURING THE     SRAT 260
C                          PERIOD                                       SRAT 270
C                  COL 3 - NUMBER OF SUBJECTS LOST TO FOLLOW-UP DURING  SRAT 280
C                          THE PERIOD                                   SRAT 290
C                  COL 4 - NUMBER OF SUBJECTS WITHDRAWN ALIVE DURING THESRAT 300
C                          PERIOD                                       SRAT 310
C                 FOR OUTPUT--STORED IN THE FIRST K ROWS OF EACH        SRAT 320
C                 COLUMN                                                SRAT 330
C                  COL 5 - EFFECTIVE NUMBER EXPOSED TO THE RISK OF DYINGSRAT 340
C                  COL 6 - PROPORTION WHO DIED DURING THE PERIOD        SRAT 350
C                  COL 7 - PROPORTION WHO SURVIVED DURING THE PERIOD    SRAT 360
C                  COL 8 - SURVIVAL RATE                                SRAT 370
C                  COL 9 - STANDARD ERROR OF THE SURVIVAL RATE          SRAT 380
C           IE- 1, IF K IS NOT IN THE CLOSED INTERVAL (0,N).            SRAT 390
C               2, IF THE NUMBER OF SUBJECTS ALIVE AT THE BEGINNING     SRAT 400
C               OF PERIOD I IS LESS THAN THE SUM OF THOSE WHICH DIED,   SRAT 410
C               WERE LOST, OR WERE WITHDRAWN DURING PERIOD I (I=1,...N) SRAT 420
C               3, IF THE NUMBER OF SUBJECTS WHICH DIED, WERE LOST, OR  SRAT 430
C               WERE WITHDRAWN IN PERIOD I IS NOT EQUAL TO THE NUMBER   SRAT 440
C               ALIVE AT THE BEGINNING OF PERIOD I LESS THE NUMBER      SRAT 450
C               ALIVE AT THE BEGINNING OF PERIOD I + 1 (I=1,...N-1)     SRAT 460
C                                                                       SRAT 470
C        REMARKS                                                        SRAT 480
C           IF THE SUBJECTS IN A GIVEN GROUP ARE ALL DIAGNOSED OR       SRAT 490
C           TREATED AT THE SAME TIME, THE CONSTANT K MAY BE SET EQUAL TOSRAT 500
C           N .  IF THE SUBJECTS IN A GIVEN GOUP ENTER THE STUDY AT     SRAT 510
C           VARYING TIMES, K CAN BE NO GREATER THAN N-1.                SRAT 520
C                                                                       SRAT 530
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SRAT 540
C           NONE                                                        SRAT 550
C                                                                       SRAT 560
C        METHOD                                                         SRAT 570
C           REFER TO S. J. CUTLER AND F. EDERER 'MAXIMUM UTILIZATION OF SRAT 580
C           THE LIFE TABLE METHOD IN ANALYZING SURVIVAL', JOURNAL OF    SRAT 590
C           CHRONIC DISEASES, DECEMBER, 1958.  PP 699-712.              SRAT 600
```

```
C                                                                       SRAT 610
C     ..................................................................SRAT 620
C                                                                       SRAT 630
      SUBROUTINE SRATE (N,K,X,IE)                                       SRAT 640
C                                                                       SRAT 650
      DIMENSION X(1)                                                    SRAT 660
C                                                                       SRAT 670
C        INITIALIZATION AND ERROR CHECKING                              SRAT 680
C                                                                       SRAT 690
      IE=0                                                              SRAT 700
      NP4=4*N+1                                                         SRAT 710
      NP9=NP4+NP4+N-2                                                   SRAT 720
      DO 1 I=NP4,NP9                                                    SRAT 730
    1 X(I)=0.0                                                          SRAT 740
      IF (K) 2,2,3                                                      SRAT 750
    2 IE=1                                                              SRAT 760
      GO TO 45                                                          SRAT 770
    3 IF(K-N) 4,4,2                                                     SRAT 780
    4 DO 9 I=1,N                                                        SRAT 790
      NP4=I+N                                                           SRAT 800
      NP9=NP4+N                                                         SRAT 810
      NP1=NP9+N                                                         SRAT 820
      IF(INT(X(I)-X(NP4)-X(NP9)-X(NP1)+.01)) 5,6,6                      SRAT 830
    5 IE=2                                                              SRAT 840
      GO TO 45                                                          SRAT 850
    6 IF(I-N) 7,9,9                                                     SRAT 860
    7 IF (INT(X(I+1)-X(I)+X(NP4)+X(NP9)+X(NP1)+.01)) 8,9,8              SRAT 870
    8 IE=3                                                              SRAT 880
        GO TO 45                                                        SRAT 890
    9 CONTINUE                                                          SRAT 900
   15 L1=0                                                              SRAT 910
      L2=L1+N                                                           SRAT 920
      L3=L2+N                                                           SRAT 930
      L4=L3+N                                                           SRAT 940
      L5=L4+N                                                           SRAT 950
      L6=L5+N                                                           SRAT 960
      L7=L6+N                                                           SRAT 970
      L8=L7+N                                                           SRAT 980
      L9=L8+N                                                           SRAT 990
      LD=L2                                                             SRAT1000
      LE=L5                                                             SRAT1010
      LQ=L6                                                             SRAT1020
      SUM=0.0                                                           SRAT1030
C                                                                       SRAT1040
      DO 40 I=1,K                                                       SRAT1050
C                                                                       SRAT1060
C        COMPUTE EFFECTIVE NUMBER EXPOSED TO RISK OF DYING              SRAT1070
C                                                                       SRAT1080
      L1=L1+1                                                           SRAT1090
      L3=L3+1                                                           SRAT1100
      L4=L4+1                                                           SRAT1110
      L5=L5+1                                                           SRAT1120
      X(L5)=X(L1)-(X(L3)+X(L4))/2.0                                     SRAT1130
C                                                                       SRAT1140
C        COMPUTE PROPORTION OF DYING                                    SRAT1150
C                                                                       SRAT1160
      L2=L2+1                                                           SRAT1170
      L6=L6+1                                                           SRAT1180
      X(L6)=X(L2)/X(L5)                                                 SRAT1190
C                                                                       SRAT1200
C        COMPUTE PROPORTION OF SURVIVING                                SRAT1210
C                                                                       SRAT1220
      L7=L7+1                                                           SRAT1230
      X(L7)=1.0-X(L6)                                                   SRAT1240
C                                                                       SRAT1250
C        COMPUTE SURVIVAL RATE                                          SRAT1260
C                                                                       SRAT1270
      L8=L8+1                                                           SRAT1280
      IF (I-1) 20, 20, 25                                               SRAT1290
   20 X(LQ)=X(L7)                                                       SRAT1300
      GO TO 30                                                          SRAT1310
   25 X(L8)=X(L8-1)*X(L7)                                               SRAT1320
C                                                                       SRAT1330
C        COMPUTE STANDARD ERROR OF SURVIVAL RATE                        SRAT1340
C                                                                       SRAT1350
   30 L9=L9+1                                                           SRAT1360
      SUM=SUM+X(L6)/(X(L5)-X(L2))                                       SRAT1370
   40 X(L9)=X(L8)*SQRT(SUM)                                             SRAT1380
C                                                                       SRAT1390
   45 RETURN                                                            SRAT1400
      END                                                               SRAT1410
```

Matrices: Storage

Subroutine MCPY

```
C
C     ..................................................................
C
C     SUBROUTINE MCPY
C
C     PURPOSE
C        COPY ENTIRE MATRIX
C
C     USAGE
C        CALL MCPY (A,R,N,M,MS)
C
C     DESCRIPTION OF PARAMETERS
C        A - NAME OF INPUT MATRIX
C        R - NAME OF OUTPUT MATRIX
C        N - NUMBER OF ROWS IN A OR R
C        M - NUMBER OF COLUMNS IN A OR R
C        MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R)
C              0 - GENERAL
C              1 - SYMMETRIC
C              2 - DIAGONAL
C
C     REMARKS
C        NONE
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        LOC
C
C     METHOD
C        EACH ELEMENT OF MATRIX A IS MOVED TO THE CORRESPONDING
C        ELEMENT OF MATRIX R
C
C     ..................................................................
      SUBROUTINE MCPY(A,R,N,M,MS)
      DIMENSION A(1),R(1)
C
C        COMPUTE VECTOR LENGTH, IT
C
      CALL LOC(N,M,IT,N,M,MS)
C
C        COPY MATRIX
C
      DO 1 I=1,IT
    1 R(I)=A(I)
      RETURN
      END
```

Subroutine RCPY

```
C
C     ..................................................................
C
C     SUBROUTINE RCPY
C
C     PURPOSE
C        COPY SPECIFIED ROW OF A MATRIX INTO A VECTOR
C
C     USAGE
C        CALL RCPY (A,L,R,N,M,MS)
C
C     DESCRIPTION OF PARAMETERS
C        A - NAME OF INPUT MATRIX
C        L - ROW OF A TO BE MOVED TO R
C        R - NAME OF OUTPUT VECTOR OF LENGTH M
C        N - NUMBER OR ROWS IN A
C        M - NUMBER OF COLUMNS IN A
C        MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C              0 - GENERAL
C              1 - SYMMETRIC
C              2 - DIAGONAL
C
C     REMARKS
C        NONE
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        LOC
C
C     METHOD
C        ELEMENTS OF ROW L ARE MOVED TO CORRESPONDING POSITIONS
C        OF VECTOR R
C
C     ..................................................................
      SUBROUTINE RCPY(A,L,R,N,M,MS)
      DIMENSION A(1),R(1)
C
      DO 3 J=1,M
C
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE
C
      CALL LOC(L,J,LJ,N,M,MS)
C
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX
C
      IF(LJ) 1,2,1
C
C        MOVE ELEMENT TO R
C
    1 R(J)=A(LJ)
      GO TO 3
    2 R(J)=0.0
    3 CONTINUE
      RETURN
      END
```

Subroutine CCPY

```
C                                                                       CCPY  10
C     ..................................................................CCPY  20
C                                                                       CCPY  30
C        SUBROUTINE CCPY                                                CCPY  40
C                                                                       CCPY  50
C        PURPOSE                                                        CCPY  60
C           COPY SPECIFIED COLUMN OF A MATRIX INTO A VECTOR             CCPY  70
C                                                                       CCPY  80
C        USAGE                                                          CCPY  90
C           CALL CCPY(A,L,R,N,M,MS)                                     CCPY 100
C                                                                       CCPY 110
C        DESCRIPTION OF PARAMETERS                                      CCPY 120
C           A - NAME OF INPUT MATRIX                                    CCPY 130
C           L - COLUMN OF A TO BE MOVED TO R                            CCPY 140
C           R - NAME OF OUTPUT VECTOR OF LENGTH N                       CCPY 150
C           N - NUMBER OR ROWS IN A                                     CCPY 160
C           M - NUMBER OF COLUMNS IN A                                  CCPY 170
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         CCPY 180
C                 0 - GENERAL                                           CCPY 190
C                 1 - SYMMETRIC                                         CCPY 200
C                 2 - DIAGONAL                                          CCPY 210
C                                                                       CCPY 220
C        REMARKS                                                        CCPY 230
C           NONE                                                        CCPY 240
C                                                                       CCPY 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CCPY 260
C           LOC                                                         CCPY 270
C                                                                       CCPY 280
C        METHOD                                                         CCPY 290
C           ELEMENTS OF COLUMN L ARE MOVED TO CORRESPONDING POSITIONS   CCPY 300
C           OF VECTOR R                                                 CCPY 310
C                                                                       CCPY 320
C     ..................................................................CCPY 330
C                                                                       CCPY 340
      SUBROUTINE CCPY(A,L,R,N,M,MS)                                     CCPY 350
      DIMENSION A(1),R(1)                                               CCPY 360
C                                                                       CCPY 370
      DO 3 I=1,N                                                        CCPY 380
C                                                                       CCPY 390
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE                     CCPY 400
C                                                                       CCPY 410
      CALL LOC(I,L,IL,N,M,MS)                                           CCPY 420
C                                                                       CCPY 430
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       CCPY 440
C                                                                       CCPY 450
      IF(IL) 1,2,1                                                      CCPY 460
C                                                                       CCPY 470
C        MOVE ELEMENT TO R                                              CCPY 480
C                                                                       CCPY 490
    1 R(I)=A(IL)                                                        CCPY 500
      GO TO 3                                                           CCPY 510
    2 R(I)=0.0                                                          CCPY 520
    3 CONTINUE                                                          CCPY 530
      RETURN                                                            CCPY 540
      END                                                               CCPY 550
```

Subroutine DCPY

```
C                                                                       DCPY  10
C     ..................................................................DCPY  20
C                                                                       DCPY  30
C        SUBROUTINE DCPY                                                DCPY  40
C                                                                       DCPY  50
C        PURPOSE                                                        DCPY  60
C           COPY DIAGONAL ELEMENTS OF A MATRIX INTO A VECTOR            DCPY  70
C                                                                       DCPY  80
C        USAGE                                                          DCPY  90
C           CALL DCPY (A,R,N,MS)                                        DCPY 100
C                                                                       DCPY 110
C        DESCRIPTION OF PARAMETERS                                      DCPY 120
C           A - NAME OF INPUT MATRIX                                    DCPY 130
C           R - NAME OF OUTPUT VECTOR OF LENGTH N                       DCPY 140
C           N - NUMBER OF ROWS AND COLUMNS IN MATRIX A                  DCPY 150
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         DCPY 160
C                 0 - GENERAL                                           DCPY 170
C                 1 - SYMMETRIC                                         DCPY 180
C                 2 - DIAGONAL                                          DCPY 190
C                                                                       DCPY 200
C        REMARKS                                                        DCPY 210
C           INPUT MATRIX MUST BE A SQUARE MATRIX                        DCPY 220
C                                                                       DCPY 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DCPY 240
C           LOC                                                         DCPY 250
C                                                                       DCPY 260
C        METHOD                                                         DCPY 270
C           ELEMENTS ON DIAGONAL OF MATRIX ARE MOVED TO CORRESPONDING   DCPY 280
C           POSITIONS OF VECTOR R                                       DCPY 290
C                                                                       DCPY 300
C     ..................................................................DCPY 310
C                                                                       DCPY 320
      SUBROUTINE DCPY(A,R,N,MS)                                         DCPY 330
      DIMENSION A(1),R(1)                                               DCPY 340
C                                                                       DCPY 350
      DO 3 J=1,N                                                        DCPY 360
C                                                                       DCPY 370
C        LOCATE DIAGONAL ELEMENT FOR ANY MATRIX STORAGE MODE            DCPY 380
C                                                                       DCPY 390
      CALL LOC(J,J,IJ,N,N,MS)                                           DCPY 400
C                                                                       DCPY 410
C        MOVE DIAGONAL ELEMENT TO VECTOR R                              DCPY 420
C                                                                       DCPY 430
    3 R(J)=A(IJ)                                                        DCPY 440
      RETURN                                                            DCPY 450
      END                                                               DCPY 460
```

Subroutine XCPY

```
C
C     ..................................................................
C
C        SUBROUTINE XCPY
C
C        PURPOSE
C           COPY A PORTION OF A MATRIX
C
C        USAGE
C           CALL XCPY(A,R,L,K,NR,MR,NA,MA,MS)
C
C        DESCRIPTION OF PARAMETERS
C           A  - NAME OF INPUT MATRIX
C           R  - NAME OF OUTPUT MATRIX
C           L  - ROW OF A WHERE FIRST ELEMENT OF R CAN BE FOUND
C           K  - COLUMN OF A WHERE FIRST ELEMENT OF R CAN BE FOUND
C           NR - NUMBER OF ROWS TO BE COPIED INTO R
C           MR - NUMBER OF COLUMNS TO BE COPIED INTO R
C           NA - NUMBER OF ROWS IN A
C           MA - NUMBER OF COLUMNS IN A
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                 0 - GENERAL
C                 1 - SYMMETRIC
C                 2 - DIAGONAL
C
C        REMARKS
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A
C           MATRIX R IS ALWAYS A GENERAL MATRIX
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           MATRIX R IS FORMED BY COPYING A PORTION OF MATRIX A. THIS
C           IS DONE BY EXTRACTING NR ROWS AND MR COLUMNS OF MATRIX A,
C           STARTING WITH ELEMENT AT ROW L, COLUMN K
C
C     ..................................................................
C
      SUBROUTINE XCPY(A,R,L,K,NR,MR,NA,MA,MS)
      DIMENSION A(1),R(1)
C
C        INITIALIZE
C
      IR=0
      L2=L+NR-1
      K2=K+MR-1
C
      DO 5 J=K,K2
      DO 5 I=L,L2
      IR=IR+1
      R(IR)=0.0
C
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE
C
      CALL LOC(I,J,IA,NA,MA,MS)
C
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX
C
      IF(IA) 4,5,4
    4 R(IR)=A(IA)
    5 CONTINUE
      RETURN
      END
```

Subroutine MSTR

```
C
C     ..................................................................
C
C        SUBROUTINE MSTR
C
C        PURPOSE
C           CHANGE STORAGE MODE OF A MATRIX
C
C        USAGE
C           CALL MSTR(A,R,N,MSA,MSR)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS AND COLUMNS IN A AND R
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                 0 - GENERAL
C                 1 - SYMMETRIC
C                 2 - DIAGONAL
C           MSR - SAME AS MSA EXCEPT FOR MATRIX R
C
C        REMARKS
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A
C           MATRIX A MUST BE A SQUARE MATRIX
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           MATRIX A IS RESTRUCTURED TO FORM MATRIX R.
C            MSA MSR
C             0   0  MATRIX A IS MOVED TO MATRIX R
C             0   1  THE UPPER TRIANGLE ELEMENTS OF A GENERAL MATRIX
C                    ARE USED TO FORM A SYMMETRIC MATRIX
C             0   2  THE DIAGONAL ELEMENTS OF A GENERAL MATRIX ARE USED
C                    TO FORM A DIAGONAL MATRIX
C             1   0  A SYMMETRIC MATRIX IS EXPANDED TO FORM A GENERAL
C                    MATRIX
C             1   1  MATRIX A IS MOVED TO MATRIX R
C             1   2  THE DIAGONAL ELEMENTS OF A SYMMETRIC MATRIX ARE
C                    USED TO FORM A DIAGONAL MATRIX
C             2   0  A DIAGONAL MATRIX IS EXPANDED BY INSERTING MISSING
C                    ZERO ELEMENTS TO FORM A GENERAL MATRIX
C             2   1  A DIAGONAL MATRIX IS EXPANDED BY INSERTING MISSING
C                    ZERO ELEMENTS TO FORM A SYMMETRIC MATRIX
C             2   2  MATRIX A IS MOVED TO MATRIX R
C
C     ..................................................................
C
      SUBROUTINE MSTR(A,R,N,MSA,MSR)
      DIMENSION A(1),R(1)
C
      DO 20 I=1,N
      DO 20 J=1,N
C
C        IF R IS GENERAL, FORM ELEMENT
C
      IF(MSR) 5,10,5
C
C        IF IN LOWER TRIANGLE OF SYMMETRIC OR DIAGONAL R, BYPASS
C
    5 IF(I-J) 10,10,20
   10 CALL LOC(I,J,IR,N,N,MSR)
C
C        IF IN UPPER AND OFF DIAGONAL  OF DIAGONAL R, BYPASS
C
      IF(IR) 20,20,15
C
C        OTHERWISE, FORM R(I,J)
C
   15 R(IR)=0.0
      CALL LOC(I,J,IA,N,N,MSA)
C
C        IF THERE IS NO A(I,J), LEAVE R(I,J) AT C.0
C
      IF(IA) 20,20,18
   18 R(IR)=A(IA)
   20 CONTINUE
      RETURN
      END
```

Subroutine LOC

```
C
C     ..................................................................
C
C        SUBROUTINE LOC
C
C        PURPOSE
C           COMPUTE A VECTOR SUBSCRIPT FOR AN ELEMENT IN A MATRIX OF
C           SPECIFIED STORAGE MODE
C
C        USAGE
C           CALL LOC (I,J,IR,N,M,MS)
C
C        DESCRIPTION OF PARAMETERS
C           I   - ROW NUMBER OF ELEMENT
C           J   - COLUMN NUMBER  OF ELEMENT
C           IR  - RESULTANT VECTOR SUBSCRIPT
C           N   - NUMBER OF ROWS IN MATRIX
C           M   - NUMBER OF COLUMNS IN MATRIX
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX
C                 0 - GENERAL
C                 1 - SYMMETRIC
C                 2 - DIAGONAL
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           MS=0   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*M ELEMENTS
C                  IN STORAGE (GENERAL MATRIX)
C           MS=1   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N*(N+1)/2 IN
C                  STORAGE (UPPER TRIANGLE OF SYMMETRIC MATRIX). IF
C                  ELEMENT IS IN LOWER TRIANGULAR PORTION, SUBSCRIPT IS
C                  CORRESPONDING ELEMENT IN UPPER TRIANGLE.
C           MS=2   SUBSCRIPT IS COMPUTED FOR A MATRIX WITH N ELEMENTS
C                  IN STORAGE (DIAGONAL ELEMENTS OF DIAGONAL MATRIX).
C                  IF ELEMENT IS NOT ON DIAGONAL (AND THEREFORE NOT IN
C                  STORAGE), IR IS SET TO ZERO.
C
C     ..................................................................
C
      SUBROUTINE LOC(I,J,IR,N,M,MS)
C
      IX=I
      JX=J
      IF(MS-1) 10,20,30
   10 IRX=N*(JX-1)+IX
      GO TO 36
   20 IF(IX-JX) 22,24,24
   22 IRX=IX+(JX*JX-JX)/2
      GO TO 36
   24 IRX=JX+(IX*IX-IX)/2
      GO TO 36
   30 IRX=0
      IF(IX-JX) 36,32,36
   32 IRX=IX
   36 IR=IRX
      RETURN
      END
```

Subroutine CONVT

```
C
C     ..................................................................
C
C        SUBROUTINE CONVT
C
C        PURPOSE
C           CONVERT NUMBERS FROM SINGLE PRECISION TO DOUBLE PRECISION
C           OR FROM DOUBLE PRECISION TO SINGLE PRECISION.
C
C        USAGE
C           CALL CONVT (N,M,MODE,S,D,MS)
C
C        DESCRIPTION OF PARAMETERS
C           N    - NUMBER OF ROWS IN MATRICES S AND D.
C           M    - NUMBER OF COLUMNS IN MATRICES S AND D.
C           MODE - CODE INDICATING TYPE OF CONVERSION
C                  1 - FROM SINGLE PRECISION TO DOUBLE PRECISION
C                  2 - FROM DOUBLE PRECISION TO SINGLE PRECISION
C           S    - IF MODE=1, THIS MATRIX CONTAINS SINGLE PRECISION
C                  NUMBERS AS INPUT.  IF MODE=2, IT CONTAINS SINGLE
C                  PRECISION NUMBERS AS OUTPUT.  THE SIZE OF MATRIX S
C                  IS N BY M.
C           D    - IF MODE=1, THIS MATRIX CONTAINS DOUBLE PRECISION
C                  NUMBERS AS OUTPUT.  IF MODE=2, IT CONTAINS DOUBLE
C                  PRECISION NUMBERS AS INPUT.  THE SIZE OF MATRIX D IS
C                  N BY M.
C           MS   - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX
C                  0 - GENERAL
C                  1 - SYMMETRIC
C                  2 - DIAGONAL
C
C        REMARKS
C           MATRIX D CANNOT BE IN THE SAME LOCATION AS MATRIX S.
C           MATRIX D MUST BE DEFINED BY A DOUBLE PRECISION STATEMENT IN
C           THE CALLING PROGRAM.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           ACCORDING TO THE TYPE OF CONVERSION INDICATED IN MODE, THIS
C           SUBROUTINE COPIES NUMBERS FROM MATRIX S TO MATRIX D OR FROM
C           MATRIX D TO MATRIX S.
C
C     ..................................................................
C
      SUBROUTINE CONVT (N,M,MODE,S,D,MS)
      DIMENSION S(1),D(1)
      DOUBLE PRECISION D
C
C        FIND STORAGE MODE OF MATRIX AND NUMBER OF DATA POINTS
C
      IF(MS-1) 2, 4, 6
    2 NM=N*M
      GO TO 8
    4 NM=((N+1)*N)/2
      GO TO 8
    6 NM=N
C
C        TEST TYPE OF CONVERSION
C
    8 IF(MODE-1) 10, 10, 20
C
C        SINGLE PRECISION TO DOUBLE PRECISION
C
   10 DO 15 L=1,NM
   15 D(L)=S(L)
      GO TO 30
C
C        DOUBLE PRECISION TO SINGLE PRECISION
C
   20 DO 25 L=1,NM
   25 S(L)=D(L)
C
   30 RETURN
      END
```

Subroutine GMADD

```
C
C     ..................................................................
C
C        SUBROUTINE ARRAY
C
C        PURPOSE
C           CONVERT DATA ARRAY FROM SINGLE TO DOUBLE DIMENSION OR VICE
C           VERSA.  THIS SUBROUTINE IS USED TO LINK THE USER PROGRAM
C           WHICH HAS DOUBLE DIMENSION ARRAYS AND THE SSP SUBROUTINES
C           WHICH OPERATE ON ARRAYS OF DATA IN A VECTOR FASHION.
C
C        USAGE
C           CALL ARRAY (MODE,I,J,N,M,S,D)
C
C        DESCRIPTION OF PARAMETERS
C           MODE - CODE INDICATING TYPE OF CONVERSION
C                     1 - FROM SINGLE TO DOUBLE DIMENSION
C                     2 - FROM DOUBLE TO SINGLE DIMENSION
C           I    - NUMBER OF ROWS IN ACTUAL DATA MATRIX
C           J    - NUMBER OF COLUMNS IN ACTUAL DATA MATRIX
C           N    - NUMBER OF ROWS SPECIFIED FOR THE MATRIX D IN
C                  DIMENSION STATEMENT
C           M    - NUMBER OF COLUMNS SPECIFIED FOR THE MATRIX D IN
C                  DIMENSION STATEMENT
C           S    - IF MODE=1, THIS VECTOR IS INPUT WHICH CONTAINS THE
C                  ELEMENTS OF A DATA MATRIX OF SIZE I BY J. COLUMN I+1
C                  OF DATA MATRIX FOLLOWS COLUMN I, ETC. IF MODE=2,
C                  THIS VECTOR IS OUTPUT REPRESENTING A DATA MATRIX OF
C                  SIZE I BY J CONTAINING ITS COLUMNS CONSECUTIVELY.
C                  THE LENGTH OF S IS IJ, WHERE IJ=I*J.
C           D    - IF MODE=1, THIS MATRIX OF SIZE N BY M IS OUTPUT,
C                  CONTAINING A DATA MATRIX OF SIZE I BY J IN THE FIRST
C                  I ROWS AND J COLUMNS. IF MODE=2, THIS N BY M MATRIX
C                  IS INPUT CONTAINING A DATA MATRIX OF SIZE I BY J IN
C                  THE FIRST I ROWS AND J COLUMNS.
C
C        REMARKS
C           VECTOR S CAN BE IN THE SAME LOCATION AS MATRIX D.  VECTOR S
C           IS REFERRED AS A MATRIX IN OTHER SSP ROUTINES, SINCE IT
C           CONTAINS A DATA MATRIX.
C           THIS SUBROUTINE CONVERTS ONLY GENERAL DATA MATRICES (STORAGE
C           MODE OF 0).
C
C        SUBROUTINES AND FUNCTION SUBROUTINES REQUIRED
C           NONE
C
C        METHOD
C           REFER TO THE DISCUSSION ON VARIABLE DATA SIZE IN THE SECTION
C           DESCRIBING OVERALL RULES FOR USAGE IN THIS MANUAL.
C
C     ..................................................................
C
      SUBROUTINE ARRAY (MODE,I,J,N,M,S,D)
      DIMENSION S(1),D(1)
C
      NI=N-I
C
C        TEST TYPE OF CONVERSION
C
      IF(MODE-1) 100, 100, 120
C
C        CONVERT FROM SINGLE TO DOUBLE DIMENSION
C
  100 IJ=I*J+1
      NM=N*J+1
      DO 110 K=1,J
      NM=NM-NI
      DO 110 L=1,I
      IJ=IJ-1
      NM=NM-1
  110 D(NM)=S(IJ)
      GO TO 140
C
C        CONVERT FROM DOUBLE TO SINGLE DIMENSION
C
  120 IJ=0
      NM=0
      DO 130 K=1,J
      DO 125 L=1,I
      IJ=IJ+1
      NM=NM+1
  125 S(IJ)=D(NM)
  130 NM=NM+NI
C
  140 RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE GMADD
C
C        PURPOSE
C           ADD TWO GENERAL MATRICES TO FORM RESULTANT GENERAL MATRIX
C
C        USAGE
C           CALL GMADD(A,B,R,N,M)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF FIRST INPUT MATRIX
C           B - NAME OF SECOND INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS IN A,B,R
C           M - NUMBER OF COLUMNS IN A,B,R
C
C        REMARKS
C           ALL MATRICES MUST BE STORED AS GENERAL MATRICES
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           ADDITION IS PERFORMED ELEMENT BY ELEMENT
C
C     ..................................................................
C
      SUBROUTINE GMADD(A,B,R,N,M)
      DIMENSION A(1),B(1),R(1)
C
C        CALCULATE NUMBER OF ELEMENTS
C
      NM=N*M
C
C        ADD MATRICES
C
      DO 10 I=1,NM
   10 R(I)=A(I)+B(I)
      RETURN
      END
```

Subroutine GMSUB

```
C                                                                       GMSU  10
C     ...................................................................GMSU  20
C                                                                       GMSU  30
C        SUBROUTINE GMSUB                                               GMSU  40
C                                                                       GMSU  50
C        PURPOSE                                                        GMSU  60
C           SUBTRACT ONE GENERAL MATRIX FROM ANOTHER TO FORM RESULTANT  GMSU  70
C           MATRIX                                                      GMSU  80
C                                                                       GMSU  90
C        USAGE                                                          GMSU 100
C           CALL GMSUB(A,B,R,N,M)                                       GMSU 110
C                                                                       GMSU 120
C        DESCRIPTION OF PARAMETERS                                      GMSU 130
C           A - NAME OF FIRST INPUT MATRIX                              GMSU 140
C           B - NAME OF SECOND INPUT MATRIX                             GMSU 150
C           R - NAME OF OUTPUT MATRIX                                   GMSU 160
C           N - NUMBER OF ROWS IN A,B,R                                 GMSU 170
C           M - NUMBER OF COLUMNS IN A,B,R                              GMSU 180
C                                                                       GMSU 190
C        REMARKS                                                        GMSU 200
C           ALL MATRICES MUST BE STORED AS GENERAL MATRICES             GMSU 210
C                                                                       GMSU 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  GMSU 230
C           NONE                                                        GMSU 240
C                                                                       GMSU 250
C        METHOD                                                         GMSU 260
C           MATRIX B ELEMENTS ARE SUBTRACTED FROM CORRESPONDING MATRIX AGMSU 270
C           ELEMENTS                                                    GMSU 280
C                                                                       GMSU 290
C     ...................................................................GMSU 300
C                                                                       GMSU 310
      SUBROUTINE GMSUB(A,B,R,N,M)                                       GMSU 320
      DIMENSION A(1),B(1),R(1)                                          GMSU 330
C                                                                       GMSU 340
C        CALCULATE NUMBER OF ELEMENTS                                   GMSU 350
C                                                                       GMSU 360
      NM=N*M                                                            GMSU 370
C                                                                       GMSU 380
C        SUBTRACT MATRICES                                              GMSU 390
C                                                                       GMSU 400
      DO 10 I=1,NM                                                      GMSU 410
   10 R(I)=A(I)-B(I)                                                    GMSU 420
      RETURN                                                            GMSU 430
      END                                                               GMSU 440
```

Subroutine GMPRD

```
C                                                                       GMPR  10
C     ...................................................................GMPR  20
C                                                                       GMPR  30
C        SUBROUTINE GMPRD                                               GMPR  40
C                                                                       GMPR  50
C        PURPOSE                                                        GMPR  60
C           MULTIPLY TWO GENERAL MATRICES TO FORM A RESULTANT GENERAL   GMPR  70
C           MATRIX                                                      GMPR  80
C                                                                       GMPR  90
C        USAGE                                                          GMPR 100
C           CALL GMPRD(A,B,R,N,M,L)                                     GMPR 110
C                                                                       GMPR 120
C        DESCRIPTION OF PARAMETERS                                      GMPR 130
C           A - NAME OF FIRST INPUT MATRIX                              GMPR 140
C           B - NAME OF SECOND INPUT MATRIX                             GMPR 150
C           R - NAME OF OUTPUT MATRIX                                   GMPR 160
C           N - NUMBER OF ROWS IN A                                     GMPR 170
C           M - NUMBER OF COLUMNS IN A AND ROWS IN B                    GMPR 180
C           L - NUMBER OF COLUMNS IN B                                  GMPR 190
C                                                                       GMPR 200
C        REMARKS                                                        GMPR 210
C           ALL MATRICES MUST BE STORED AS GENERAL MATRICES             GMPR 220
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         GMPR 230
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX B         GMPR 240
C           NUMBER OF COLUMNS OF MATRIX A MUST BE EQUAL TO NUMBER OF ROWGMPR 250
C           OF MATRIX B                                                 GMPR 260
C                                                                       GMPR 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  GMPR 280
C           NONE                                                        GMPR 290
C                                                                       GMPR 300
C        METHOD                                                         GMPR 310
C           THE M BY L MATRIX B IS PREMULTIPLIED BY THE N BY M MATRIX A GMPR 320
C           AND THE RESULT IS STORED IN THE N BY L MATRIX R.            GMPR 330
C                                                                       GMPR 340
C     ...................................................................GMPR 350
C                                                                       GMPR 360
      SUBROUTINE GMPRD(A,B,R,N,M,L)                                     GMPR 370
      DIMENSION A(1),B(1),R(1)                                          GMPR 380
C                                                                       GMPR 390
      IR=0                                                              GMPR 400
      IK=-M                                                             GMPR 410
      DO 10 K=1,L                                                       GMPR 420
      IK=IK+M                                                           GMPR 430
      DO 10 J=1,N                                                       GMPR 440
      IR=IR+1                                                           GMPR 450
      JI=J-N                                                            GMPR 460
      IB=IK                                                             GMPR 470
      R(IR)=0                                                           GMPR 480
      DO 10 I=1,M                                                       GMPR 490
      JI=JI+N                                                           GMPR 500
      IB=IB+1                                                           GMPR 510
   10 R(IR)=R(IR)+A(JI)*B(IB)                                           GMPR 520
      RETURN                                                            GMPR 530
      END                                                               GMPR 540
```

Subroutine GMTRA

```
C                                                                       GMTR  10
C     ..................................................................GMTR  20
C                                                                       GMTR  30
C        SUBROUTINE GMTRA                                               GMTR  40
C                                                                       GMTR  50
C        PURPOSE                                                        GMTR  60
C           TRANSPOSE A GENERAL MATRIX                                  GMTR  70
C                                                                       GMTR  80
C        USAGE                                                          GMTR  90
C           CALL GMTRA(A,R,N,M)                                         GMTR 100
C                                                                       GMTR 110
C        DESCRIPTION OF PARAMETERS                                      GMTR 120
C           A - NAME OF MATRIX TO BE TRANSPOSED                         GMTR 130
C           R - NAME OF RESULTANT MATRIX                                GMTR 140
C           N - NUMBER OF ROWS OF A AND COLUMNS OF R                    GMTR 150
C           M - NUMBER OF COLUMNS OF A AND ROWS OF R                    GMTR 160
C                                                                       GMTR 170
C        REMARKS                                                        GMTR 180
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         GMTR 190
C           MATRICES A AND R MUST BE STORED AS GENERAL MATRICES         GMTR 200
C                                                                       GMTR 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  GMTR 220
C           NONE                                                        GMTR 230
C                                                                       GMTR 240
C        METHOD                                                         GMTR 250
C           TRANSPOSE N BY M MATRIX A TO FORM M BY N MATRIX R           GMTR 260
C                                                                       GMTR 270
C     ..................................................................GMTR 280
C                                                                       GMTR 290
      SUBROUTINE GMTRA(A,R,N,M)                                         GMTR 300
      DIMENSION A(1),R(1)                                               GMTR 310
C                                                                       GMTR 320
      IR=0                                                              GMTR 330
      DO 10 I=1,N                                                       GMTR 340
      IJ=I-N                                                            GMTR 350
      DO 10 J=1,M                                                       GMTR 360
      IJ=IJ+N                                                           GMTR 370
      IR=IR+1                                                           GMTR 380
   10 R(IR)=A(IJ)                                                       GMTR 390
      RETURN                                                            GMTR 400
      END                                                               GMTR 410
```

Subroutine GTPRD

```
C                                                                       GTPR  10
C     ..................................................................GTPR  20
C                                                                       GTPR  30
C        SUBROUTINE GTPRD                                               GTPR  40
C                                                                       GTPR  50
C        PURPOSE                                                        GTPR  60
C           PREMULTIPLY A GENERAL MATRIX BY THE TRANSPOSE OF ANOTHER    GTPR  70
C           GENERAL MATRIX                                              GTPR  80
C                                                                       GTPR  90
C        USAGE                                                          GTPR 100
C           CALL GTPRD(A,B,R,N,M,L)                                     GTPR 110
C                                                                       GTPR 120
C        DESCRIPTION OF PARAMETERS                                      GTPR 130
C           A - NAME OF FIRST INPUT MATRIX                              GTPR 140
C           B - NAME OF SECOND INPUT MATRIX                             GTPR 150
C           R - NAME OF OUTPUT MATRIX                                   GTPR 160
C           N - NUMBER OF ROWS IN A AND B                               GTPR 170
C           M - NUMBER OF COLUMNS IN A AND ROWS IN R                    GTPR 180
C           L - NUMBER OF COLUMNS IN B AND R                            GTPR 190
C                                                                       GTPR 200
C        REMARKS                                                        GTPR 210
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         GTPR 220
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX B         GTPR 230
C           ALL MATRICES MUST BE STORED AS GENERAL MATRICES             GTPR 240
C                                                                       GTPR 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  GTPR 260
C           NONE                                                        GTPR 270
C                                                                       GTPR 280
C        METHOD                                                         GTPR 290
C           MATRIX TRANSPOSE OF A IS NOT ACTUALLY CALCULATED. INSTEAD,  GTPR 300
C           ELEMENTS OF MATRIX A ARE TAKEN COLUMNWISE RATHER THAN       GTPR 310
C           ROWWISE FOR POSTMULTIPLICATION BY MATRIX B.                 GTPR 320
C                                                                       GTPR 330
C     ..................................................................GTPR 340
C                                                                       GTPR 350
      SUBROUTINE GTPRD(A,B,R,N,M,L)                                     GTPR 360
      DIMENSION A(1),B(1),R(1)                                          GTPR 370
C                                                                       GTPR 380
      IR=0                                                              GTPR 390
      IK=-N                                                             GTPR 400
      DO 10 K=1,L                                                       GTPR 410
      IJ=0                                                              GTPR 420
      IK=IK+N                                                           GTPR 430
      DO 10 J=1,M                                                       GTPR 440
      IB=IK                                                             GTPR 450
      IR=IR+1                                                           GTPR 460
      R(IR)=0                                                           GTPR 470
      DO 10 I=1,N                                                       GTPR 480
      IJ=IJ+1                                                           GTPR 490
      IB=IB+1                                                           GTPR 500
   10 R(IR)=R(IR)+A(IJ)*B(IB)                                           GTPR 510
      RETURN                                                            GTPR 520
      END                                                               GTPR 530
```

Subroutine MADD

```
C                                                                       MADD  10
C     ..................................................................MADD  20
C                                                                       MADD  30
C        SUBROUTINE MADD                                                MADD  40
C                                                                       MADD  50
C        PURPOSE                                                        MADD  60
C           ADD TWO MATRICES ELEMENT BY ELEMENT TO FORM RESULTANT       MADD  70
C           MATRIX                                                      MADD  80
C                                                                       MADD  90
C        USAGE                                                          MADD 100
C           CALL MADD(A,B,R,N,M,MSA,MSB)                                MADD 110
C                                                                       MADD 120
C        DESCRIPTION OF PARAMETERS                                      MADD 130
C           A - NAME OF INPUT MATRIX                                    MADD 140
C           B - NAME OF INPUT MATRIX                                    MADD 150
C           R - NAME OF OUTPUT MATRIX                                   MADD 160
C           N - NUMBER OF ROWS IN A,B,R                                 MADD 170
C           M - NUMBER OF COLUMNS IN A,B,R                              MADD 180
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         MADD 190
C                 0 - GENERAL                                           MADD 200
C                 1 - SYMMETRIC                                         MADD 210
C                 2 - DIAGONAL                                          MADD 220
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B                       MADD 230
C                                                                       MADD 240
C        REMARKS                                                        MADD 250
C           NONE                                                        MADD 260
C                                                                       MADD 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MADD 280
C           LOC                                                         MADD 290
C                                                                       MADD 300
C        METHOD                                                         MADD 310
C           STORAGE MODE OF OUTPUT MATRIX IS FIRST DETERMINED. ADDITION MADD 320
C           OF CORRESPONDING ELEMENTS IS THEN PERFORMED.                MADD 330
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT    MADD 340
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES               MADD 350
C                        A              B              R                MADD 360
C                    GENERAL        GENERAL        GENERAL             MADD 370
C                    GENERAL        SYMMETRIC      GENERAL             MADD 380
C                    GENERAL        DIAGONAL       GENERAL             MADD 390
C                    SYMMETRIC      GENERAL        GENERAL             MADD 400
C                    SYMMETRIC      SYMMETRIC      SYMMETRIC           MADD 410
C                    SYMMETRIC      DIAGONAL       SYMMETRIC           MADD 420
C                    DIAGONAL       GENERAL        GENERAL             MADD 430
C                    DIAGONAL       SYMMETRIC      SYMMETRIC           MADD 440
C                    DIAGONAL       DIAGONAL       DIAGONAL            MADD 450
C                                                                       MADD 460
C     ..................................................................MADD 470
C                                                                       MADD 480
      SUBROUTINE MADD(A,B,R,N,M,MSA,MSB)                                MADD 490
      DIMENSION A(1),B(1),R(1)                                          MADD 500
C                                                                       MADD 510
C        DETERMINE STORAGE MODE OF OUTPUT MATRIX                        MADD 520
C                                                                       MADD 530
      IF(MSA-MSB) 7,5,7                                                 MADD 540
    5 CALL LOC(N,M,NM,N,M,MSA)                                          MADD 550
      GO TO 100                                                         MADD 560
    7 MTEST=MSA*MSB                                                     MADD 570
      MSR=0                                                             MADD 580
      IF(MTEST) 20,20,10                                                MADD 590
   10 MSR=1                                                             MADD 600
   20 IF(MTEST-2) 35,35,30                                              MADD 610
   30 MSR=2                                                             MADD 620
C                                                                       MADD 630
C        LOCATE ELEMENTS AND PERFORM ADDITION                           MADD 640
C                                                                       MADD 650
   35 DO 90 J=1,M                                                       MADD 660
      DO 90 I=1,N                                                       MADD 670
      CALL LOC(I,J,IJR,N,M,MSR)                                         MADD 680
      IF(IJR) 40,90,40                                                  MADD 690
   40 CALL LOC(I,J,IJA,N,M,MSA)                                         MADD 700
      AEL=0.0                                                           MADD 710
      IF(IJA) 50,60,50                                                  MADD 720
   50 AEL=A(IJA)                                                        MADD 730
   60 CALL LOC(I,J,IJB,N,M,MSB)                                         MADD 740
      BEL=0.0                                                           MADD 750
      IF(IJB) 70,80,70                                                  MADD 760
   70 BEL=B(IJB)                                                        MADD 770
   80 R(IJR)=AEL+BEL                                                    MADD 780
   90 CONTINUE                                                          MADD 790
      RETURN                                                            MADD 800
C                                                                       MADD 810
C        ADD MATRICES FOR OTHER CASES                                   MADD 820
C                                                                       MADD 830
  100 DO 110 I=1,NM                                                     MADD 840
  110 R(I)=A(I)+B(I)                                                    MADD 850
      RETURN                                                            MADD 860
      END                                                               MADD 870
```

Subroutine MSUB

```
C                                                                       MSUB  10
C     ..................................................................MSUB  20
C                                                                       MSUB  30
C        SUBROUTINE MSUB                                                MSUB  40
C                                                                       MSUB  50
C        PURPOSE                                                        MSUB  60
C           SUBTRACT TWO MATRICES ELEMENT BY ELEMENT TO FORM RESULTANT  MSUB  70
C           MATRIX                                                      MSUB  80
C                                                                       MSUB  90
C        USAGE                                                          MSUB 100
C           CALL MSUB(A,B,R,N,M,MSA,MSB)                                MSUB 110
C                                                                       MSUB 120
C        DESCRIPTION OF PARAMETERS                                      MSUB 130
C           A - NAME OF INPUT MATRIX                                    MSUB 140
C           B - NAME OF INPUT MATRIX                                    MSUB 150
C           R - NAME OF OUTPUT MATRIX                                   MSUB 160
C           N - NUMBER OF ROWS IN A,B,R                                 MSUB 170
C           M - NUMBER OF COLUMNS IN A,B,R                              MSUB 180
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         MSUB 190
C                 0 - GENERAL                                           MSUB 200
C                 1 - SYMMETRIC                                         MSUB 210
C                 2 - DIAGONAL                                          MSUB 220
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B                       MSUB 230
C                                                                       MSUB 240
C        REMARKS                                                        MSUB 250
C           NONE                                                        MSUB 260
C                                                                       MSUB 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MSUB 280
C           LOC                                                         MSUB 290
C                                                                       MSUB 300
C        METHOD                                                         MSUB 310
C           STRUCTURE OF OUTPUT MATRIX IS FIRST DETERMINED. SUBTRACTION MSUB 320
C           OF MATRIX B ELEMENTS FROM CORRESPONDING MATRIX A ELEMENTS   MSUB 330
C           IS THEN PERFORMED.                                          MSUB 340
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT    MSUB 350
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES               MSUB 360
C                        A              B              R                MSUB 370
C                    GENERAL        GENERAL        GENERAL             MSUB 380
C                    GENERAL        SYMMETRIC      GENERAL             MSUB 390
C                    GENERAL        DIAGONAL       GENERAL             MSUB 400
C                    SYMMETRIC      GENERAL        GENERAL             MSUB 410
C                    SYMMETRIC      SYMMETRIC      SYMMETRIC           MSUB 420
C                    SYMMETRIC      DIAGONAL       SYMMETRIC           MSUB 430
C                    DIAGONAL       GENERAL        GENERAL             MSUB 440
C                    DIAGONAL       SYMMETRIC      SYMMETRIC           MSUB 450
C                    DIAGONAL       DIAGONAL       DIAGONAL            MSUB 460
C                                                                       MSUB 470
C     ..................................................................MSUB 480
C                                                                       MSUB 490
      SUBROUTINE MSUB(A,B,R,N,M,MSA,MSB)                                MSUB 500
      DIMENSION A(1),B(1),R(1)                                          MSUB 510
C                                                                       MSUB 520
C        DETERMINE STORAGE MODE OF OUTPUT MATRIX                        MSUB 530
C                                                                       MSUB 540
      IF(MSA-MSB) 7,5,7                                                 MSUB 550
    5 CALL LOC(N,M,NM,N,M,MSA)                                          MSUB 560
      GO TO 100                                                         MSUB 570
    7 MTEST=MSA*MSB                                                     MSUB 580
      MSR=0                                                             MSUB 590
      IF(MTEST) 20,20,10                                                MSUB 600
   10 MSR=1                                                             MSUB 610
   20 IF(MTEST-2) 35,35,30                                              MSUB 620
   30 MSR=2                                                             MSUB 630
C                                                                       MSUB 640
C        LOCATE ELEMENTS AND PERFORM SUBTRACTION                        MSUB 650
C                                                                       MSUB 660
   35 DO 90 J=1,M                                                       MSUB 670
      DO 90 I=1,N                                                       MSUB 680
      CALL LOC(I,J,IJR,N,M,MSR)                                         MSUB 690
      IF(IJR) 40,90,40                                                  MSUB 700
   40 CALL LOC(I,J,IJA,N,M,MSA)                                         MSUB 710
      AEL=0.0                                                           MSUB 720
      IF(IJA) 50,60,50                                                  MSUB 730
   50 AEL=A(IJA)                                                        MSUB 740
   60 CALL LOC(I,J,IJB,N,M,MSB)                                         MSUB 750
      BEL=0.0                                                           MSUB 760
      IF(IJB) 70,80,70                                                  MSUB 770
   70 BEL=B(IJB)                                                        MSUB 780
   80 R(IJR)=AEL-BEL                                                    MSUB 790
   90 CONTINUE                                                          MSUB 800
      RETURN                                                            MSUB 810
C                                                                       MSUB 820
C        SUBTRACT MATRICES FOR OTHER CASES                              MSUB 830
C                                                                       MSUB 840
  100 DO 110 I=1,NM                                                     MSUB 850
  110 R(I)=A(I)-B(I)                                                    MSUB 860
      RETURN                                                            MSUB 870
      END                                                               MSUB 880
```

Subroutine MPRD

```
C                                                                       MPRD  10
C     ..................................................................MPRD  20
C                                                                       MPRD  30
C        SUBROUTINE MPRD                                                MPRD  40
C                                                                       MPRD  50
C        PURPOSE                                                        MPRD  60
C           MULTIPLY TWO MATRICES TO FORM A RESULTANT MATRIX            MPRD  70
C                                                                       MPRD  80
C        USAGE                                                          MPRD  90
C           CALL MPRD(A,B,R,N,M,MSA,MSB,L)                              MPRD 100
C                                                                       MPRD 110
C        DESCRIPTION OF PARAMETERS                                      MPRD 120
C           A - NAME OF FIRST INPUT MATRIX                              MPRD 130
C           B - NAME OF SECOND INPUT MATRIX                             MPRD 140
C           R - NAME OF OUTPUT MATRIX                                   MPRD 150
C           N - NUMBER OF ROWS IN A AND R                               MPRD 160
C           M - NUMBER OF COLUMNS IN A AND ROWS IN B                    MPRD 170
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         MPRD 180
C                 0 - GENERAL                                           MPRD 190
C                 1 - SYMMETRIC                                         MPRD 200
C                 2 - DIAGONAL                                          MPRD 210
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B                       MPRD 220
C           L - NUMBER OF COLUMNS IN B AND R                            MPRD 230
C                                                                       MPRD 240
C        REMARKS                                                        MPRD 250
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRICES A OR B  MPRD 260
C           NUMBER OF COLUMNS OF MATRIX A MUST BE EQUAL TO NUMBER OF ROWMPRD 270
C           OF MATRIX B                                                 MPRD 280
C                                                                       MPRD 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MPRD 300
C           LOC                                                         MPRD 310
C                                                                       MPRD 320
C        METHOD                                                         MPRD 330
C           THE M BY L MATRIX B IS PREMULTIPLIED BY THE N BY M MATRIX A MPRD 340
C           AND THE RESULT IS STORED IN THE N BY L MATRIX R. THIS IS A  MPRD 350
C           ROW INTO COLUMN PRODUCT.                                    MPRD 360
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT    MPRD 370
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES               MPRD 380
C                        A                B                R           MPRD 390
C                   GENERAL          GENERAL          GENERAL           MPRD 400
C                   GENERAL          SYMMETRIC        GENERAL           MPRD 410
C                   GENERAL          DIAGONAL         GENERAL           MPRD 420
C                   SYMMETRIC        GENERAL          GENERAL           MPRD 430
C                   SYMMETRIC        SYMMETRIC        GENERAL           MPRD 440
C                   SYMMETRIC        DIAGONAL         GENERAL           MPRD 450
C                   DIAGONAL         GENERAL          GENERAL           MPRD 460
C                   DIAGONAL         SYMMETRIC        GENERAL           MPRD 470
C                   DIAGONAL         DIAGONAL         DIAGONAL          MPRD 480
C                                                                       MPRD 490
C     ..................................................................MPRD 500
C                                                                       MPRD 510
      SUBROUTINE MPRD(A,B,R,N,M,MSA,MSB,L)                              MPRD 520
      DIMENSION A(1),B(1),R(1)                                          MPRD 530
C                                                                       MPRD 540
C        SPECIAL CASE FOR DIAGONAL BY DIAGONAL                          MPRD 550
C                                                                       MPRD 560
      MS=MSA*10+MSB                                                     MPRD 570
      IF(MS-22) 30,10,30                                                MPRD 580
   10 DO 20 I=1,N                                                       MPRD 590
   20 R(I)=A(I)*B(I)                                                    MPRD 600
      RETURN                                                            MPRD 610
C                                                                       MPRD 620
C        ALL OTHER CASES                                                MPRD 630
C                                                                       MPRD 640
   30 IR=1                                                              MPRD 650
      DO 90 K=1,L                                                       MPRD 660
      DO 90 J=1,N                                                       MPRD 670
      R(IR)=0                                                           MPRD 680
      DO 80 I=1,M                                                       MPRD 690
      IF(MS) 40,60,40                                                   MPRD 700
   40 CALL LOC(J,I,IA,N,M,MSA)                                          MPRD 710
      CALL LOC(I,K,IB,M,L,MSB)                                          MPRD 720
      IF(IA) 50,80,50                                                   MPRD 730
   50 IF(IB) 70,80,70                                                   MPRD 740
   60 IA=N*(I-1)+J                                                      MPRD 750
      IB=M*(K-1)+I                                                      MPRD 760
   70 R(IR)=R(IR)+A(IA)*B(IB)                                           MPRD 770
   80 CONTINUE                                                          MPRD 780
   90 IR=IR+1                                                           MPRD 790
      RETURN                                                            MPRD 800
      END                                                               MPRD 810
```

Subroutine MTRA

```
C                                                                       MTRA  10
C     ..................................................................MTRA  20
C                                                                       MTRA  30
C        SUBROUTINE MTRA                                                MTRA  40
C                                                                       MTRA  50
C        PURPOSE                                                        MTRA  60
C           TRANSPOSE A MATRIX                                          MTRA  70
C                                                                       MTRA  80
C        USAGE                                                          MTRA  90
C           CALL MTRA(A,R,N,M,MS)                                       MTRA 100
C                                                                       MTRA 110
C        DESCRIPTION OF PARAMETERS                                      MTRA 120
C           A - NAME OF MATRIX TO BE TRANSPOSED                         MTRA 130
C           R - NAME OF OUTPUT MATRIX                                   MTRA 140
C           N - NUMBER OF ROWS OF A AND COLUMNS OF R                    MTRA 150
C           M - NUMBER OF COLUMNS OF A AND ROWS OF R                    MTRA 160
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R) MTRA 170
C                 0 - GENERAL                                           MTRA 180
C                 1 - SYMMETRIC                                         MTRA 190
C                 2 - DIAGONAL                                          MTRA 200
C                                                                       MTRA 210
C        REMARKS                                                        MTRA 220
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         MTRA 230
C                                                                       MTRA 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MTRA 250
C           MCPY                                                        MTRA 260
C                                                                       MTRA 270
C        METHOD                                                         MTRA 280
C           TRANSPOSE N BY M MATRIX A TO FORM M BY N MATRIX R BY MOVING MTRA 290
C           EACH ROW OF A INTO THE CORRESPONDING COLUMN OF R. IF MATRIX MTRA 300
C           A IS SYMMETRIC OR DIAGONAL, MATRIX R IS THE SAME AS A.      MTRA 310
C                                                                       MTRA 320
C     ..................................................................MTRA 330
C                                                                       MTRA 340
      SUBROUTINE MTRA(A,R,N,M,MS)                                       MTRA 350
      DIMENSION A(1),R(1)                                               MTRA 360
C                                                                       MTRA 370
C        IF MS IS 1 OR 2, COPY A                                        MTRA 380
C                                                                       MTRA 390
      IF(MS) 10,20,10                                                   MTRA 400
   10 CALL MCPY(A,R,N,N,MS)                                             MTRA 410
      RETURN                                                            MTRA 420
C                                                                       MTRA 430
C        TRANSPOSE GENERAL MATRIX                                       MTRA 440
C                                                                       MTRA 450
   20 IR=0                                                              MTRA 460
      DO 30 I=1,N                                                       MTRA 470
      IJ=I-N                                                            MTRA 480
      DO 30 J=1,M                                                       MTRA 490
      IJ=IJ+N                                                           MTRA 500
      IR=IR+1                                                           MTRA 510
   30 R(IR)=A(IJ)                                                       MTRA 520
      RETURN                                                            MTRA 530
      END                                                               MTRA 540
```

Subroutine TPRD

```
C                                                                       TPRD  10
C     ..................................................................TPRD  20
C                                                                       TPRD  30
C        SUBROUTINE TPRD                                                TPRD  40
C                                                                       TPRD  50
C        PURPOSE                                                        TPRD  60
C           TRANSPOSE A MATRIX AND POSTMULTIPLY BY ANOTHER TO FORM      TPRD  70
C           A RESULTANT MATRIX                                          TPRD  80
C                                                                       TPRD  90
C        USAGE                                                          TPRD 100
C           CALL TPRD(A,B,R,N,M,MSA,MSB,L)                              TPRD 110
C                                                                       TPRD 120
C        DESCRIPTION OF PARAMETERS                                      TPRD 130
C           A - NAME OF FIRST INPUT MATRIX                              TPRD 140
C           B - NAME OF SECOND INPUT MATRIX                             TPRD 150
C           R - NAME OF OUTPUT MATRIX                                   TPRD 160
C           N - NUMBER OF ROWS IN A AND B                               TPRD 170
C           M - NUMBER OF COLUMNS IN A AND ROWS IN R                    TPRD 180
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         TPRD 190
C                 0 - GENERAL                                           TPRD 200
C                 1 - SYMMETRIC                                         TPRD 210
C                 2 - DIAGONAL                                          TPRD 220
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B                       TPRD 230
C           L - NUMBER OF COLUMNS IN B AND R                            TPRD 240
C                                                                       TPRD 250
C        REMARKS                                                        TPRD 260
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRICES A OR B  TPRD 270
C                                                                       TPRD 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TPRD 290
C           LOC                                                         TPRD 300
C                                                                       TPRD 310
C        METHOD                                                         TPRD 320
C           MATRIX TRANSPOSE OF A IS NOT ACTUALLY CALCULATED. INSTEAD,  TPRD 330
C           ELEMENTS IN MATRIX A ARE TAKEN COLUMNWISE RATHER THAN       TPRD 340
C           ROWWISE FOR MULTIPLICATION BY MATRIX B.                     TPRD 350
C           THE FOLLOWING TABLE SHOWS THE STORAGE MODE OF THE OUTPUT    TPRD 360
C           MATRIX FOR ALL COMBINATIONS OF INPUT MATRICES               TPRD 370
C                        A              B              R                TPRD 380
C                     GENERAL        GENERAL        GENERAL             TPRD 390
C                     GENERAL        SYMMETRIC      GENERAL             TPRD 400
C                     GENERAL        DIAGONAL       GENERAL             TPRD 410
C                     SYMMETRIC      GENERAL        GENERAL             TPRD 420
C                     SYMMETRIC      SYMMETRIC      GENERAL             TPRD 430
C                     SYMMETRIC      DIAGONAL       GENERAL             TPRD 440
C                     DIAGONAL       GENERAL        GENERAL             TPRD 450
C                     DIAGONAL       SYMMETRIC      GENERAL             TPRD 460
C                     DIAGONAL       DIAGONAL       DIAGONAL            TPRD 470
C                                                                       TPRD 480
C     ..................................................................TPRD 490
C                                                                       TPRD 500
      SUBROUTINE TPRD(A,B,R,N,M,MSA,MSB,L)                              TPRD 510
      DIMENSION A(1),B(1),R(1)                                          TPRD 520
C                                                                       TPRD 530
C        SPECIAL CASE FOR DIAGONAL BY DIAGONAL                          TPRD 540
C                                                                       TPRD 550
      MS=MSA*10+MSB                                                     TPRD 560
      IF(MS-22) 30,10,30                                                TPRD 570
   10 DO 20 I=1,N                                                       TPRD 580
   20 R(I)=A(I)*B(I)                                                    TPRD 590
      RETURN                                                            TPRD 600
C                                                                       TPRD 610
C        MULTIPLY TRANSPOSE OF A BY B                                   TPRD 620
C                                                                       TPRD 630
   30 IR=1                                                              TPRD 640
      DO 90 K=1,L                                                       TPRD 650
      DO 90 J=1,M                                                       TPRD 660
      R(IR)=0.0                                                         TPRD 670
      DO 80 I=1,N                                                       TPRD 680
      IF(MS) 40,60,40                                                   TPRD 690
   40 CALL LOC(I,J,IA,N,M,MSA)                                          TPRD 700
      CALL LOC(I,K,IB,N,L,MSB)                                          TPRD 710
      IF(IA) 50,80,50                                                   TPRD 720
   50 IF(IB) 70,80,70                                                   TPRD 730
   60 IA=N*(J-1)+I                                                      TPRD 740
      IB=N*(K-1)+I                                                      TPRD 750
   70 R(IR)=R(IR)+A(IA)*B(IB)                                           TPRD 760
   80 CONTINUE                                                          TPRD 770
   90 IR=IR+1                                                           TPRD 780
      RETURN                                                            TPRD 790
      END                                                               TPRD 800
```

Subroutine MATA

```
C                                                                       MATA  10
C     ..................................................................MATA  20
C                                                                       MATA  30
C        SUBROUTINE MATA                                                MATA  40
C                                                                       MATA  50
C        PURPOSE                                                        MATA  60
C           PREMULTIPLY A MATRIX BY ITS TRANSPOSE TO FORM A             MATA  70
C           SYMMETRIC MATRIX                                            MATA  80
C                                                                       MATA  90
C        USAGE                                                          MATA 100
C           CALL MATA(A,R,N,M,MS)                                       MATA 110
C                                                                       MATA 120
C        DESCRIPTION OF PARAMETERS                                      MATA 130
C           A  - NAME OF INPUT MATRIX                                   MATA 140
C           R  - NAME OF OUTPUT MATRIX                                  MATA 150
C           N  - NUMBER OF ROWS IN A                                    MATA 160
C           M  - NUMBER OF COLUMNS IN A. ALSO NUMBER OF ROWS AND        MATA 170
C                NUMBER OF COLUMNS OF R.                                MATA 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         MATA 190
C                 0 - GENERAL                                           MATA 200
C                 1 - SYMMETRIC                                         MATA 210
C                 2 - DIAGONAL                                          MATA 220
C                                                                       MATA 230
C        REMARKS                                                        MATA 240
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         MATA 250
C           MATRIX R IS ALWAYS A SYMMETRIC MATRIX WITH A STORAGE MODE=1 MATA 260
C                                                                       MATA 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MATA 280
C           LOC                                                         MATA 290
C                                                                       MATA 300
C        METHOD                                                         MATA 310
C           CALCULATION OF (A TRANSPOSE A) RESULTS IN A SYMMETRIC MATRIXMATA 320
C           REGARDLESS OF THE STORAGE MODE OF THE INPUT MATRIX. THE     MATA 330
C           ELEMENTS OF MATRIX A ARE NOT CHANGED.                       MATA 340
C                                                                       MATA 350
C     ..................................................................MATA 360
C                                                                       MATA 370
      SUBROUTINE MATA(A,R,N,M,MS)                                       MATA 380
      DIMENSION A(1),R(1)                                               MATA 390
C                                                                       MATA 400
      DO 60 K=1,M                                                       MATA 410
      KX=(K*K-K)/2                                                      MATA 420
      DO 60 J=1,M                                                       MATA 430
      IF(J-K) 10,10,60                                                  MATA 440
   10 IR=J+KX                                                           MATA 450
      R(IR)=0                                                           MATA 460
      DO 60 I=1,N                                                       MATA 470
      IF(MS) 20,40,20                                                   MATA 480
   20 CALL LOC(I,J,IA,N,M,MS)                                           MATA 490
      CALL LOC(I,K,IB,N,M,MS)                                           MATA 500
      IF(IA) 30,60,30                                                   MATA 510
   30 IF(IB) 50,60,50                                                   MATA 520
   40 IA=N*(J-1)+I                                                      MATA 530
      IB=N*(K-1)+I                                                      MATA 540
   50 R(IR)=R(IR)+A(IA)*A(IB)                                           MATA 550
   60 CONTINUE                                                          MATA 560
      RETURN                                                            MATA 570
      END                                                               MATA 580
```

Subroutine SADD

```
C                                                                       SADD  10
C     ..................................................................SADD  20
C                                                                       SADD  30
C        SUBROUTINE SADD                                                SADD  40
C                                                                       SADD  50
C        PURPOSE                                                        SADD  60
C           ADD A SCALAR TO EACH ELEMENT OF A MATRIX TO FORM A RESULTANTSADD  70
C           MATRIX                                                      SADD  80
C                                                                       SADD  90
C        USAGE                                                          SADD 100
C           CALL SADD(A,C,R,N,M,MS)                                     SADD 110
C                                                                       SADD 120
C        DESCRIPTION OF PARAMETERS                                      SADD 130
C           A - NAME OF INPUT MATRIX                                    SADD 140
C           C - SCALAR                                                  SADD 150
C           R - NAME OF OUTPUT MATRIX                                   SADD 160
C           N - NUMBER OF ROWS IN MATRIX A AND R                        SADD 170
C           M - NUMBER OF COLUMNS IN MATRIX A AND R                     SADD 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R) SADD 190
C                 0 - GENERAL                                           SADD 200
C                 1 - SYMMETRIC                                         SADD 210
C                 2 - DIAGONAL                                          SADD 220
C                                                                       SADD 230
C        REMARKS                                                        SADD 240
C           NONE                                                        SADD 250
C                                                                       SADD 260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SADD 270
C           LOC                                                         SADD 280
C                                                                       SADD 290
C        METHOD                                                         SADD 300
C           SCALAR IS ADDED TO EACH ELEMENT OF MATRIX                   SADD 310
C                                                                       SADD 320
C     ..................................................................SADD 330
C                                                                       SADD 340
      SUBROUTINE SADD(A,C,R,N,M,MS)                                     SADD 350
      DIMENSION A(1),R(1)                                               SADD 360
C                                                                       SADD 370
C        COMPUTE VECTOR LENGTH, IT                                      SADD 380
C                                                                       SADD 390
      CALL LOC(N,M,IT,N,M,MS)                                           SADD 400
C                                                                       SADD 410
C        ADD SCALAR                                                     SADD 420
C                                                                       SADD 430
      DO 1 I=1,IT                                                       SADD 440
    1 R(I)=A(I)+C                                                       SADD 450
      RETURN                                                            SADD 460
      END                                                               SADD 470
```

Subroutine SSUB

```
C                                                                       SSUB  10
C     ..................................................................SSUB  20
C                                                                       SSUB  30
C        SUBROUTINE SSUB                                                SSUB  40
C                                                                       SSUB  50
C        PURPOSE                                                        SSUB  60
C           SUBTRACT A SCALAR FROM EACH ELEMENT OF A MATRIX TO FORM A   SSUB  70
C           RESULTANT MATRIX                                            SSUB  80
C                                                                       SSUB  90
C        USAGE                                                          SSUB 100
C           CALL SSUB(A,C,R,N,M,MS)                                     SSUB 110
C                                                                       SSUB 120
C        DESCRIPTION OF PARAMETERS                                      SSUB 130
C           A - NAME OF INPUT MATRIX                                    SSUB 140
C           C - SCALAR                                                  SSUB 150
C           R - NAME OF OUTPUT MATRIX                                   SSUB 160
C           N - NUMBER OF ROWS IN MATRIX A AND R                        SSUB 170
C           M - NUMBER OF COLUMNS IN MATRIX A AND R                     SSUB 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R) SSUB 190
C                 0 - GENERAL                                           SSUB 200
C                 1 - SYMMETRIC                                         SSUB 210
C                 2 - DIAGONAL                                          SSUB 220
C                                                                       SSUB 230
C        REMARKS                                                        SSUB 240
C           NONE                                                        SSUB 250
C                                                                       SSUB 260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SSUB 270
C           LOC                                                         SSUB 280
C                                                                       SSUB 290
C        METHOD                                                         SSUB 300
C           SCALAR IS SUBTRACTED FROM EACH EACH ELEMENT OF MATRIX       SSUB 310
C                                                                       SSUB 320
C     ..................................................................SSUB 330
C                                                                       SSUB 340
      SUBROUTINE SSUB(A,C,R,N,M,MS)                                     SSUB 350
      DIMENSION A(1),R(1)                                               SSUB 360
C                                                                       SSUB 370
C        COMPUTE VECTOR LENGTH, IT                                      SSUB 380
C                                                                       SSUB 390
      CALL LOC(N,M,IT,N,M,MS)                                           SSUB 400
C                                                                       SSUB 410
C        SUBTRACT SCALAR                                                SSUB 420
C                                                                       SSUB 430
      DO 1 I=1,IT                                                       SSUB 440
    1 R(I)=A(I)-C                                                       SSUB 450
      RETURN                                                            SSUB 460
      END                                                               SSUB 470
```

Subroutine SMPY

```
C                                                                       SMPY  10
C     ...................................................................SMPY  20
C                                                                       SMPY  30
C        SUBROUTINE SMPY                                                SMPY  40
C                                                                       SMPY  50
C        PURPOSE                                                        SMPY  60
C           MULTIPLY EACH ELEMENT OF A MATRIX BY A SCALAR TO FORM A     SMPY  70
C           RESULTANT MATRIX                                            SMPY  80
C                                                                       SMPY  90
C        USAGE                                                          SMPY 100
C           CALL SMPY(A,C,R,N,M,MS)                                     SMPY 110
C                                                                       SMPY 120
C        DESCRIPTION OF PARAMETERS                                      SMPY 130
C           A - NAME OF INPUT MATRIX                                    SMPY 140
C           C - SCALAR                                                  SMPY 150
C           R - NAME OF OUTPUT MATRIX                                   SMPY 160
C           N - NUMBER OF ROWS IN MATRIX A AND R                        SMPY 170
C           M - NUMBER OF COLUMNS IN MATRIX A AND R                     SMPY 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R) SMPY 190
C                 0 - GENERAL                                           SMPY 200
C                 1 - SYMMETRIC                                         SMPY 210
C                 2 - DIAGONAL                                          SMPY 220
C                                                                       SMPY 230
C        REMARKS                                                        SMPY 240
C           NONE                                                        SMPY 250
C                                                                       SMPY 260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SMPY 270
C           LOC                                                         SMPY 280
C                                                                       SMPY 290
C        METHOD                                                         SMPY 300
C           SCALAR IS MULTIPLIED BY EACH ELEMENT OF MATRIX              SMPY 310
C                                                                       SMPY 320
C     ...................................................................SMPY 330
C                                                                       SMPY 340
      SUBROUTINE SMPY(A,C,R,N,M,MS)                                     SMPY 350
      DIMENSION A(1),R(1)                                               SMPY 360
C                                                                       SMPY 370
C        COMPUTE VECTOR LENGTH, IT                                      SMPY 380
C                                                                       SMPY 390
      CALL LOC(N,M,IT,N,M,MS)                                           SMPY 400
C                                                                       SMPY 410
C        MULTIPLY BY SCALAR                                             SMPY 420
C                                                                       SMPY 430
      DO 1 I=1,IT                                                       SMPY 440
    1 R(I)=A(I)*C                                                       SMPY 450
      RETURN                                                            SMPY 460
      END                                                               SMPY 470
```

Subroutine SDIV

```
C                                                                       SDIV  10
C     ...................................................................SDIV  20
C                                                                       SDIV  30
C        SUBROUTINE SDIV                                                SDIV  40
C                                                                       SDIV  50
C        PURPOSE                                                        SDIV  60
C           DIVIDE EACH ELEMENT OF A MATRIX BY A SCALAR TO FORM A       SDIV  70
C           RESULTANT MATRIX                                            SDIV  80
C                                                                       SDIV  90
C        USAGE                                                          SDIV 100
C           CALL SDIV(A,C,R,N,M,MS)                                     SDIV 110
C                                                                       SDIV 120
C        DESCRIPTION OF PARAMETERS                                      SDIV 130
C           A - NAME OF INPUT MATRIX                                    SDIV 140
C           C - SCALAR                                                  SDIV 150
C           R - NAME OF OUTPUT MATRIX                                   SDIV 160
C           N - NUMBER OF ROWS IN MATRIX A AND R                        SDIV 170
C           M - NUMBER OF COLUMNS IN MATRIX A AND R                     SDIV 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R) SDIV 190
C                 0 - GENERAL                                           SDIV 200
C                 1 - SYMMETRIC                                         SDIV 210
C                 2 - DIAGONAL                                          SDIV 220
C                                                                       SDIV 230
C        REMARKS                                                        SDIV 240
C           IF SCALAR IS ZERO, DIVISION IS PERFORMED ONLY ONCE TO CAUSE SDIV 250
C           FLOATING POINT OVERFLOW CONDITION                           SDIV 260
C                                                                       SDIV 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SDIV 280
C           LOC                                                         SDIV 290
C                                                                       SDIV 300
C        METHOD                                                         SDIV 310
C           EACH ELEMENT OF MATRIX IS DIVIDED BY SCALAR                 SDIV 320
C                                                                       SDIV 330
C     ...................................................................SDIV 340
C                                                                       SDIV 350
      SUBROUTINE SDIV(A,C,R,N,M,MS)                                     SDIV 360
      DIMENSION A(1),R(1)                                               SDIV 370
C                                                                       SDIV 380
C        COMPUTE VECTOR LENGTH, IT                                      SDIV 390
C                                                                       SDIV 400
      CALL LOC(N,M,IT,N,M,MS)                                           SDIV 410
C                                                                       SDIV 420
C        DIVIDE BY SCALAR (IF SCALAR IS ZERO, DIVIDE ONLY ONCE)         SDIV 430
C                                                                       SDIV 440
      IF(C) 2,1,2                                                       SDIV 450
    1 IT=1                                                              SDIV 460
    2 DO 3 I=1,IT                                                       SDIV 470
    3 R(I)=A(I)/C                                                       SDIV 480
      RETURN                                                            SDIV 490
      END                                                               SDIV 500
```

Subroutine SCLA

```
C                                                                       SCLA  10
C     ..................................................................SCLA  20
C                                                                       SCLA  30
C        SUBROUTINE SCLA                                                SCLA  40
C                                                                       SCLA  50
C        PURPOSE                                                        SCLA  60
C           SET EACH ELEMENT OF A MATRIX EQUAL TO A GIVEN SCALAR        SCLA  70
C                                                                       SCLA  80
C        USAGE                                                          SCLA  90
C           CALL SCLA (A,C,N,M,MS)                                      SCLA 100
C                                                                       SCLA 110
C        DESCRIPTION OF PARAMETERS                                      SCLA 120
C           A - NAME OF INPUT MATRIX                                    SCLA 130
C           C - SCALAR                                                  SCLA 140
C           N - NUMBER OF ROWS IN MATRIX A                              SCLA 150
C           M - NUMBER OF COLUMNS IN MATRIX A                           SCLA 160
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         SCLA 170
C                 0 - GENERAL                                           SCLA 180
C                 1 - SYMMETRIC                                         SCLA 190
C                 2 - DIAGONAL                                          SCLA 200
C                                                                       SCLA 210
C        REMARKS                                                        SCLA 220
C           NONE                                                        SCLA 230
C                                                                       SCLA 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SCLA 250
C           LOC                                                         SCLA 260
C                                                                       SCLA 270
C        METHOD                                                         SCLA 280
C           EACH ELEMENT OF MATRIX A IS REPLACED BY SCALAR C            SCLA 290
C                                                                       SCLA 300
C     ..................................................................SCLA 310
C                                                                       SCLA 320
      SUBROUTINE SCLA(A,C,N,M,MS)                                       SCLA 330
      DIMENSION A(1)                                                    SCLA 340
C                                                                       SCLA 350
C        COMPUTE VECTOR LENGTH, IT                                      SCLA 360
C                                                                       SCLA 370
      CALL LOC(N,M,IT,N,M,MS)                                           SCLA 380
C                                                                       SCLA 390
C        REPLACE BY SCALAR                                              SCLA 400
C                                                                       SCLA 410
      DO 1 I=1,IT                                                       SCLA 420
    1 A(I)=C                                                            SCLA 430
      RETURN                                                            SCLA 440
      END                                                               SCLA 450
```

Subroutine DCLA

```
C                                                                       DCLA  10
C     ..................................................................DCLA  20
C                                                                       DCLA  30
C        SUBROUTINE DCLA                                                DCLA  40
C                                                                       DCLA  50
C        PURPOSE                                                        DCLA  60
C           SET EACH DIAGONAL ELEMENT OF A MATRIX EQUAL TO A SCALAR     DCLA  70
C                                                                       DCLA  80
C        USAGE                                                          DCLA  90
C           CALL DCLA (A,C,N,MS)                                        DCLA 100
C                                                                       DCLA 110
C        DESCRIPTION OF PARAMETERS                                      DCLA 120
C           A - NAME OF INPUT MATRIX                                    DCLA 130
C           C - SCALAR                                                  DCLA 140
C           N - NUMBER OF ROWS AND COLUMNS IN MATRIX A                  DCLA 150
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         DCLA 160
C                 0 - GENERAL                                           DCLA 170
C                 1 - SYMMETRIC                                         DCLA 180
C                 2 - DIAGONAL                                          DCLA 190
C                                                                       DCLA 200
C        REMARKS                                                        DCLA 210
C           INPUT MATRIX MUST BE A SQUARE MATRIX                        DCLA 220
C                                                                       DCLA 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DCLA 240
C           LOC                                                         DCLA 250
C                                                                       DCLA 260
C        METHOD                                                         DCLA 270
C           EACH ELEMENT ON DIAGONAL OF MATRIX IS REPLACED BY SCALAR C  DCLA 280
C                                                                       DCLA 290
C     ..................................................................DCLA 300
C                                                                       DCLA 310
      SUBROUTINE DCLA(A,C,N,MS)                                         DCLA 320
      DIMENSION A(1)                                                    DCLA 330
C                                                                       DCLA 340
      DO 3 I=1,N                                                        DCLA 350
C                                                                       DCLA 360
C        LOCATE DIAGONAL ELEMENT FOR ANY MATRIX STORAGE MODE            DCLA 370
C                                                                       DCLA 380
      CALL LOC(I,I,ID,N,N,MS)                                           DCLA 390
C                                                                       DCLA 400
C        REPLACE DIAGONAL ELEMENTS                                      DCLA 410
C                                                                       DCLA 420
    3 A(ID)=C                                                           DCLA 430
      RETURN                                                            DCLA 440
      END                                                               DCLA 450
```

Subroutine RADD

```
C                                                                       RADD  10
C     ...................................................................RADD  20
C                                                                       RADD  30
C        SUBROUTINE RADD                                                RADD  40
C                                                                       RADD  50
C        PURPOSE                                                        RADD  60
C           ADD ROW OF ONE MATRIX TO ROW OF ANOTHER MATRIX              RADD  70
C                                                                       RADD  80
C        USAGE                                                          RADD  90
C           CALL RADD(A,IRA,R,IRR,N,M,MS,L)                             RADD 100
C                                                                       RADD 110
C        DESCRIPTION OF PARAMETERS                                      RADD 120
C           A   - NAME OF INPUT MATRIX                                  RADD 130
C           IRA - ROW IN MATRIX A TO BE ADDED TO ROW IRR OF MATRIX R    RADD 140
C           R   - NAME OF OUTPUT MATRIX                                 RADD 150
C           IRR - ROW IN MATRIX R WHERE SUMMATION IS DEVELOPED          RADD 160
C           N   - NUMBER OF ROWS IN A                                   RADD 170
C           M   - NUMBER OF COLUMNS IN A AND R                          RADD 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         RADD 190
C                 0 - GENERAL                                           RADD 200
C                 1 - SYMMETRIC                                         RADD 210
C                 2 - DIAGONAL                                          RADD 220
C           L   - NUMBER OF ROWS IN R                                   RADD 230
C                                                                       RADD 240
C        REMARKS                                                        RADD 250
C           MATRIX R MUST BE A GENERAL MATRIX                           RADD 260
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A UNLESS  RADD 270
C           A IS GENERAL                                                RADD 280
C                                                                       RADD 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RADD 300
C           LOC                                                         RADD 310
C                                                                       RADD 320
C        METHOD                                                         RADD 330
C           EACH ELEMENT OF ROW IRA OF MATRIX A IS ADDED TO             RADD 340
C           CORRESPONDING ELEMENT OF ROW IRR OF MATRIX R                RADD 350
C                                                                       RADD 360
C     ...................................................................RADD 370
C                                                                       RADD 380
      SUBROUTINE RADD(A,IRA,R,IRR,N,M,MS,L)                             RADD 390
      DIMENSION A(1),R(1)                                               RADD 400
C                                                                       RADD 410
      IR=IRR-L                                                          RADD 420
      DO 2 J=1,M                                                        RADD 430
      IR=IR+L                                                           RADD 440
C                                                                       RADD 450
C        LOCATE INPUT ELEMENT FOR ANY MATRIX STORAGE MODE               RADD 460
C                                                                       RADD 470
      CALL LOC(IRA,J,IA,N,M,MS)                                         RADD 480
C                                                                       RADD 490
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       RADD 500
C                                                                       RADD 510
      IF(IA) 1,2,1                                                      RADD 520
C                                                                       RADD 530
C        ADD ELEMENTS                                                   RADD 540
C                                                                       RADD 550
    1 R(IR)=R(IR)+A(IA)                                                 RADD 560
    2 CONTINUE                                                          RADD 570
      RETURN                                                            RADD 580
      END                                                               RADD 590
```

Subroutine CADD

```
C                                                                       CADD  10
C     ...................................................................CADD  20
C                                                                       CADD  30
C        SUBROUTINE CADD                                                CADD  40
C                                                                       CADD  50
C        PURPOSE                                                        CADD  60
C           ADD COLUMN OF ONE MATRIX TO COLUMN OF ANOTHER MATRIX        CADD  70
C                                                                       CADD  80
C        USAGE                                                          CADD  90
C           CALL CADD(A,ICA,R,ICR,N,M,MS,L)                             CADD 100
C                                                                       CADD 110
C        DESCRIPTION OF PARAMETERS                                      CADD 120
C           A   - NAME OF INPUT MATRIX                                  CADD 130
C           ICA - COLUMN IN MATRIX A TO BE ADDED TO COLUMN ICR OF R     CADD 140
C           R   - NAME OF OUTPUT MATRIX                                 CADD 150
C           ICR - COLUMN IN MATRIX R WHERE SUMMATION IS DEVELOPED       CADD 160
C           N   - NUMBER OF ROWS IN A AND R                             CADD 170
C           M   - NUMBER OF COLUMNS IN A                                CADD 180
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         CADD 190
C                 0 - GENERAL                                           CADD 200
C                 1 - SYMMETRIC                                         CADD 210
C                 2 - DIAGONAL                                          CADD 220
C           L   - NUMBER OF COLUMNS IN R                                CADD 230
C                                                                       CADD 240
C        REMARKS                                                        CADD 250
C           MATRIX R MUST BE A GENERAL MATRIX                           CADD 260
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A UNLESS  CADD 270
C           A IS GENERAL                                                CADD 280
C                                                                       CADD 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CADD 300
C           LOC                                                         CADD 310
C                                                                       CADD 320
C        METHOD                                                         CADD 330
C           EACH ELEMENT OF COLUMN ICA OF MATRIX A IS ADDED TO          CADD 340
C           CORRESPONDING ELEMENT OF COLUMN ICR OF MATRIX R             CADD 350
C                                                                       CADD 360
C     ...................................................................CADD 370
C                                                                       CADD 380
      SUBROUTINE CADD(A,ICA,R,ICR,N,M,MS,L)                             CADD 390
      DIMENSION A(1),R(1)                                               CADD 400
C                                                                       CADD 410
      IR=N*(ICR-1)                                                      CADD 420
      DO 2 I=1,N                                                        CADD 430
      IR=IR+1                                                           CADD 440
C                                                                       CADD 450
C        LOCATE INPUT ELEMENT FOR ANY MATRIX STORAGE MODE               CADD 460
C                                                                       CADD 470
      CALL LOC(I,ICA,IA,N,M,MS)                                         CADD 480
C                                                                       CADD 490
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       CADD 500
C                                                                       CADD 510
      IF(IA) 1,2,1                                                      CADD 520
C                                                                       CADD 530
C        ADD ELEMENTS                                                   CADD 540
C                                                                       CADD 550
    1 R(IR)=R(IR)+A(IA)                                                 CADD 560
    2 CONTINUE                                                          CADD 570
      RETURN                                                            CADD 580
      END                                                               CADD 590
```

Subroutine SRMA

```
C                                                                       SRMA  10
C     ...................................................................SRMA  20
C                                                                       SRMA  30
C        SUBROUTINE SRMA                                                SRMA  40
C                                                                       SRMA  50
C        PURPOSE                                                        SRMA  60
C           MULTIPLY ROW OF MATRIX BY A SCALAR AND ADD TO ANOTHER ROW   SRMA  70
C           OF THE SAME MATRIX                                          SRMA  80
C                                                                       SRMA  90
C        USAGE                                                          SRMA 100
C           CALL SRMA(A,C,N,M,LA,LB)                                    SRMA 110
C                                                                       SRMA 120
C        DESCRIPTION OF PARAMETERS                                      SRMA 130
C           A  - NAME OF MATRIX                                         SRMA 140
C           C  - SCALAR                                                 SRMA 150
C           N  - NUMBER OF ROWS IN A                                    SRMA 160
C           M  - NUMBER OF COLUMNS IN A                                 SRMA 170
C           LA - ROW IN A TO BE MULTIPLIED BY SCALAR                    SRMA 180
C           LB - ROW IN A TO WHICH PRODUCT IS ADDED                     SRMA 190
C                IF 0 IS SPECIFIED, PRODUCT REPLACES ELEMENTS IN ROW LA SRMA 200
C                                                                       SRMA 210
C        REMARKS                                                        SRMA 220
C           MATRIX A MUST BE A GENERAL MATRIX                           SRMA 230
C                                                                       SRMA 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SRMA 250
C           NONE                                                        SRMA 260
C                                                                       SRMA 270
C        METHOD                                                         SRMA 280
C           EACH ELEMENT OF ROW LA IS MULTIPLIED BY SCALAR C AND THE    SRMA 290
C           PRODUCT IS ADDED TO THE CORRESPONDING ELEMENT OF ROW LB.    SRMA 300
C           ROW LA REMAINS UNAFFECTED BY THE OPERATION.                 SRMA 310
C           IF PARAMETER LB CONTAINS ZERO, MULTIPLICATION BY THE SCALAR SRMA 320
C           IS PERFORMED AND THE PRODUCT REPLACES ELEMENTS IN ROW LA.   SRMA 330
C                                                                       SRMA 340
C     ...................................................................SRMA 350
C                                                                       SRMA 360
      SUBROUTINE SRMA(A,C,N,M,LA,LB)                                    SRMA 370
      DIMENSION A(1)                                                    SRMA 380
C                                                                       SRMA 390
      LAJ=LA-N                                                          SRMA 400
      LBJ=LB-N                                                          SRMA 410
      DO 3 J=1,M                                                        SRMA 420
C                                                                       SRMA 430
C        LOCATE ELEMENT IN BOTH ROWS                                    SRMA 440
C                                                                       SRMA 450
      LAJ=LAJ+N                                                         SRMA 460
      LBJ=LBJ+N                                                         SRMA 470
C                                                                       SRMA 480
C        CHECK LB FOR ZERO                                              SRMA 490
C                                                                       SRMA 500
      IF(LB) 1,2,1                                                      SRMA 510
C                                                                       SRMA 520
C        IF NOT, MULTIPLY BY CONSTANT AND ADD TO OTHER ROW              SRMA 530
C                                                                       SRMA 540
    1 A(LBJ)=A(LAJ)*C+A(LBJ)                                            SRMA 550
      GO TO 3                                                           SRMA 560
C                                                                       SRMA 570
C        OTHERWISE, MULTIPLY ROW BY CONSTANT                            SRMA 580
C                                                                       SRMA 590
    2 A(LAJ)=A(LAJ)*C                                                   SRMA 600
    3 CONTINUE                                                          SRMA 610
      RETURN                                                            SRMA 620
      END                                                               SRMA 630
```

Subroutine SCMA

```
C                                                                       SCMA  10
C     ...................................................................SCMA  20
C                                                                       SCMA  30
C        SUBROUTINE SCMA                                                SCMA  40
C                                                                       SCMA  50
C        PURPOSE                                                        SCMA  60
C           MULTIPLY COLUMN OF MATRIX BY A SCALAR AND ADD TO ANOTHER    SCMA  70
C           COLUMN OF THE SAME MATRIX                                   SCMA  80
C                                                                       SCMA  90
C        USAGE                                                          SCMA 100
C           CALL SCMA(A,C,N,LA,LB)                                      SCMA 110
C                                                                       SCMA 120
C        DESCRIPTION OF PARAMETERS                                      SCMA 130
C           A  - NAME OF MATRIX                                         SCMA 140
C           C  - SCALAR                                                 SCMA 150
C           N  - NUMBER OF ROWS IN A                                    SCMA 160
C           LA - COLUMN IN A TO BE MULTIPLIED BY SCALAR                 SCMA 170
C           LB - COLUMN IN A TO WHICH PRODUCT IS ADDED                  SCMA 180
C                IF 0 IS SPECIFIED, PRODUCT REPLACES ELEMENTS IN LA     SCMA 190
C                                                                       SCMA 200
C        REMARKS                                                        SCMA 210
C           MATRIX A MUST BE A GENERAL MATRIX                           SCMA 220
C                                                                       SCMA 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SCMA 240
C           NONE                                                        SCMA 250
C                                                                       SCMA 260
C        METHOD                                                         SCMA 270
C           EACH ELEMENT OF COLUMN LA IS MULTIPLIED BY SCALAR C AND THE SCMA 280
C           PRODUCT IS ADDED TO THE CORRESPONDING ELEMENT OF COLUMN LB. SCMA 290
C           COLUMN LA REMAINS UNAFFECTED BY THE OPERATION.              SCMA 300
C           IF PARAMETER LB CONTAINS ZERO, MULTIPLICATION BY THE SCALAR SCMA 310
C           IS PERFORMED AND THE PRODUCT REPLACES ELEMENTS IN LA.       SCMA 320
C                                                                       SCMA 330
C     ...................................................................SCMA 340
C                                                                       SCMA 350
      SUBROUTINE SCMA(A,C,N,LA,LB)                                      SCMA 360
      DIMENSION A(1)                                                    SCMA 370
C                                                                       SCMA 380
C        LOCATE STARTING POINT OF BOTH COLUMNS                          SCMA 390
C                                                                       SCMA 400
      ILA=N*(LA-1)                                                      SCMA 410
      ILB=N*(LB-1)                                                      SCMA 420
C                                                                       SCMA 430
      DO 3 I=1,N                                                        SCMA 440
      ILA=ILA+1                                                         SCMA 450
      ILB=ILB+1                                                         SCMA 460
C                                                                       SCMA 470
C        CHECK LB FOR ZERO                                              SCMA 480
C                                                                       SCMA 490
      IF(LB) 1,2,1                                                      SCMA 500
C                                                                       SCMA 510
C        IF NOT MULTIPLY BY CONSTANT AND ADD TO SECOND COLUMN           SCMA 520
C                                                                       SCMA 530
    1 A(ILB)=A(ILA)*C+A(ILB)                                            SCMA 540
      GO TO 3                                                           SCMA 550
C                                                                       SCMA 560
C        OTHERWISE, MULTIPLY COLUMN BY CONSTANT                         SCMA 570
C                                                                       SCMA 580
    2 A(ILA)=A(ILA)*C                                                   SCMA 590
    3 CONTINUE                                                          SCMA 600
      RETURN                                                            SCMA 610
      END                                                               SCMA 620
```

Subroutine RINT

```
C                                                                       RINT  10
C     ..................................................................RINT  20
C                                                                       RINT  30
C        SUBROUTINE RINT                                                RINT  40
C                                                                       RINT  50
C        PURPOSE                                                        RINT  60
C           INTERCHANGE TWO ROWS OF A MATRIX                            RINT  70
C                                                                       RINT  80
C        USAGE                                                          RINT  90
C           CALL RINT(A,N,M,LA,LB)                                      RINT 100
C                                                                       RINT 110
C        DESCRIPTION OF PARAMETERS                                      RINT 120
C           A  - NAME OF MATRIX                                         RINT 130
C           N  - NUMBER OF ROWS IN A                                    RINT 140
C           M  - NUMBER OF COLUMNS IN A                                 RINT 150
C           LA - ROW TO BE INTERCHANGED WITH ROW LB                     RINT 160
C           LB - ROW TO BE INTERCHANGED WITH ROW LA                     RINT 170
C                                                                       RINT 180
C        REMARKS                                                        RINT 190
C           MATRIX A MUST BE A GENERAL MATRIX                           RINT 200
C                                                                       RINT 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RINT 220
C           NONE                                                        RINT 230
C                                                                       RINT 240
C        METHOD                                                         RINT 250
C           EACH ELEMENT OF ROW LA IS INTERCHANGED WITH CORRESPONDING   RINT 260
C           ELEMENT OF ROW LB                                           RINT 270
C                                                                       RINT 280
C     ..................................................................RINT 290
      SUBROUTINE RINT(A,N,M,LA,LB)                                      RINT 300
      DIMENSION A(1)                                                    RINT 310
C                                                                       RINT 320
      LAJ=LA-N                                                          RINT 330
      LBJ=LB-N                                                          RINT 340
      DO 3 J=1,M                                                        RINT 350
C                                                                       RINT 360
C        LOCATE ELEMENTS IN BOTH ROWS                                   RINT 370
C                                                                       RINT 380
      LAJ=LAJ+N                                                         RINT 390
      LBJ=LBJ+N                                                         RINT 400
C                                                                       RINT 410
C        INTERCHANGE ELEMENTS                                           RINT 420
C                                                                       RINT 430
      SAVE=A(LAJ)                                                       RINT 440
      A(LAJ)=A(LBJ)                                                     RINT 450
    3 A(LBJ)=SAVE                                                       RINT 460
      RETURN                                                            RINT 470
      END                                                               RINT 480
```

Subroutine CINT

```
C                                                                       CINT  10
C     ..................................................................CINT  20
C                                                                       CINT  30
C        SUBROUTINE CINT                                                CINT  40
C                                                                       CINT  50
C        PURPOSE                                                        CINT  60
C           INTERCHANGE TWO COLUMNS OF A MATRIX                         CINT  70
C                                                                       CINT  80
C        USAGE                                                          CINT  90
C           CALL CINT(A,N,LA,LB)                                        CINT 100
C                                                                       CINT 110
C        DESCRIPTION OF PARAMETERS                                      CINT 120
C           A  - NAME OF MATRIX                                         CINT 130
C           N  - NUMBER OF ROWS IN A                                    CINT 140
C           LA - COLUMN TO BE INTERCHANGED WITH COLUMN LB               CINT 150
C           LB - COLUMN TO BE INTERCHANGED WITH COLUMN LA               CINT 160
C                                                                       CINT 170
C        REMARKS                                                        CINT 180
C           MATRIX A MUST BE A GENERAL MATRIX                           CINT 190
C                                                                       CINT 200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CINT 210
C           NONE                                                        CINT 220
C                                                                       CINT 230
C        METHOD                                                         CINT 240
C           EACH ELEMENT OF COLUMN LA IS INTERCHANGED WITH CORRESPONDINGCINT 250
C           ELEMENT OF COLUMN LB                                        CINT 260
C                                                                       CINT 270
C     ..................................................................CINT 280
      SUBROUTINE CINT(A,N,LA,LB)                                        CINT 290
      DIMENSION A(1)                                                    CINT 300
C                                                                       CINT 310
C        LOCATE STARTING POINT OF BOTH COLUMNS                          CINT 320
C                                                                       CINT 330
      ILA=N*(LA-1)                                                      CINT 340
      ILB=N*(LB-1)                                                      CINT 350
C                                                                       CINT 360
      DO 3 I=1,N                                                        CINT 370
      ILA=ILA+1                                                         CINT 380
      ILB=ILB+1                                                         CINT 390
C                                                                       CINT 400
C        INTERCHANGE ELEMENTS                                           CINT 410
C                                                                       CINT 420
      SAVE=A(ILA)                                                       CINT 430
      A(ILA)=A(ILB)                                                     CINT 440
    3 A(ILB)=SAVE                                                       CINT 450
      RETURN                                                            CINT 460
      END                                                               CINT 470
                                                                        CINT 480
```

Subroutine RSUM

```
C
C     ..................................................................
C
C        SUBROUTINE RSUM
C
C        PURPOSE
C           SUM ELEMENTS OF EACH ROW TO FORM COLUMN VECTOR
C
C        USAGE
C           CALL RSUM (A,R,N,M,MS)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF INPUT MATRIX
C           R - NAME OF VECTOR OF LENGTH N
C           N - NUMBER OF ROWS IN A
C           M - NUMBER OF COLUMNS IN A
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                 0 - GENERAL
C                 1 - SYMMETRIC
C                 2 - DIAGONAL
C
C        REMARKS
C           VECTOR R CANNOT BE IN THE SAME LOCATION AS MATRIX A
C           UNLESS A IS GENERAL
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           ELEMENTS ARE SUMMED ACROSS EACH ROW INTO A CORRESPONDING
C           ELEMENT OF OUTPUT COLUMN VECTOR R
C
C     ..................................................................
      SUBROUTINE RSUM(A,R,N,M,MS)
      DIMENSION A(1),R(1)
C
      DO 3 I=1,N
C
C        CLEAR OUTPUT LOCATION
C
      R(I)=0.0
C
      DO 3 J=1,M
C
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE
C
      CALL LOC(I,J,IJ,N,M,MS)
C
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX
C
      IF(IJ) 2,3,2
C
C        ACCUMULATE IN OUTPUT VECTOR
C
    2 R(I)=R(I)+A(IJ)
    3 CONTINUE
      RETURN
      END
```

Subroutine CSUM

```
C
C     ..................................................................
C
C        SUBROUTINE CSUM
C
C        PURPOSE
C           SUM ELEMENTS OF EACH COLUMN TO FORM ROW VECTOR
C
C        USAGE
C           CALL CSUM(A,R,N,M,MS)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF INPUT MATRIX
C           R - NAME OF VECTOR OF LENGTH M
C           N - NUMBER OF ROWS IN A
C           M - NUMBER OF COLUMNS IN A
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                 0 - GENERAL
C                 1 - SYMMETRIC
C                 2 - DIAGONAL
C
C        REMARKS
C           VECTOR R CANNOT BE IN THE SAME LOCATION AS MATRIX A
C           UNLESS A IS GENERAL
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           ELEMENTS ARE SUMMED DOWN EACH COLUMN INTO A CORRESPONDING
C           ELEMENT OF OUTPUT ROW VECTOR R
C
C     ..................................................................
      SUBROUTINE CSUM(A,R,N,M,MS)
      DIMENSION A(1),R(1)
C
      DO 3 J=1,M
C
C        CLEAR OUTPUT LOCATION
C
      R(J)=0.0
C
      DO 3 I=1,N
C
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE
C
      CALL LOC(I,J,IJ,N,M,MS)
C
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX
C
      IF(IJ) 2,3,2
C
C        ACCUMULATE IN OUTPUT VECTOR
C
    2 R(J)=R(J)+A(IJ)
    3 CONTINUE
      RETURN
      END
```

Subroutine RTAB

The function of this subroutine is graphically displayed by Figure 7 (see description under "Method" in the program listing).

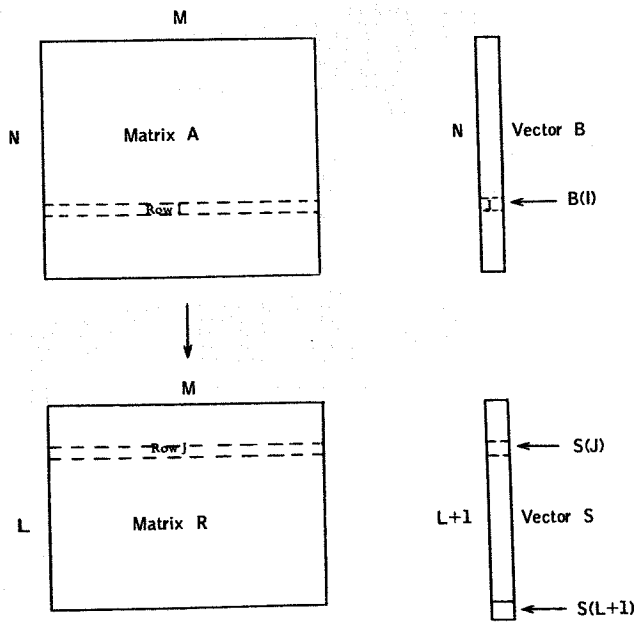


Figure 7. Row tabulation

```
C                                                                       RTAB  10
C     ...................................................................RTAB  20
C                                                                       RTAB  30
C        SUBROUTINE RTAB                                                RTAB  40
C                                                                       RTAB  50
C        PURPOSE                                                        RTAB  60
C           TABULATE ROWS OF A MATRIX TO FORM A SUMMARY MATRIX          RTAB  70
C                                                                       RTAB  80
C        USAGE                                                          RTAB  90
C           CALL RTAB(A,B,M,S,N,M,MS,L)                                 RTAB 100
C                                                                       RTAB 110
C        DESCRIPTION OF PARAMETERS                                      RTAB 120
C           A - NAME OF INPUT MATRIX                                    RTAB 130
C           B - NAME OF INPUT VECTOR OF LENGTH N CONTAINING KEY         RTAB 140
C           R - NAME OF OUTPUT MATRIX CONTAINING SUMMARY OF ROW DATA.   RTAB 150
C               IT IS INITIALLY SET TO ZERO BY THIS SUBROUTINE.         RTAB 160
C           S - NAME OF OUTPUT VECTOR OF LENGTH L+1 CONTAINING COUNTS   RTAB 170
C           N - NUMBER OF ROWS IN A                                     RTAB 180
C           M - NUMBER OF COLUMNS IN A AND R                            RTAB 190
C           L - NUMBER OF ROWS IN R                                     RTAB 200
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         RTAB 210
C                 0 - GENERAL                                           RTAB 220
C                 1 - SYMMETRIC                                         RTAB 230
C                 2 - DIAGONAL                                          RTAB 240
C                                                                       RTAB 250
C        REMARKS                                                        RTAB 260
C           MATRIX R IS ALWAYS A GENERAL MATRIX                         RTAB 270
C                                                                       RTAB 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RTAB 290
C           LOC                                                         RTAB 300
C           RADD                                                        RTAB 310
C                                                                       RTAB 320
C        METHOD                                                         RTAB 330
C           ROWS OF DATA IN MATRIX A ARE TABULATED BASED ON THE KEY     RTAB 340
C           CONTAINED IN VECTOR B. THE FLOATING POINT NUMBER IN B(I) IS RTAB 350
C           TRUNCATED TO FORM J. THE ITH ROW OF A IS ADDED TO THE JTH   RTAB 360
C           ROW OF R ELEMENT BY ELEMENT AND ONE IS ADDED TO S(J). IF J  RTAB 370
C           IS NOT BETWEEN ONE AND L, ONE IS ADDED TO S(L+1). THIS      RTAB 380
C           PROCEDURE IS REPEATED FOR EVERY ELEMENT IN VECTOR B.        RTAB 390
C           UPON COMPLETION, THE OUTPUT MATRIX R CONTAINS A SUMMARY OF  RTAB 400
C           ROW DATA AS SPECIFIED BY VECTOR B. EACH ELEMENT IN VECTOR S RTAB 410
C           CONTAINS A COUNT OF THE NUMBER OF ROWS OF A USED TO FORM THERTAB 420
C           CORRESPONDING ROW OF R. ELEMENT S(L+1) CONTAINS A COUNT OF  RTAB 430
C           THE NUMBER OF ROWS OF A NOT INCLUDED IN R AS A RESULT OF J  RTAB 440
C           BEING LESS THAN ONE OR GREATER THAN L.                      RTAB 450
C                                                                       RTAB 460
C     ...................................................................RTAB 470
C                                                                       RTAB 480
      SUBROUTINE RTAB(A,B,R,S,N,M,MS,L)                                 RTAB 490
      DIMENSION A(1),B(1),R(1),S(1)                                     RTAB 500
C                                                                       RTAB 510
C        CLEAR OUTPUT AREAS                                             RTAB 520
C                                                                       RTAB 530
      CALL LOC(M,L,IT,M,L,C)                                            RTAB 540
      DO 1C IR=1,IT                                                     RTAB 550
   1C R(IR)=C.C                                                         RTAB 560
      DO 2C IS=1,L                                                      RTAB 570
   2C S(IS)=C.C                                                         RTAB 580
      S(L+1)=C.C                                                        RTAB 590
C                                                                       RTAB 600
      DO 6C I=1,N                                                       RTAB 610
C                                                                       RTAB 620
C        TEST FOR THE KEY OUTSIDE THE RANGE                             RTAB 630
C                                                                       RTAB 640
      JR=B(I)                                                           RTAB 650
      IF (JR-1) 5C,4C,3C                                                RTAB 660
   3C IF (JR-L) 4C,4C,5C                                                RTAB 670
C                                                                       RTAB 680
C                                                                       RTAB 690
C        ADD ROW OF A TO ROW OF R AND 1 TO COUNT                        RTAB 700
C                                                                       RTAB 710
   4C CALL RADD (A,I,R,JR,N,M,MS,L)                                     RTAB 720
      S(JR)=S(JR)+1.C                                                   RTAB 730
      GO TO 6C                                                          RTAB 740
C                                                                       RTAB 750
   5C S(L+1)=S(L+1)+1.C                                                 RTAB 760
   6C CONTINUE                                                          RTAB 770
      RETURN                                                            RTAB 780
      END                                                               RTAB 790
```

Subroutine CTAB

The function of this subroutine is graphically displayed by Figure 8 (see description under "Method" in the program listing).

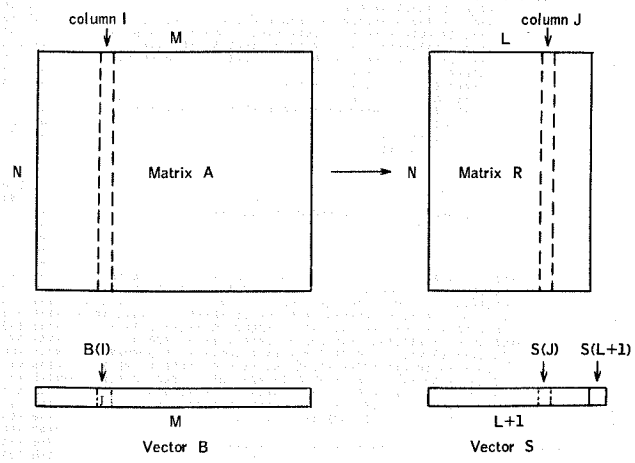


Figure 8. Column tabulation

```
C                                                                       CTAB  10
C     ..................................................................CTAB  20
C                                                                       CTAB  30
C        SUBROUTINE CTAB                                                CTAB  40
C                                                                       CTAB  50
C        PURPOSE                                                        CTAB  60
C           TABULATE COLUMNS OF A MATRIX TO FORM A SUMMARY MATRIX       CTAB  70
C                                                                       CTAB  80
C        USAGE                                                          CTAB  90
C           CALL CTAB(A,B,R,S,N,M,MS,L)                                 CTAB 100
C                                                                       CTAB 110
C        DESCRIPTION OF PARAMETERS                                      CTAB 120
C           A - NAME OF INPUT MATRIX                                    CTAB 130
C           B - NAME OF INPUT VECTOR OF LENGTH M CONTAINING KEY         CTAB 140
C           R - NAME OF OUTPUT MATRIX CONTAINING SUMMARY OF COLUMN DATA.CTAB 150
C               IT IS INITIALLY SET TO ZERO BY THIS SUBROUTINE.         CTAB 160
C           S - NAME OF OUTPUT VECTOR OF LENGTH L+1 CONTAINING COUNTS   CTAB 170
C           N - NUMBER OF ROWS IN A AND R                               CTAB 180
C           M - NUMBER OF COLUMNS IN A                                  CTAB 190
C           L - NUMBER OF COLUMNS IN R                                  CTAB 200
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         CTAB 210
C                 0 - GENERAL                                           CTAB 220
C                 1 - SYMMETRIC                                         CTAB 230
C                 2 - DIAGONAL                                          CTAB 240
C                                                                       CTAB 250
C        REMARKS                                                        CTAB 260
C           MATRIX R IS ALWAYS A GENERAL MATRIX                         CTAB 270
C                                                                       CTAB 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CTAB 290
C           LOC                                                         CTAB 300
C           CADD                                                        CTAB 310
C                                                                       CTAB 320
C        METHOD                                                         CTAB 330
C           COLUMNS OF DATA IN MATRIX A ARE TABULATED BASED ON THE KEY  CTAB 340
C           CONTAINED IN VECTOR B. THE FLOATING POINT NUMBER IN B(I) IS CTAB 350
C           TRUNCATED TO FORM J. THE ITH COLUMN OF A IS ADDED TO THE JTHCTAB 360
C           COLUMN OF MATRIX R AND ONE IS ADDED TO S(J). IF THE VALUE OFCTAB 370
C           J IS NOT BETWEEN 1 AND L, ONE IS ADDED TO S(L+1)            CTAB 380
C           UPON COMPLETION, THE OUTPUT MATRIX R CONTAINS A SUMMARY OF  CTAB 390
C           COLUMN DATA AS SPECIFIED BY VECTOR B. EACH ELEMENT IN VECTORCTAB 400
C           S CONTAINS A COUNT OF THE NUMBER OF COLUMNS OF A USED TO    CTAB 410
C           FORM R. ELEMENT S(L+1) CONTAINS THE NUMBER OF COLUMNS OF A  CTAB 420
C           NOT INCLUDED IN R AS A RESULT OF J BEING LESS THAN ONE OR   CTAB 430
C           GREATER THAN L.                                             CTAB 440
C                                                                       CTAB 450
C     ..................................................................CTAB 460
C                                                                       CTAB 470
      SUBROUTINE CTAB(A,B,R,S,N,M,MS,L)                                 CTAB 480
      DIMENSION A(1),B(1),R(1),S(1)                                     CTAB 490
C                                                                       CTAB 500
C        CLEAR OUTPUT AREAS                                             CTAB 510
C                                                                       CTAB 520
      CALL LOC(N,L,IT,N,L,C)                                            CTAB 530
      DO 1C IR=1,IT                                                     CTAB 540
   1C R(IR)=C.C                                                         CTAB 550
      DO 2C IS=1,L                                                      CTAB 560
   2C S(IS)=C.C                                                         CTAB 570
      S(L+1)=C.C                                                        CTAB 580
C                                                                       CTAB 590
      DO 6C I=1,M                                                       CTAB 600
C                                                                       CTAB 610
C        TEST FOR THE KEY OUTSIDE THE RANGE                             CTAB 620
C                                                                       CTAB 630
      JR=B(I)                                                           CTAB 640
      IF (JR-1) 5C,4C,3C                                                CTAB 650
   3C IF (JR-L) 4C,4C,5C                                                CTAB 660
C                                                                       CTAB 670
C                                                                       CTAB 680
C        ADD COLUMN OF A TO COLUMN OF R AND 1 TO COUNT                  CTAB 690
C                                                                       CTAB 700
   4C CALL CADD (A,I,R,JR,N,M,MS,L)                                     CTAB 710
      S(JR)=S(JR)+1.C                                                   CTAB 720
      GO TO 6C                                                          CTAB 730
C                                                                       CTAB 740
   5C S(L+1)=S(L+1)+1.C                                                 CTAB 750
   6C CONTINUE                                                          CTAB 760
      RETURN                                                            CTAB 770
      END                                                               CTAB 780
```

Subroutine RSRT

```
C                                                                       RSRT  10
C     ..................................................................RSRT  20
C                                                                       RSRT  30
C        SUBROUTINE RSRT                                                RSRT  40
C                                                                       RSRT  50
C        PURPOSE                                                        RSRT  60
C           SORT ROWS OF A MATRIX                                       RSRT  70
C                                                                       RSRT  80
C        USAGE                                                          RSRT  90
C           CALL RSRT(A,B,R,N,M,MS)                                     RSRT 100
C                                                                       RSRT 110
C        DESCRIPTION OF PARAMETERS                                      RSRT 120
C           A - NAME OF INPUT MATRIX TO BE SORTED                       RSRT 130
C           B - NAME OF INPUT VECTOR WHICH CONTAINS SORTING KEY         RSRT 140
C           R - NAME OF SORTED OUTPUT MATRIX                            RSRT 150
C           N - NUMBER OF ROWS IN A AND R AND LENGTH OF B               RSRT 160
C           M - NUMBER OF COLUMNS IN A AND R                            RSRT 170
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         RSRT 180
C                 0 - GENERAL                                           RSRT 190
C                 1 - SYMMETRIC                                         RSRT 200
C                 2 - DIAGONAL                                          RSRT 210
C                                                                       RSRT 220
C        REMARKS                                                        RSRT 230
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         RSRT 240
C           MATRIX R IS ALWAYS A GENERAL MATRIX                         RSRT 250
C           N MUST BE GREATER THAN ONE.                                 RSRT 260
C                                                                       RSRT 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RSRT 280
C           LOC                                                         RSRT 290
C                                                                       RSRT 300
C        METHOD                                                         RSRT 310
C           ROWS OF INPUT MATRIX A ARE SORTED TO FORM OUTPUT MATRIX R.  RSRT 320
C           THE SORTED ROW SEQUENCE IS DETERMINED BY THE VALUES OF      RSRT 330
C           ELEMENTS IN COLUMN VECTOR B. THE LOWEST VALUED ELEMENT IN   RSRT 340
C           B WILL CAUSE THE CORRESPONDING ROW OF A TO BE PLACED IN THE RSRT 350
C           FIRST ROW OF R. THE HIGHEST VALUED ELEMENT OF B WILL CAUSE  RSRT 360
C           THE CORRESPONDING ROW OF A TO BE PLACED IN THE LAST ROW OF  RSRT 370
C           R. IF DUPLICATE VALUES EXIST IN B, THE CORRESPONDING ROWS   RSRT 380
C           OF A ARE MOVED TO R IN THE SAME ORDER AS IN A.              RSRT 390
C                                                                       RSRT 400
C     ..................................................................RSRT 410
C                                                                       RSRT 420
      SUBROUTINE RSRT(A,B,R,N,M,MS)                                     RSRT 430
      DIMENSION A(1),B(1),R(1)                                          RSRT 440
C                                                                       RSRT 450
C        MOVE SORTING KEY VECTOR TO FIRST COLUMN OF OUTPUT MATRIX       RSRT 460
C        AND BUILD ORIGINAL SEQUENCE LIST IN SECOND COLUMN              RSRT 470
C                                                                       RSRT 480
      DO 1C I=1,N                                                       RSRT 490
      R(I)=B(I)                                                         RSRT 500
      I2=I+N                                                            RSRT 510
   1C R(I2)=I                                                           RSRT 520
C                                                                       RSRT 530
C        SORT ELEMENTS IN SORTING KEY VECTOR (ORIGINAL SEQUENCE LIST    RSRT 540
C        IS RESEQUENCED ACCORDINGLY)                                    RSRT 550
C                                                                       RSRT 560
      L=N+1                                                             RSRT 570
   2C ISORT=0                                                           RSRT 580
      L=L-1                                                             RSRT 590
      DO 4C I=2,L                                                       RSRT 600
      IF(R(I)-R(I-1)) 3C,4C,4C                                          RSRT 610
   3C ISORT=1                                                           RSRT 620
      RSAVE=R(I)                                                        RSRT 630
      R(I)=R(I-1)                                                       RSRT 640
      R(I-1)=RSAVE                                                      RSRT 650
      I2=I+N                                                            RSRT 660
      SAVER=R(I2)                                                       RSRT 670
      R(I2)=R(I2-1)                                                     RSRT 680
      R(I2-1)=SAVER                                                     RSRT 690
   4C CONTINUE                                                          RSRT 700
      IF(ISORT) 2C,5C,2C                                                RSRT 710
C                                                                       RSRT 720
C        MOVE ROWS FROM MATRIX A TO MATRIX R (NUMBER IN SECOND COLUMN   RSRT 730
C        OF R REPRESENTS ROW NUMBER OF MATRIX A TO BE MOVED)            RSRT 740
C                                                                       RSRT 750
   5C DO 8C I=1,N                                                       RSRT 760
C                                                                       RSRT 770
C        GET ROW NUMBER IN MATRIX A                                     RSRT 780
C                                                                       RSRT 790
      I2=I+N                                                            RSRT 800
      IN=R(I2)                                                          RSRT 810
C                                                                       RSRT 820
      IR=I-N                                                            RSRT 830
      DO 8C J=1,M                                                       RSRT 840
C                                                                       RSRT 850
C        LOCATE ELEMENT IN OUTPUT MATRIX                                RSRT 860
C                                                                       RSRT 870
      IR=IR+N                                                           RSRT 880
C                                                                       RSRT 890
C        LOCATE ELEMENT IN INPUT MATRIX                                 RSRT 900
C                                                                       RSRT 910
      CALL LOC(IN,J,IA,N,M,MS)                                          RSRT 920
C                                                                       RSRT 930
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       RSRT 940
C                                                                       RSRT 950
      IF(IA) 6C,7C,6C                                                   RSRT 960
C                                                                       RSRT 970
C        MOVE ELEMENT TO OUTPUT MATRIX                                  RSRT 980
C                                                                       RSRT 990
   6C R(IR)=A(IA)                                                       RSRT1000
      GO TO 8C                                                          RSRT1010
   7C R(IR)=0                                                           RSRT1020
   8C CONTINUE                                                          RSRT1030
      RETURN                                                            RSRT1040
      END                                                               RSRT1050
```

Subroutine CSRT

```
C                                                                       CSRT  10
C     ..................................................................CSRT  20
C                                                                       CSRT  30
C        SUBROUTINE CSRT                                                CSRT  40
C                                                                       CSRT  50
C        PURPOSE                                                        CSRT  60
C           SORT COLUMNS OF A MATRIX                                    CSRT  70
C                                                                       CSRT  80
C        USAGE                                                          CSRT  90
C           CALL CSRT(A,B,R,N,M,MS)                                     CSRT 100
C                                                                       CSRT 110
C        DESCRIPTION OF PARAMETERS                                      CSRT 120
C           A - NAME OF INPUT MATRIX TO BE SORTED                       CSRT 130
C           B - NAME OF INPUT VECTOR WHICH CONTAINS SORTING KEY         CSRT 140
C           R - NAME OF SORTED OUTPUT MATRIX                            CSRT 150
C           N - NUMBER OF ROWS IN A AND R                               CSRT 160
C           M - NUMBER OF COLUMNS IN A AND R AND LENGTH OF B            CSRT 170
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         CSRT 180
C                 0 - GENERAL                                           CSRT 190
C                 1 - SYMMETRIC                                         CSRT 200
C                 2 - DIAGONAL                                          CSRT 210
C                                                                       CSRT 220
C        REMARKS                                                        CSRT 230
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRIX A         CSRT 240
C           MATRIX R IS ALWAYS A GENERAL MATRIX                         CSRT 250
C           M MUST BE GREATER THAN ONE.                                 CSRT 260
C                                                                       CSRT 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CSRT 280
C           LOC                                                         CSRT 290
C           CCPY                                                        CSRT 300
C                                                                       CSRT 310
C        METHOD                                                         CSRT 320
C           COLUMNS OF INPUT MATRIX A ARE SORTED TO FORM OUTPUT MATRIX  CSRT 330
C           R. THE SORTED COLUMN SEQUENCE IS DETERMINED BY THE VALUES OFCSRT 340
C           ELEMENTS IN ROW VECTOR B. THE LOWEST VALUED ELEMENT IN      CSRT 350
C           B WILL CAUSE THE CORRESPONDING COLUMN OF A TO BE PLACED IN  CSRT 360
C           THE FIRST COLUMN OF R. THE HIGHEST VALUED ELEMENT OF B WILL CSRT 370
C           CAUSE THE CORRESPONDING ROW OF A TO BE PLACED IN THE LAST   CSRT 380
C           COLUMN OF R. IF DUPLICATE VALUES EXIST IN B, THE            CSRT 390
C           CORRESPONDING COLUMNS OF A ARE MOVED TO R IN THE SAME ORDER CSRT 400
C           AS IN A.                                                    CSRT 410
C                                                                       CSRT 420
C     ..................................................................CSRT 430
C                                                                       CSRT 440
      SUBROUTINE CSRT(A,B,R,N,M,MS)                                     CSRT 450
      DIMENSION A(1),B(1),R(1)                                          CSRT 460
C                                                                       CSRT 470
C        MOVE SORTING KEY VECTOR TO FIRST ROW OF OUTPUT MATRIX          CSRT 480
C        AND BUILD ORIGINAL SEQUENCE LIST IN SECOND ROW                 CSRT 490
C                                                                       CSRT 500
      IK=1                                                              CSRT 510
      DO 10 J=1,M                                                       CSRT 520
      R(IK)=B(J)                                                        CSRT 530
      R(IK+1)=J                                                         CSRT 540
   10 IK=IK+N                                                           CSRT 550
C                                                                       CSRT 560
C        SORT ELEMENTS IN SORTING KEY VECTOR (ORIGINAL SEQUENCE LIST    CSRT 570
C        IS RESEQUENCED ACCORDINGLY)                                    CSRT 580
C                                                                       CSRT 590
      L=M+1                                                             CSRT 600
   20 ISORT=0                                                           CSRT 610
      L=L-1                                                             CSRT 620
      IP=1                                                              CSRT 630
      IQ=N+1                                                            CSRT 640
      DO 50 J=2,L                                                       CSRT 650
      IF(R(IQ)-R(IP)) 30,40,40                                          CSRT 660
   30 ISORT=1                                                           CSRT 670
      RSAVE=R(IQ)                                                       CSRT 680
      R(IQ)=R(IP)                                                       CSRT 690
      R(IP)=RSAVE                                                       CSRT 700
      SAVER=R(IQ+1)                                                     CSRT 710
      R(IQ+1)=R(IP+1)                                                   CSRT 720
      R(IP+1)=SAVER                                                     CSRT 730
   40 IP=IP+N                                                           CSRT 740
      IQ=IQ+N                                                           CSRT 750
   50 CONTINUE                                                          CSRT 760
      IF(ISORT) 20,60,20                                                CSRT 770
C                                                                       CSRT 780
C        MOVE COLUMNS FROM MATRIX A TO MATRIX R (NUMBER IN SECOND ROW   CSRT 790
C        OF R REPRESENTS COLUMN NUMBER OF MATRIX A TO BE MOVED)         CSRT 800
C                                                                       CSRT 810
   60 IQ=-N                                                             CSRT 820
      DO 70 J=1,M                                                       CSRT 830
      IQ=IQ+N                                                           CSRT 840
C                                                                       CSRT 850
C        GET COLUMN NUMBER IN MATRIX A                                  CSRT 860
C                                                                       CSRT 870
      I2=IQ+2                                                           CSRT 880
      IN=R(I2)                                                          CSRT 890
C                                                                       CSRT 900
C        MOVE COLUMN                                                    CSRT 910
C                                                                       CSRT 920
      IR=IQ+1                                                           CSRT 930
      CALL CCPY(A,IN,R(IR),N,M,MS)                                      CSRT 940
   70 CONTINUE                                                          CSRT 950
      RETURN                                                            CSRT 960
      END                                                               CSRT 970
```

Subroutine RCUT

```
C                                                                       RCUT  10
C     ..................................................................RCUT  20
C                                                                       RCUT  30
C        SUBROUTINE RCUT                                                RCUT  40
C                                                                       RCUT  50
C        PURPOSE                                                        RCUT  60
C           PARTITION A MATRIX BETWEEN SPECIFIED ROWS TO FORM TWO       RCUT  70
C           RESULTANT MATRICES                                          RCUT  80
C                                                                       RCUT  90
C        USAGE                                                          RCUT 100
C           CALL RCUT (A,L,R,S,N,M,MS)                                  RCUT 110
C                                                                       RCUT 120
C        DESCRIPTION OF PARAMETERS                                      RCUT 130
C           A - NAME OF INPUT MATRIX                                    RCUT 140
C           L - ROW OF A ABOVE WHICH PARTITIONING TAKES PLACE           RCUT 150
C           R - NAME OF MATRIX TO BE FORMED FROM UPPER PORTION OF A     RCUT 160
C           S - NAME OF MATRIX TO BE FORMED FROM LOWER PORTION OF A     RCUT 170
C           N - NUMBER OF ROWS IN A                                     RCUT 180
C           M - NUMBER OF COLUMNS IN A                                  RCUT 190
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         RCUT 200
C                 0 - GENERAL                                           RCUT 210
C                 1 - SYMMETRIC                                         RCUT 220
C                 2 - DIAGONAL                                          RCUT 230
C                                                                       RCUT 240
C        REMARKS                                                        RCUT 250
C           MATRIX R CANNOT BE IN SAME LOCATION AS MATRIX A             RCUT 260
C           MATRIX S CANNOT BE IN SAME LOCATION AS MATRIX A             RCUT 270
C           MATRIX R CANNOT BE IN SAME LOCATION AS MATRIX S             RCUT 280
C           MATRIX R AND MATRIX S ARE ALWAYS GENERAL MATRICES           RCUT 290
C                                                                       RCUT 300
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RCUT 310
C           LOC                                                         RCUT 320
C                                                                       RCUT 330
C        METHOD                                                         RCUT 340
C           ELEMENTS OF MATRIX A ABOVE ROW L ARE MOVED TO FORM MATRIX R RCUT 350
C           OF L-1 ROWS AND M COLUMNS. ELEMENTS OF MATRIX A IN ROW L    RCUT 360
C           AND BELOW ARE MOVED TO FORM MATRIX S OF N-L+1 ROWS AND M    RCUT 370
C           COLUMNS                                                     RCUT 380
C                                                                       RCUT 390
C     ..................................................................RCUT 400
C                                                                       RCUT 410
      SUBROUTINE RCUT(A,L,R,S,N,M,MS)                                   RCUT 420
      DIMENSION A(1),R(1),S(1)                                          RCUT 430
C                                                                       RCUT 440
      IR=0                                                              RCUT 450
      IS=0                                                              RCUT 460
      DO 70 J=1,M                                                       RCUT 470
      DO 70 I=1,N                                                       RCUT 480
C                                                                       RCUT 490
C        FIND LOCATION IN OUTPUT MATRIX AND SET TO ZERO                 RCUT 500
C                                                                       RCUT 510
      IF(I-L) 20,10,10                                                  RCUT 520
   10 IS=IS+1                                                           RCUT 530
      S(IS)=0.0                                                         RCUT 540
      GO TO 30                                                          RCUT 550
   20 IR=IR+1                                                           RCUT 560
      R(IR)=0.0                                                         RCUT 570
C                                                                       RCUT 580
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE                     RCUT 590
C                                                                       RCUT 600
   30 CALL LOC(I,J,IJ,N,M,MS)                                           RCUT 610
C                                                                       RCUT 620
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       RCUT 630
C                                                                       RCUT 640
      IF(IJ) 40,70,40                                                   RCUT 650
C                                                                       RCUT 660
C        DETERMINE WHETHER ABOVE OR BELOW L                             RCUT 670
C                                                                       RCUT 680
   40 IF(I-L) 50,50,50                                                  RCUT 690
   50 S(IS)=A(IJ)                                                       RCUT 700
      GO TO 70                                                          RCUT 710
   60 R(IR)=A(IJ)                                                       RCUT 720
   70 CONTINUE                                                          RCUT 730
      RETURN                                                            RCUT 740
      END                                                               RCUT 750
```

Subroutine CCUT

```
C                                                                       CCUT  10
C     ..................................................................CCUT  20
C                                                                       CCUT  30
C        SUBROUTINE CCUT                                                CCUT  40
C                                                                       CCUT  50
C        PURPOSE                                                        CCUT  60
C           PARTITION A MATRIX BETWEEN SPECIFIED COLUMNS TO FORM TWO    CCUT  70
C           RESULTANT MATRICES                                          CCUT  80
C                                                                       CCUT  90
C        USAGE                                                          CCUT 100
C           CALL CCUT (A,L,R,S,N,M,MS)                                  CCUT 110
C                                                                       CCUT 120
C        DESCRIPTION OF PARAMETERS                                      CCUT 130
C           A - NAME OF INPUT MATRIX                                    CCUT 140
C           L - COLUMN OF A TO THE LEFT OF WHICH PARTITIONING TAKES     CCUT 150
C               PLACE                                                   CCUT 160
C           R - NAME OF MATRIX TO BE FORMED FROM LEFT PORTION OF A      CCUT 170
C           S - NAME OF MATRIX TO BE FORMED FROM RIGHT PORTION OF A     CCUT 180
C           N - NUMBER OF ROWS IN A                                     CCUT 190
C           M - NUMBER OF COLUMNS IN A                                  CCUT 200
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         CCUT 210
C                 0 - GENERAL                                           CCUT 220
C                 1 - SYMMETRIC                                         CCUT 230
C                 2 - DIAGONAL                                          CCUT 240
C                                                                       CCUT 250
C        REMARKS                                                        CCUT 260
C           MATRIX R CANNOT BE IN SAME LOCATION AS MATRIX A             CCUT 270
C           MATRIX S CANNOT BE IN SAME LOCATION AS MATRIX A             CCUT 280
C           MATRIX R CANNOT BE IN SAME LOCATION AS MATRIX S             CCUT 290
C           MATRIX R AND MATRIX S ARE ALWAYS GENERAL MATRICES           CCUT 300
C                                                                       CCUT 310
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CCUT 320
C           LOC                                                         CCUT 330
C                                                                       CCUT 340
C        METHOD                                                         CCUT 350
C           ELEMENTS OF MATRIX A TO THE LEFT OF COLUMN L ARE MOVED TO   CCUT 360
C           FORM MATRIX R OF N ROWS AND L-1 COLUMNS. ELEMENTS OF        CCUT 370
C           MATRIX A IN COLUMN L AND TO THE RIGHT OF L ARE MOVED TO FORMCCUT 380
C           MATRIX S OF N ROWS AND M-L+1 COLUMNS.                       CCUT 390
C                                                                       CCUT 400
C     ..................................................................CCUT 410
C                                                                       CCUT 420
      SUBROUTINE CCUT(A,L,R,S,N,M,MS)                                   CCUT 430
      DIMENSION A(1),R(1),S(1)                                          CCUT 440
C                                                                       CCUT 450
      IR=0                                                              CCUT 460
      IS=0                                                              CCUT 470
      DO 70 J=1,M                                                       CCUT 480
      DO 70 I=1,N                                                       CCUT 490
C                                                                       CCUT 500
C        FIND LOCATION IN OUTPUT MATRIX AND SET TO ZERO                 CCUT 510
C                                                                       CCUT 520
      IF(J-L) 20,10,10                                                  CCUT 530
   10 IS=IS+1                                                           CCUT 540
      S(IS)=0.0                                                         CCUT 550
      GO TO 30                                                          CCUT 560
   20 IR=IR+1                                                           CCUT 570
      R(IR)=0.0                                                         CCUT 580
C                                                                       CCUT 590
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE                     CCUT 600
C                                                                       CCUT 610
   30 CALL LOC(I,J,IJ,N,M,MS)                                           CCUT 620
C                                                                       CCUT 630
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       CCUT 640
C                                                                       CCUT 650
      IF(IJ) 40,70,40                                                   CCUT 660
C                                                                       CCUT 670
C        DETERMINE WHETHER RIGHT OR LEFT OF L                           CCUT 680
C                                                                       CCUT 690
   40 IF(J-L) 60,50,50                                                  CCUT 700
   50 S(IS)=A(IJ)                                                       CCUT 710
      GO TO 70                                                          CCUT 720
   60 R(IR)=A(IJ)                                                       CCUT 730
   70 CONTINUE                                                          CCUT 740
      RETURN                                                            CCUT 750
      END                                                               CCUT 760
```

Subroutine RTIE

```
C                                                                       RTIE  10
C     ..................................................................RTIE  20
C                                                                       RTIE  30
C        SUBROUTINE RTIE                                                RTIE  40
C                                                                       RTIE  50
C        PURPOSE                                                        RTIE  60
C           ADJOIN TWO MATRICES WITH SAME COLUMN DIMENSION TO FORM ONE  RTIE  70
C           RESULTANT MATRIX (SEE METHOD)                               RTIE  80
C                                                                       RTIE  90
C        USAGE                                                          RTIE 100
C           CALL RTIE(A,B,R,N,M,MSA,MSB,L)                              RTIE 110
C                                                                       RTIE 120
C        DESCRIPTION OF PARAMETERS                                      RTIE 130
C           A - NAME OF FIRST INPUT MATRIX                              RTIE 140
C           B - NAME OF SECOND INPUT MATRIX                             RTIE 150
C           R - NAME OF OUTPUT MATRIX                                   RTIE 160
C           N - NUMBER OF ROWS IN A                                     RTIE 170
C           M - NUMBER OF COLUMNS IN A,B,R                              RTIE 180
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A         RTIE 190
C                 0 - GENERAL                                           RTIE 200
C                 1 - SYMMETRIC                                         RTIE 210
C                 2 - DIAGONAL                                          RTIE 220
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B                       RTIE 230
C           L - NUMBER OF ROWS IN B                                     RTIE 240
C                                                                       RTIE 250
C        REMARKS                                                        RTIE 260
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRICES A OR B  RTIE 270
C           MATRIX R IS ALWAYS A GENERAL MATRIX                         RTIE 280
C           MATRIX A MUST HAVE THE SAME NUMBER OF COLUMNS AS MATRIX B   RTIE 290
C                                                                       RTIE 300
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RTIE 310
C           LOC                                                         RTIE 320
C                                                                       RTIE 330
C        METHOD                                                         RTIE 340
C           MATRIX B IS ATTACHED TO THE BOTTOM OF MATRIX A .            RTIE 350
C           THE RESULTANT MATRIX R CONTAINS N+L ROWS AND M COLUMNS.     RTIE 360
C                                                                       RTIE 370
C     ..................................................................RTIE 380
C                                                                       RTIE 390
      SUBROUTINE RTIE(A,B,R,N,M,MSA,MSB,L)                              RTIE 400
      DIMENSION A(1),B(1),R(1)                                          RTIE 410
C                                                                       RTIE 420
      NN=N                                                              RTIE 430
      IR=0                                                              RTIE 440
      NX=NN                                                             RTIE 450
      MSX=MSA                                                           RTIE 460
      DO 9 J=1,M                                                        RTIE 470
      DO 8 II=1,2                                                       RTIE 480
      DO 7 I=1,NN                                                       RTIE 490
      IR=IR+1                                                           RTIE 500
      R(IR)=0.0                                                         RTIE 510
C                                                                       RTIE 520
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE                     RTIE 530
C                                                                       RTIE 540
      CALL LOC(I,J,IJ,NN,M,MSX)                                         RTIE 550
C                                                                       RTIE 560
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX                       RTIE 570
C                                                                       RTIE 580
      IF(IJ) 2,7,2                                                      RTIE 590
C                                                                       RTIE 600
C        MOVE ELEMENT TO MATRIX R                                       RTIE 610
C                                                                       RTIE 620
    2 GO TO(3,4),II                                                     RTIE 630
    3 R(IR)=A(IJ)                                                       RTIE 640
      GO TO 7                                                           RTIE 650
    4 R(IR)=B(IJ)                                                       RTIE 660
    7 CONTINUE                                                          RTIE 670
C                                                                       RTIE 680
C        REPEAT ABOVE FOR MATRIX B                                      RTIE 690
C                                                                       RTIE 700
      MSX=MSB                                                           RTIE 710
    8 NN=L                                                              RTIE 720
C                                                                       RTIE 730
C        RESET FOR NEXT COLUMN                                          RTIE 740
C                                                                       RTIE 750
      MSX=MSA                                                           RTIE 760
    9 NN=NX                                                             RTIE 770
      RETURN                                                            RTIE 780
      END                                                               RTIE 790
```

Subroutine CTIE

```
C
C     ..................................................................
C
C        SUBROUTINE CTIE
C
C        PURPOSE
C           ADJOIN TWO MATRICES WITH SAME ROW DIMENSION TO FORM ONE
C           RESULTANT MATRIX (SEE METHOD)
C
C        USAGE
C           CALL CTIE(A,B,R,N,M,MSA,MSB,L)
C
C        DESCRIPTION OF PARAMETERS
C           A - NAME OF FIRST INPUT MATRIX
C           B - NAME OF SECOND INPUT MATRIX
C           R - NAME OF OUTPUT MATRIX
C           N - NUMBER OF ROWS IN A,B,R
C           M - NUMBER OF COLUMNS IN A
C           MSA - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A
C                 0 - GENERAL
C                 1 - SYMMETRIC
C                 2 - DIAGONAL
C           MSB - SAME AS MSA EXCEPT FOR MATRIX B
C           L - NUMBER OF COLUMNS IN B
C
C        REMARKS
C           MATRIX R CANNOT BE IN THE SAME LOCATION AS MATRICES A OR B
C           MATRIX R IS ALWAYS A GENERAL MATRIX
C           MATRIX A MUST HAVE THE SAME NUMBER OF ROWS AS MATRIX B
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           LOC
C
C        METHOD
C           MATRIX B IS ATTACHED TO THE RIGHT OF MATRIX A .
C           THE RESULTANT MATRIX R CONTAINS N ROWS AND M+L COLUMNS
C
C     ..................................................................
C
      SUBROUTINE CTIE(A,B,R,N,M,MSA,MSB,L)
      DIMENSION A(1),B(1),R(1)
C
      MM=M
      IR=0
      MSX=MSA
      DO 6 JJ=1,2
      DO 5 J=1,MM
      DO 5 I=1,N
      IR=IR+1
      R(IR)=0.0
C
C        LOCATE ELEMENT FOR ANY MATRIX STORAGE MODE
C
      CALL LOC(I,J,IJ,N,MM,MSX)
C
C        TEST FOR ZERO ELEMENT IN DIAGONAL MATRIX
C
      IF(IJ) 2,5,2
C
C        MOVE ELEMENT TO MATRIX R
C
    2 GO TO(3,4),JJ
    3 R(IR)=A(IJ)
      GO TO 5
    4 R(IR)=B(IJ)
    5 CONTINUE
C
C        REPEAT ABOVE FOR MATRIX B
C
      MSX=MSB
      MM=L
    6 CONTINUE
      RETURN
      END
```

Subroutines MPRC and DMPRC

These subroutines permute the rows or columns of a given m by n matrix A according to a transposition vector $ITRA = (t_1, \ldots, t_s)$ or its inverse. (See the general discussion under "Permutations" for definitions and notation.)

1. Mathematical background

Permuting the rows or columns of A according to the transposition vector ITRA means applying the permutation $(1, t_1), \ldots, (s, t_s)$ to the rows or columns of A, while permuting according to $ITRA^{-1}$ means applying the permutation $(s, t_s), \ldots, (1, t_1)$.

2. Programming considerations

If the columns of A are to be permuted, the input parameter IROCO must be nonzero, and $s = n$; if the rows are to be permuted, IROCO must be zero, and $s = m$. Similarly, if the input parameter INV is nonzero, the rows or columns of A are permuted according to ITRA; while if INV is zero, the rows or columns of A are permuted according to $ITRA^{-1}$.

If m and n are not both positive, or if ITRA is not a transposition vector on $S = \{1, \ldots, s\}$, the subroutines are prematurely terminated and the error parameter IER is set to -1 or 1, respectively. If there is no error, IER is set to zero.

The matrix A is assumed to be stored columnwise, and the permutation of the rows or columns of A is performed in s steps. At the ith step, if the permutation is according to $ITRA^{-1}$, rows or columns i and s_i are interchanged; if the permutation is according to ITRA, rows or columns $s - i + 1$ and t_{s-i+1} are interchanged.

```
C
C     ..................................................................
C
C        SUBROUTINE MPRC
C
C        PURPOSE
C           TO PERMUTE THE ROWS OR COLUMNS OF A GIVEN MATRIX ACCORDING
C           TO A GIVEN TRANSPOSITION VECTOR OR ITS INVERSE. (SEE THE
C           DISCUSSION ON PERMUTATIONS FOR DEFINITIONS AND NOTATION.)
C
C        USAGE
C           CALL MPRC(A,M,N,ITRA,INV,IROCO,IER)
C
C        DESCRIPTION OF PARAMETERS
C           A     - GIVEN M BY N MATRIX AND RESULTING PERMUTED MATRIX
C           M     - NUMBER OF ROWS OF A
C           N     - NUMBER OF COLUMNS OF A
C           ITRA  - GIVEN TRANSPOSITION VECTOR (DIMENSION M IF ROWS ARE
C                   PERMUTED, N IF COLUMNS ARE PERMUTED)
C           INV   - INPUT PARAMETER
C                   INV NON-ZERO  -  PERMUTE ACCORDING TO ITRA
C                   INV     =   0  -  PERMUTE ACCORDING TO ITRA INVERSE
C           IROCO - INPUT PARAMETER
C                   IROCO NON-ZERO  -  PERMUTE THE COLUMNS OF A
C                   IROCO     =   0  -  PERMUTE THE ROWS OF A
C           IER   - RESULTING ERROR PARAMETER
C                   IER = -1  -  M AND N ARE NOT BOTH POSITIVE
C                   IER =  0  -  NO ERROR
C                   IER =  1  -  ITRA IS NOT A TRANSPOSITION VECTOR ON
C                                1,...,M IF ROWS ARE PERMUTED, 1,...,N
C                                IF COLUMNS ARE PERMUTED
C
C        REMARKS
C           (1)  IF IER=-1 THERE IS NO COMPUTATION.
C           (2)  IF IER= 1, THEN COMPUTATION HAS BEEN UNSUCCESSFUL DUE
C                TO ERROR, BUT THE MATRIX A WILL REFLECT THE ROW OR
C                COLUMN INTERCHANGES PERFORMED BEFORE THE ERROR WAS
C                DETECTED.
C           (3)  THE MATRIX A IS ASSUMED TO BE STORED COLUMNWISE.
```

```
C                                                                       MPRC 400
C       SUBROUTINES AND SUBPROGRAMS REQUIRED                            MPRC 410
C          NONE                                                         MPRC 420
C                                                                       MPRC 430
C       METHOD                                                          MPRC 440
C          THE ROWS OR COLUMNS ARE PERMUTED ELEMENTWISE, INTERCHANGING  MPRC 450
C          ROW OR COLUMN 1 AND ITRA(1),....,ROW OR COLUMN K AND ITRA(K) MPRC 460
C          IN THAT ORDER IF INV=0, AND OTHERWISE INTERCHANGING ROW OR   MPRC 470
C          COLUMN K AND ITRA(K),....,ROW OR COLUMN 1 AND ITRA(1), WHERE MPRC 480
C          K IS M OR N DEPENDING ON WHETHER WE PERMUTE ROWS OR COLUMNS.MPRC 490
C                                                                       MPRC 500
C     ..................................................................MPRC 510
C                                                                       MPRC 520
      SUBROUTINE MPRC(A,M,N,ITRA,INV,IROCO,IER)                         MPRC 530
C                                                                       MPRC 540
C                                                                       MPRC 550
      DIMENSION A(1),ITRA(1)                                            MPRC 560
C                                                                       MPRC 570
C        TEST OF DIMENSIONS                                             MPRC 580
      IF(M)14,14,1                                                      MPRC 590
    1 IF(N)14,14,2                                                      MPRC 600
C                                                                       MPRC 610
C        DETERMINE WHICH ARE TO BE PERMUTED-THE ROWS OR THE COLUMNS     MPRC 620
    2 IF(IROCO)3,4,2                                                    MPRC 630
C                                                                       MPRC 640
C        INITIALIZE FOR COLUMN INTERCHANGES                             MPRC 650
    3 MM=M                                                              MPRC 660
      MMM=-1                                                            MPRC 670
      L=M                                                               MPRC 680
      LL=N                                                              MPRC 690
      GO TO 5                                                           MPRC 700
C                                                                       MPRC 710
C        INITIALIZE FOR ROW INTERCHANGES                                MPRC 720
    4 MM=1                                                              MPRC 730
      MMM=M                                                             MPRC 740
      L=N                                                               MPRC 750
      LL=M                                                              MPRC 760
C                                                                       MPRC 770
C        INITIALIZE LOOP OVER ALL ROWS OR COLUMNS                       MPRC 780
    5 IA=1                                                              MPRC 790
      ID=1                                                              MPRC 800
C                                                                       MPRC 810
C        TEST FOR INVERSE OPERATION                                     MPRC 820
      IF(INV)6,7,6                                                      MPRC 830
    6 IA=LL                                                             MPRC 840
      ID=-1                                                             MPRC 850
    7 DO 12 I=1,LL                                                      MPRC 860
      K=ITRA(IA)                                                        MPRC 870
      IF(K-IA)8,12,9                                                    MPRC 880
    8 IF(K)13,13,10                                                     MPRC 890
    9 IF(LL-K)13,10,10                                                  MPRC 900
C                                                                       MPRC 910
C        INITIALIZE ROW OR COLUMN INTERCHANGE                           MPRC 920
   10 IL=IA*MM                                                          MPRC 930
      K=K*MM                                                            MPRC 940
C                                                                       MPRC 950
C        PERFORM ROW OR COLUMN INTERCHANGE                              MPRC 960
      DO 11 J=1,L                                                       MPRC 970
      SAVE=A(IL)                                                        MPRC 980
      A(IL)=A(K)                                                        MPRC 990
      A(K)=SAVE                                                         MPRC1000
      K=K+MMM                                                           MPRC1010
   11 IL=IL+MMM                                                         MPRC1020
C                                                                       MPRC1030
C        ADDRESS NEXT INTERCHANGE STEP                                  MPRC1040
   12 IA=IA+ID                                                          MPRC1050
C                                                                       MPRC1060
C        NORMAL EXIT                                                    MPRC1070
      IER=0                                                             MPRC1080
      RETURN                                                            MPRC1090
C                                                                       MPRC1100
C        ERROR RETURN IN CASE ITRA IS NOT A TRANSPOSITION VECTOR        MPRC1110
   13 IER=1                                                             MPRC1120
      RETURN                                                            MPRC1130
C                                                                       MPRC1140
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSIONS                     MPRC1150
   14 IER=-1                                                            MPRC1160
      RETURN                                                            MPRC1170
      END                                                               MPRC1180
```

```
C                                                                       DMPR 420
C       SUBROUTINES AND SUBPROGRAMS REQUIRED                            DMPR 430
C          NONE                                                         DMPR 440
C                                                                       DMPR 450
C       METHOD                                                          DMPR 460
C          THE ROWS OR COLUMNS ARE PERMUTED ELEMENTWISE, INTERCHANGING  DMPR 470
C          ROW OR COLUMN 1 AND ITRA(1),....,ROW OR COLUMN K AND ITRA(K) DMPR 480
C          IN THAT ORDER IF INV=0, AND OTHERWISE INTERCHANGING ROW OR   DMPR 490
C          COLUMN K AND ITRA(K),....,ROW OR COLUMN 1 AND ITRA(1), WHERE DMPR 500
C          K IS M OR N DEPENDING ON WHETHER WE PERMUTE ROWS OR COLUMNS.DMPR 510
C                                                                       DMPR 520
C     ..................................................................DMPR 530
C                                                                       DMPR 540
      SUBROUTINE DMPRC(A,M,N,ITRA,INV,IROCO,IER)                        DMPR 550
C                                                                       DMPR 560
C                                                                       DMPR 570
      DIMENSION A(1),ITRA(1)                                            DMPR 580
      DOUBLE PRECISION A,SAVE                                           DMPR 590
C                                                                       DMPR 600
C        TEST OF DIMENSIONS                                             DMPR 610
      IF(M)14,14,1                                                      DMPR 620
    1 IF(N)14,14,2                                                      DMPR 630
C                                                                       DMPR 640
C        DETERMINE WHICH ARE TO BE PERMUTED-THE ROWS OR THE COLUMNS     DMPR 650
    2 IF(IROCO)3,4,3                                                    DMPR 660
C                                                                       DMPR 670
C        INITIALIZE FOR COLUMN INTERCHANGES                             DMPR 680
    3 MM=M                                                              DMPR 690
      MMM=-1                                                            DMPR 700
      L=M                                                               DMPR 710
      LL=N                                                              DMPR 720
      GO TO 5                                                           DMPR 730
C                                                                       DMPR 740
C        INITIALIZE FOR ROW INTERCHANGES                                DMPR 750
    4 MM=1                                                              DMPR 760
      MMM=M                                                             DMPR 770
      L=N                                                               DMPR 780
      LL=M                                                              DMPR 790
C                                                                       DMPR 800
C        INITIALIZE LOOP OVER ALL ROWS OR COLUMNS                       DMPR 810
    5 IA=1                                                              DMPR 820
      ID=1                                                              DMPR 830
C                                                                       DMPR 840
C        TEST FOR INVERSE OPERATION                                     DMPR 850
      IF(INV)6,7,6                                                      DMPR 860
    6 IA=LL                                                             DMPR 870
      ID=-1                                                             DMPR 880
    7 DO 12 I=1,LL                                                      DMPR 890
      K=ITRA(IA)                                                        DMPR 900
      IF(K-IA)8,12,9                                                    DMPR 910
    8 IF(K)13,13,10                                                     DMPR 920
    9 IF(LL-K)13,10,10                                                  DMPR 930
C                                                                       DMPR 940
C        INITIALIZE ROW OR COLUMN INTERCHANGE                           DMPR 950
   10 IL=IA*MM                                                          DMPR 960
      K=K*MM                                                            DMPR 970
C                                                                       DMPR 980
C        PERFORM ROW OR COLUMN INTERCHANGE                              DMPR 990
      DO 11 J=1,L                                                       DMPR1000
      SAVE=A(IL)                                                        DMPR1010
      A(IL)=A(K)                                                        DMPR1020
      A(K)=SAVE                                                         DMPR1030
      K=K+MMM                                                           DMPR1040
   11 IL=IL+MMM                                                         DMPR1050
C                                                                       DMPR1060
C        ADDRESS NEXT INTERCHANGE STEP                                  DMPR1070
   12 IA=IA+ID                                                          DMPR1080
C                                                                       DMPR1090
C        NORMAL EXIT                                                    DMPR1100
      IER=0                                                             DMPR1110
      RETURN                                                            DMPR1120
C                                                                       DMPR1130
C        ERROR RETURN IN CASE ITRA IS NOT A TRANSPOSITION VECTOR        DMPR1140
   13 IER=1                                                             DMPR1150
      RETURN                                                            DMPR1160
C                                                                       DMPR1170
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSIONS                     DMPR1180
   14 IER=-1                                                            DMPR1190
      RETURN                                                            DMPR1200
      END                                                               DMPR1210
```

```
C                                                                       DMPR  10
C     ..................................................................DMPR  20
C                                                                       DMPR  30
C       SUBROUTINE DMPRC                                                DMPR  40
C                                                                       DMPR  50
C       PURPOSE                                                         DMPR  60
C          TO PERMUTE THE ROWS OR COLUMNS OF A GIVEN MATRIX ACCORDING   DMPR  70
C          TO A GIVEN TRANSPOSITION VECTOR OR ITS INVERSE.  (SEE THE    DMPR  80
C          DISCUSSION ON PERMUTATIONS FOR DEFINITIONS AND NOTATION.)    DMPR  90
C                                                                       DMPR 100
C       USAGE                                                           DMPR 110
C          CALL DMPRC(A,M,N,ITRA,INV,IROCO,IER)                         DMPR 120
C                                                                       DMPR 130
C       DESCRIPTION OF PARAMETERS                                       DMPR 140
C          A     - GIVEN DOUBLE PRECISION M BY N MATRIX AND RESULTING   DMPR 150
C                  PERMUTED MATRIX                                      DMPR 160
C          M     - NUMBER OF ROWS OF A                                  DMPR 170
C          N     - NUMBER OF COLUMNS OF A                               DMPR 180
C          ITRA  - GIVEN TRANSPOSITION VECTOR (DIMENSION M IF ROWS ARE  DMPR 190
C                  PERMUTED, N IF COLUMNS ARE PERMUTED)                 DMPR 200
C          INV   - INPUT PARAMETER                                      DMPR 210
C                  INV NON-ZERO  -  PERMUTE ACCORDING TO ITRA           DMPR 220
C                  INV    =   0  -  PERMUTE ACCORDING TO ITRA INVERSE   DMPR 230
C          IROCO - INPUT PARAMETER                                      DMPR 240
C                  IROCO NON-ZERO  -  PERMUTE THE COLUMNS OF A          DMPR 250
C                  IROCO    =   0  -  PERMUTE THE ROWS OF A             DMPR 260
C          IER   - RESULTING ERROR PARAMETER                            DMPR 270
C                  IER = -1  -  M AND N ARE NOT BOTH POSITIVE           DMPR 280
C                  IER =  0  -  NO ERROR                                DMPR 290
C                  IER =  1  -  ITRA IS NOT A TRANSPOSITION VECTOR ON   DMPR 300
C                               1,...,M IF ROWS ARE PERMUTED, 1,...,N   DMPR 310
C                               IF COLUMNS ARE PERMUTED                 DMPR 320
C                                                                       DMPR 330
C       REMARKS                                                         DMPR 340
C          (1)  IF IER=-1 THERE IS NO COMPUTATION.                      DMPR 350
C          (2)  IF IER= 1, THEN COMPUTATION HAS BEEN UNSUCCESSFUL DUE   DMPR 360
C               TO ERROR, BUT THE MATRIX A WILL REFLECT THE ROW OR      DMPR 370
C               COLUMN INTERCHANGES PERFORMED BEFORE THE ERROR WAS      DMPR 380
C               DETECTED.                                               DMPR 390
C          (3)  THE MATRIX A IS ASSUMED TO BE STORED COLUMNWISE.        DMPR 400
C                                                                       DMPR 410
```

Subroutine MFUN

```
C                                                                       MFUN  10
C     ..................................................................MFUN  20
C                                                                       MFUN  30
C        SUBROUTINE MFUN                                                MFUN  40
C                                                                       MFUN  50
C        PURPOSE                                                        MFUN  60
C           APPLY A FUNCTION TO EACH ELEMENT OF A MATRIX TO FORM A      MFUN  70
C           RESULTANT MATRIX                                            MFUN  80
C                                                                       MFUN  90
C        USAGE                                                          MFUN 100
C           CALL MFUN (A,F,R,N,M,MS)                                    MFUN 110
C           AN EXTERNAL STATEMENT MUST PRECEDE CALL STATEMENT IN ORDER  MFUN 120
C           TO IDENTIFY PARAMETER F AS THE NAME OF A FUNCTION           MFUN 130
C                                                                       MFUN 140
C        DESCRIPTION OF PARAMETERS                                      MFUN 150
C           A - NAME OF INPUT MATRIX                                    MFUN 160
C           F - NAME OF FORTRAN-FURNISHED OR USER FUNCTION SUBPROGRAM   MFUN 170
C           R - NAME OF OUTPUT MATRIX                                   MFUN 180
C           N - NUMBER OF ROWS IN MATRIX A AND R                        MFUN 190
C           M - NUMBER OF COLUMNS IN MATRIX A AND R                     MFUN 200
C           MS  - ONE DIGIT NUMBER FOR STORAGE MODE OF MATRIX A (AND R) MFUN 210
C                 0 - GENERAL                                           MFUN 220
C                 1 - SYMMETRIC                                         MFUN 230
C                 2 - DIAGONAL                                          MFUN 240
C                                                                       MFUN 250
C        REMARKS                                                        MFUN 260
C           PRECISION IS DEPENDENT UPON PRECISION OF FUNCTION USED      MFUN 270
C                                                                       MFUN 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MFUN 290
C           LOC                                                         MFUN 300
C                                                                       MFUN 310
C        METHOD                                                         MFUN 320
C           FUNCTION F IS APPLIED TO EACH ELEMENT OF MATRIX A           MFUN 330
C           TO FORM MATRIX R                                            MFUN 340
C                                                                       MFUN 350
C     ..................................................................MFUN 360
C                                                                       MFUN 370
      SUBROUTINE MFUN(A,F,R,N,M,MS)                                     MFUN 380
      DIMENSION A(1),R(1)                                               MFUN 390
C                                                                       MFUN 400
C        COMPUTE VECTOR LENGTH, IT                                      MFUN 410
C                                                                       MFUN 420
      CALL LOC(N,M,IT,N,M,MS)                                           MFUN 430
C                                                                       MFUN 440
C        BUILD MATRIX R FOR ANY STORAGE MODE                            MFUN 450
C                                                                       MFUN 460
      DO 5 I=1,IT                                                       MFUN 470
    5 R(I)=F(A(I))                                                      MFUN 480
      RETURN                                                            MFUN 490
      END                                                               MFUN 500
```

Subroutine RECP

```
C                                                                       RECP  10
C     ..................................................................RECP  20
C                                                                       RECP  30
C        FUNCTION RECP                                                  RECP  40
C                                                                       RECP  50
C        PURPOSE                                                        RECP  60
C           CALCULATE RECIPROCAL OF AN ELEMENT. THIS IS A FORTRAN       RECP  70
C           FUNCTION SUBPROGRAM WHICH MAY BE USED AS AN ARGUMENT BY     RECP  80
C           SUBROUTINE MFUN.                                            RECP  90
C                                                                       RECP 100
C        USAGE                                                          RECP 110
C           RECP(E)                                                     RECP 120
C                                                                       RECP 130
C        DESCRIPTION OF PARAMETERS                                      RECP 140
C           E - MATRIX ELEMENT                                          RECP 150
C                                                                       RECP 160
C        REMARKS                                                        RECP 170
C           RECIPROCAL OF ZERO IS TAKEN TO BE 1.0E75                    RECP 180
C                                                                       RECP 190
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RECP 200
C           NONE                                                        RECP 210
C                                                                       RECP 220
C        METHOD                                                         RECP 230
C           RECIPROCAL OF ELEMENT E IS PLACED IN RECP                   RECP 240
C                                                                       RECP 250
C     ..................................................................RECP 260
C                                                                       RECP 270
      FUNCTION RECP(E)                                                  RECP 280
C                                                                       RECP 290
      BIG=1.0E75                                                        RECP 300
C                                                                       RECP 310
C        TEST ELEMENT FOR ZERO                                          RECP 320
C                                                                       RECP 330
      IF(E) 1,2,1                                                       RECP 340
C                                                                       RECP 350
C        IF NON-ZERO, CALCULATE RECIPROCAL                              RECP 360
C                                                                       RECP 370
    1 RECP=1.0/E                                                        RECP 380
      RETURN                                                            RECP 390
C                                                                       RECP 400
C        IF ZERO, SET EQUAL TO INFINITY                                 RECP 410
C                                                                       RECP 420
    2 RECP=SIGN(BIG,E)                                                  RECP 430
      RETURN                                                            RECP 440
      END                                                               RECP 450
```

Matrices: Inversion, Systems of Linear Equations and Related Topics

Subroutine MINV

```
C                                                                       MINV  10
C     ..................................................................MINV  20
C                                                                       MINV  30
C        SUBROUTINE MINV                                                MINV  40
C                                                                       MINV  50
C        PURPOSE                                                        MINV  60
C           INVERT A MATRIX                                             MINV  70
C                                                                       MINV  80
C        USAGE                                                          MINV  90
C           CALL MINV(A,N,D,L,M)                                        MINV 100
C                                                                       MINV 110
C        DESCRIPTION OF PARAMETERS                                      MINV 120
C           A - INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY  MINV 130
C               RESULTANT INVERSE.                                      MINV 140
C           N - ORDER OF MATRIX A                                       MINV 150
C           D - RESULTANT DETERMINANT                                   MINV 160
C           L - WORK VECTOR OF LENGTH N                                 MINV 170
C           M - WORK VECTOR OF LENGTH N                                 MINV 180
C                                                                       MINV 190
C        REMARKS                                                        MINV 200
C           MATRIX A MUST BE A GENERAL MATRIX                           MINV 210
C                                                                       MINV 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MINV 230
C           NONE                                                        MINV 240
C                                                                       MINV 250
C        METHOD                                                         MINV 260
C           THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT   MINV 270
C           IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT    MINV 280
C           THE MATRIX IS SINGULAR.                                     MINV 290
C                                                                       MINV 300
C     ..................................................................MINV 310
C                                                                       MINV 320
      SUBROUTINE MINV(A,N,D,L,M)                                        MINV 330
      DIMENSION A(1),L(1),M(1)                                          MINV 340
C                                                                       MINV 350
C        ...............................................................MINV 360
C                                                                       MINV 370
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  MINV 380
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      MINV 390
C        STATEMENT WHICH FOLLOWS.                                       MINV 400
C                                                                       MINV 410
C     DOUBLE PRECISION A,D,BIGA,HOLD                                    MINV 420
C                                                                       MINV 430
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    MINV 440
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      MINV 450
C        ROUTINE.                                                       MINV 460
C                                                                       MINV 470
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      MINV 480
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  ABS IN STATEMENT  MINV 490
C        10 MUST BE CHANGED TO DABS.                                    MINV 500
C                                                                       MINV 510
C        ...............................................................MINV 520
C                                                                       MINV 530
C        SEARCH FOR LARGEST ELEMENT                                     MINV 540
C                                                                       MINV 550
      D=1.0                                                             MINV 560
      NK=-N                                                             MINV 570
      DO 80 K=1,N                                                       MINV 580
      NK=NK+N                                                           MINV 590
      L(K)=K                                                            MINV 600
      M(K)=K                                                            MINV 610
      KK=NK+K                                                           MINV 620
      BIGA=A(KK)                                                        MINV 630
      DO 20 J=K,N                                                       MINV 640
      IZ=N*(J-1)                                                        MINV 650
      DO 20 I=K,N                                                       MINV 660
      IJ=IZ+I                                                           MINV 670
   10 IF( ABS(BIGA)- ABS(A(IJ))) 15,20,20                               MINV 680
   15 BIGA=A(IJ)                                                        MINV 690
      L(K)=I                                                            MINV 700
      M(K)=J                                                            MINV 710
   20 CONTINUE                                                          MINV 720
C                                                                       MINV 730
C        INTERCHANGE ROWS                                               MINV 740
C                                                                       MINV 750
      J=L(K)                                                            MINV 760
      IF(J-K) 35,35,25                                                  MINV 770
   25 KI=K-N                                                            MINV 780
      DO 30 I=1,N                                                       MINV 790
      KI=KI+N                                                           MINV 800
      HOLD=-A(KI)                                                       MINV 810
      JI=KI-K+J                                                         MINV 820
      A(KI)=A(JI)                                                       MINV 830
   30 A(JI) =HOLD                                                       MINV 840
C                                                                       MINV 850
C        INTERCHANGE COLUMNS                                            MINV 860
C                                                                       MINV 870
   35 I=M(K)                                                            MINV 880
      IF(I-K) 45,45,38                                                  MINV 890
   38 JP=N*(I-1)                                                        MINV 900
      DO 40 J=1,N                                                       MINV 910
      JK=NK+J                                                           MINV 920
      JI=JP+J                                                           MINV 930
      HOLD=-A(JK)                                                       MINV 940
      A(JK)=A(JI)                                                       MINV 950
   40 A(JI) =HOLD                                                       MINV 960
C                                                                       MINV 970
C        DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS        MINV 980
C        CONTAINED IN BIGA)                                             MINV 990
C                                                                       MINV1000
   45 IF(BIGA) 48,46,48                                                 MINV1010
   46 D=0.0                                                             MINV1020
      RETURN                                                            MINV1030
   48 DO 55 I=1,N                                                       MINV1040
      IF(I-K) 50,55,50                                                  MINV1050
   50 IK=NK+I                                                           MINV1060
      A(IK)=A(IK)/(-BIGA)                                               MINV1070
   55 CONTINUE                                                          MINV1080
C                                                                       MINV1090
C        REDUCE MATRIX                                                  MINV1100
C                                                                       MINV1110
      DO 65 I=1,N                                                       MINV1120
      IK=NK+I                                                           MINV1130
      HOLD=A(IK)                                                        MINV1140
      IJ=I-N                                                            MINV1150
      DO 65 J=1,N                                                       MINV1160
      IJ=IJ+N                                                           MINV1170
      IF(I-K) 60,65,60                                                  MINV1180
   60 IF(J-K) 62,65,62                                                  MINV1190
   62 KJ=IJ-I+K                                                         MINV1200
      A(IJ)=HOLD*A(KJ)+A(IJ)                                            MINV1210
   65 CONTINUE                                                          MINV1220
C                                                                       MINV1230
C        DIVIDE ROW BY PIVOT                                            MINV1240
C                                                                       MINV1250
      KJ=K-N                                                            MINV1260
      DO 75 J=1,N                                                       MINV1270
      KJ=KJ+N                                                           MINV1280
      IF(J-K) 70,75,70                                                  MINV1290
   70 A(KJ)=A(KJ)/BIGA                                                  MINV1300
   75 CONTINUE                                                          MINV1310
C                                                                       MINV1320
C        PRODUCT OF PIVOTS                                              MINV1330
C                                                                       MINV1340
      D=D*BIGA                                                          MINV1350
C                                                                       MINV1360
C        REPLACE PIVOT BY RECIPROCAL                                    MINV1370
C                                                                       MINV1380
      A(KK)=1.0/BIGA                                                    MINV1390
   80 CONTINUE                                                          MINV1400
C                                                                       MINV1410
C        FINAL ROW AND COLUMN INTERCHANGE                               MINV1420
C                                                                       MINV1430
      K=N                                                               MINV1440
  100 K=(K-1)                                                           MINV1450
      IF(K) 150,150,105                                                 MINV1460
  105 I=L(K)                                                            MINV1470
      IF(I-K) 120,120,108                                               MINV1480
  108 JQ=N*(K-1)                                                        MINV1490
      JR=N*(I-1)                                                        MINV1500
      DO 110 J=1,N                                                      MINV1510
      JK=JQ+J                                                           MINV1520
      HOLD=A(JK)                                                        MINV1530
      JI=JR+J                                                           MINV1540
      A(JK)=-A(JI)                                                      MINV1550
  110 A(JI) =HOLD                                                       MINV1560
  120 J=M(K)                                                            MINV1570
      IF(J-K) 100,100,125                                               MINV1580
  125 KI=K-N                                                            MINV1590
      DO 130 I=1,N                                                      MINV1600
      KI=KI+N                                                           MINV1610
      HOLD=A(KI)                                                        MINV1620
      JI=KI-K+J                                                         MINV1630
      A(KI)=-A(JI)                                                      MINV1640
  130 A(JI) =HOLD                                                       MINV1650
      GO TO 100                                                         MINV1660
  150 RETURN                                                            MINV1670
      END                                                               MINV1680
```

Subroutines SINV and DSINV

These subroutines invert a given symmetric positive definite matrix, using factorization by subroutines MFSD and DMFSD.

1. Mathematical background

Given an n by n symmetric positive definite matrix A, an upper triangular matrix $R = (r_{ij})$ exists such that

$$A = R^T R$$

Then $A^{-1} = R^{-1}(R^{-1})^T$. The elements $\bar{r}_{ik}$ of R^{-1} are computed using the following recursive formulas:

$$\bar{r}_{ik} = -(1/r_{ii}) \left(\sum_{m=i+1}^{k} r_{im} \bar{r}_{mk} \right) \qquad i < k$$

$$\bar{r}_{ik} = 1/r_{ii} \qquad i = k$$

$$\bar{r}_{ik} = 0 \qquad i > k$$

2. Programming considerations

The given symmetric positive definite matrix A is stored columnwise in compressed form, that is, only the upper triangular part in $\frac{n(n+1)}{2}$ successive storage locations.

Subroutine MFSD (DMFSD), which is called internally, returns the elements r_{ik} $(i \leqq k)$ of the upper triangular matrix R in the storage locations of A. The inverse (upper) triangular matrix R^{-1} and at last the resultant upper triangular part of the inverse matrix A^{-1} are calculated and stored columnwise in the same storage locations.

If any calculated radicand r_{kk}^2 $(k = 1, 2, 3, \ldots, N)$ is not positive, further calculation is bypassed, and the error parameter IER is set to -1. This means that matrix A is not positive definite, possibly due to roundoff errors. IER is also set to -1 if the input parameter N is less than one.

Let all radicands be positive, and let r_{kk}^2 be the first radicand which is no longer greater than the internal tolerance TOL = $|EPS \cdot a_{kk}|$. The subroutine gives the warning IER = k-1; however, calculation is continued. The warning indicates that there may be loss of significance at factorization step k due to loss of significant digits in the calculation of r_{kk}^2.

```
C                                                                       SINV  10
C     ..................................................................SINV  20
C                                                                       SINV  30
C        SUBROUTINE SINV                                                SINV  40
C                                                                       SINV  50
C        PURPOSE                                                        SINV  60
C           INVERT A GIVEN SYMMETRIC POSITIVE DEFINITE MATRIX           SINV  70
C                                                                       SINV  80
C        USAGE                                                          SINV  90
C           CALL SINV(A,N,EPS,IER)                                      SINV 100
C                                                                       SINV 110
C        DESCRIPTION OF PARAMETERS                                      SINV 120
C           A       - UPPER TRIANGULAR PART OF THE GIVEN SYMMETRIC      SINV 130
C                     POSITIVE DEFINITE N BY N COEFFICIENT MATRIX.      SINV 140
C                     ON RETURN A CONTAINS THE RESULTANT UPPER          SINV 150
C                     TRIANGULAR MATRIX.                                SINV 160
C           N       - THE NUMBER OF ROWS (COLUMNS) IN GIVEN MATRIX.     SINV 170
C           EPS     - AN INPUT CONSTANT WHICH IS USED AS RELATIVE       SINV 180
C                     TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.       SINV 190
C           IER     - RESULTING ERROR PARAMETER CODED AS FOLLOWS        SINV 200
C                     IER=0  - NO ERROR                                 SINV 210
C                     IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARAME- SINV 220
C                              TER N OR BECAUSE SOME RADICAND IS NON-   SINV 230
C                              POSITIVE (MATRIX A IS NOT POSITIVE       SINV 240
C                              DEFINITE, POSSIBLY DUE TO LOSS OF SIGNI- SINV 250
C                              FICANCE)                                 SINV 260
C                     IER=K  - WARNING WHICH INDICATES LOSS OF SIGNIFI- SINV 270
C                              CANCE. THE RADICAND FORMED AT FACTORIZA- SINV 280
C                              TION STEP K+1 WAS STILL POSITIVE BUT NO  SINV 290
C                              LONGER GREATER THAN ABS(EPS*A(K+1,K+1)). SINV 300
C                                                                       SINV 310
C        REMARKS                                                        SINV 320
C           THE UPPER TRIANGULAR PART OF GIVEN MATRIX IS ASSUMED TO BE  SINV 330
C           STORED COLUMNWISE IN N*(N+1)/2 SUCCESSIVE STORAGE LOCATIONS.SINV 340
C           IN THE SAME STORAGE LOCATIONS THE RESULTING UPPER TRIANGU-  SINV 350
C           LAR MATRIX IS STORED COLUMNWISE TOO.                        SINV 360
C           THE PROCEDURE GIVES RESULTS IF N IS GREATER THAN 0 AND ALL  SINV 370
C           CALCULATED RADICANDS ARE POSITIVE.                          SINV 380
C                                                                       SINV 390
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SINV 400
C           MFSD                                                        SINV 410
C                                                                       SINV 420
C        METHOD                                                         SINV 430
C           SOLUTION IS DONE USING THE FACTORIZATION BY SUBROUTINE MFSD.SINV 440
C                                                                       SINV 450
C     ..................................................................SINV 460
C                                                                       SINV 470
      SUBROUTINE SINV(A,N,EPS,IER)                                      SINV 480
C                                                                       SINV 490
C                                                                       SINV 500
      DIMENSION A(1)                                                    SINV 510
      DOUBLE PRECISION DIN,WORK                                         SINV 520
C                                                                       SINV 530
C        FACTORIZE GIVEN MATRIX BY MEANS OF SUBROUTINE MFSD             SINV 540
C        A = TRANSPOSE(T) * T                                           SINV 550
      CALL MFSD(A,N,EPS,IER)                                            SINV 560
      IF(IER) 9,1,1                                                     SINV 570
C                                                                       SINV 580
C        INVERT UPPER TRIANGULAR MATRIX T                               SINV 590
C        PREPARE INVERSION-LOOP                                         SINV 600
    1 IPIV=N*(N+1)/2                                                    SINV 610
      IND=IPIV                                                          SINV 620
C                                                                       SINV 630
C        INITIALIZE INVERSION-LOOP                                      SINV 640
      DO 6 I=1,N                                                        SINV 650
      DIN=1.D0/DBLE(A(IPIV))                                            SINV 660
      A(IPIV)=DIN                                                       SINV 670
      MIN=N                                                             SINV 680
      KEND=I-1                                                          SINV 690
      LANF=N-KEND                                                       SINV 700
      IF(KEND) 5,5,2                                                    SINV 710
    2 J=IND                                                             SINV 720
C                                                                       SINV 730
C        INITIALIZE ROW-LOOP                                            SINV 740
      DO 4 K=1,KEND                                                     SINV 750
      WORK=0.D0                                                         SINV 760
      MIN=MIN-1                                                         SINV 770
      LHOR=IPIV                                                         SINV 780
      LVER=J                                                            SINV 790
C                                                                       SINV 800
C        START INNER LOOP                                               SINV 810
      DO 3 L=LANF,MIN                                                   SINV 820
      LVEP=LVEP+1                                                       SINV 830
      LHOR=LHOR+L                                                       SINV 840
    3 WORK=WORK+DBLE(A(LVER)*A(LHOR))                                   SINV 850
C        END OF INNER LOOP                                              SINV 860
C                                                                       SINV 870
      A(J)=-WORK*DIN                                                    SINV 880
    4 J=J-MIN                                                           SINV 890
C        END OF ROW-LOOP                                                SINV 900
C                                                                       SINV 910
    5 IPIV=IPIV-MIN                                                     SINV 920
    6 IND=IND-1                                                         SINV 930
C        END OF INVERSION-LOOP                                          SINV 940
C                                                                       SINV 950
C        CALCULATE INVERSE(A) BY MEANS OF INVERSE(T)                    SINV 960
C        INVERSE(A) = INVERSE(T) * TRANSPOSE(INVERSE(T))                SINV 970
C        INITIALIZE MULTIPLICATION-LOOP                                 SINV 980
      DO 8 I=1,N                                                        SINV 990
      IPIV=IPIV+I                                                       SINV1000
      J=IPIV                                                            SINV1010
C                                                                       SINV1020
C        INITIALIZE ROW-LOOP                                            SINV1030
      DO 8 K=I,N                                                        SINV1040
      WORK=0.D0                                                         SINV1050
      LHOR=J                                                            SINV1060
C                                                                       SINV1070
C        START INNER LOOP                                               SINV1080
      DO 7 L=K,N                                                        SINV1090
      LVER=LHOR+K-I                                                     SINV1100
      WORK=WORK+DBLE(A(LHOR)*A(LVER))                                   SINV1110
    7 LHOR=LHOR+L                                                       SINV1120
C        END OF INNER LOOP                                              SINV1130
C                                                                       SINV1140
      A(J)=WORK                                                         SINV1150
    8 J=J+K                                                             SINV1160
C        END OF ROW- AND MULTIPLICATION-LOOP                            SINV1170
C                                                                       SINV1180
    9 RETURN                                                            SINV1190
      END                                                               SINV1200
```

```
*********************************************************************
      SUBROUTINE DSINV
      PURPOSE
         INVERT A GIVEN SYMMETRIC POSITIVE DEFINITE MATRIX
      USAGE
         CALL DSINV(A,N,EPS,IER)
      DESCRIPTION OF PARAMETERS
         A      - DOUBLE PRECISION UPPER TRIANGULAR PART OF GIVEN
                  SYMMETRIC POSITIVE DEFINITE N BY N COEFFICIENT
                  MATRIX.
                  ON RETURN A CONTAINS THE RESULTANT UPPER
                  TRIANGULAR MATRIX IN DOUBLE PRECISION.
         N      - THE NUMBER OF ROWS (COLUMNS) IN GIVEN MATRIX.
         EPS    - SINGLE PRECISION INPUT CONSTANT WHICH IS USED
                  AS RELATIVE TOLERANCE FOR TEST ON LOSS OF
                  SIGNIFICANCE.
         IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS
                  IER=0  - NO ERROR
                  IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARAME-
                           TER N OR BECAUSE SOME RADICAND IS NON-
                           POSITIVE (MATRIX A IS NOT POSITIVE
                           DEFINITE, POSSIBLY DUE TO LOSS OF SIGNI-
                           FICANCE)
                  IER=K  - WARNING WHICH INDICATES LOSS OF SIGNIFI-
                           CANCE. THE RADICAND FORMED AT FACTORIZA-
                           TION STEP K+1 WAS STILL POSITIVE BUT NO
                           LONGER GREATER THAN ABS(EPS*A(K+1,K+1)).
      REMARKS
         THE UPPER TRIANGULAR PART OF GIVEN MATRIX IS ASSUMED TO BE
         STORED COLUMNWISE IN N*(N+1)/2 SUCCESSIVE STORAGE LOCATIONS.
         IN THE SAME STORAGE LOCATIONS THE RESULTING UPPER TRIANGU-
         LAR MATRIX IS STORED COLUMNWISE TOO.
         THE PROCEDURE GIVES RESULTS IF N IS GREATER THAN 0 AND ALL
         CALCULATED RADICANDS ARE POSITIVE.
      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
         DMFSD
      METHOD
         SOLUTION IS DONE USING FACTORIZATION BY SUBROUTINE DMFSD.
*********************************************************************
      SUBROUTINE DSINV(A,N,EPS,IER)

      DIMENSION A(1)
      DOUBLE PRECISION A,DIN,WORK

         FACTORIZE GIVEN MATRIX BY MEANS OF SUBROUTINE DMFSD
         A = TRANSPOSE(T) * T
      CALL DMFSD(A,N,EPS,IER)
      IF(IER) 9,1,1

         INVERT UPPER TRIANGULAR MATRIX T
         PREPARE INVERSION-LOOP
    1 IPIV=N*(N+1)/2
      IND=IPIV

         INITIALIZE INVERSION-LOOP
      DO 6 I=1,N
      DIN=1.D0/A(IPIV)
      A(IPIV)=DIN
      MIN=N
      KEND=I-1
      LANF=N-KEND
      IF(KEND) 5,5,2
    2 J=IND

         INITIALIZE ROW-LOOP
      DO 4 K=1,KEND
      WORK=0.D0
      MIN=MIN-1
      LHOR=IPIV
      LVER=J

         START INNER LOOP
      DO 3 L=LANF,MIN
      LVER=LVER+1
      LHOR=LHOR+L
    3 WORK=WORK+A(LVER)*A(LHOR)
         END OF INNER LOOP

      A(J)=-WORK*DIN
    4 J=J-MIN
         END OF ROW-LOOP

    5 IPIV=IPIV-MIN
    6 IND=IND-1
         END OF INVERSION-LOOP

         CALCULATE INVERSE(A) BY MEANS OF INVERSE(T)
         INVERSE(A) = INVERSE(T) * TRANSPOSE(INVERSE(T))
         INITIALIZE MULTIPLICATION-LOOP
      DO 8 I=1,N
      IPIV=IPIV+I
      J=IPIV

         INITIALIZE ROW-LOOP
      DO 8 K=I,N
      WORK=0.D0
      LHOR=J

         START INNER LOOP
      DO 7 L=K,N
      LVER=LHOR+K-I
      WORK=WORK+A(LHOR)*A(LVER)
    7 LHOR=LHOR+L
         END OF INNER LOOP

      A(J)=WORK
    8 J=J+K
         END OF ROW- AND MULTIPLICATION-LOOP

    9 RETURN
      END
```

Subroutine SIMQ

```
C
C     ******************************************************************
C        SUBROUTINE SIMQ
C
C        PURPOSE
C           OBTAIN SOLUTION OF A SET OF SIMULTANEOUS LINEAR EQUATIONS,
C           AX=B
C
C        USAGE
C           CALL SIMQ(A,B,N,KS)
C
C        DESCRIPTION OF PARAMETERS
C           A - MATRIX OF COEFFICIENTS STORED COLUMNWISE.  THESE ARE
C                 DESTROYED IN THE COMPUTATION.  THE SIZE OF MATRIX A IS
C                 N BY N.
C           B - VECTOR OF ORIGINAL CONSTANTS (LENGTH N). THESE ARE
C                 REPLACED BY FINAL SOLUTION VALUES, VECTOR X.
C           N - NUMBER OF EQUATIONS AND VARIABLES. N MUST BE .GT. ONE.
C           KS - OUTPUT DIGIT
C                 0 FOR A NORMAL SOLUTION
C                 1 FOR A SINGULAR SET OF EQUATIONS
C
C        REMARKS
C           MATRIX A MUST BE GENERAL.
C           IF MATRIX IS SINGULAR , SOLUTION VALUES ARE MEANINGLESS.
C           AN ALTERNATIVE SOLUTION MAY BE OBTAINED BY USING MATRIX
C           INVERSION (MINV) AND MATRIX PRODUCT (GMPRD).
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           METHOD OF SOLUTION IS BY ELIMINATION USING LARGEST PIVOTAL
C           DIVISOR. EACH STAGE OF ELIMINATION CONSISTS OF INTERCHANGINGS
C           ROWS WHEN NECESSARY TO AVOID DIVISION BY ZERO OR SMALL
C           ELEMENTS.
C           THE FORWARD SOLUTION TO OBTAIN VARIABLE N IS DONE IN
C           N STAGES. THE BACK SOLUTION FOR THE OTHER VARIABLES IS
C           CALCULATED BY SUCCESSIVE SUBSTITUTIONS. FINAL SOLUTION
C           VALUES ARE DEVELOPED IN VECTOR B, WITH VARIABLE 1 IN B(1),
C           VARIABLE 2 IN B(2),........., VARIABLE N IN B(N).
C           IF NO PIVOT CAN BE FOUND EXCEEDING A TOLERANCE OF 0.0,
C           THE MATRIX IS CONSIDERED SINGULAR AND KS IS SET TO 1. THIS
C           TOLERANCE CAN BE MODIFIED BY REPLACING THE FIRST STATEMENT.
C
C     ******************************************************************
C
      SUBROUTINE SIMQ(A,B,N,KS)
      DIMENSION A(1),B(1)
C
C        FORWARD SOLUTION
C
      TOL=0.0
      KS=0
      JJ=-N
      DO 65 J=1,N
      JY=J+1
      JJ=JJ+N+1
      BIGA=0
      IT=JJ-J
      DO 30 I=J,N
C
C        SEARCH FOR MAXIMUM COEFFICIENT IN COLUMN
C
      IJ=IT+I
      IF(ABS(BIGA)-ABS(A(IJ))) 20,30,30
   20 BIGA=A(IJ)
      IMAX=I
   30 CONTINUE
C
C        TEST FOR PIVOT LESS THAN TOLERANCE (SINGULAR MATRIX)
C
      IF(ABS(BIGA)-TOL) 35,35,40
   35 KS=1
      RETURN
C
C        INTERCHANGE ROWS IF NECESSARY
C
   40 I1=J+N*(J-2)
      IT=IMAX-J
      DO 50 K=J,N
      I1=I1+N
      I2=I1+IT
      SAVE=A(I1)
      A(I1)=A(I2)
      A(I2)=SAVE
C
C        DIVIDE EQUATION BY LEADING COEFFICIENT
C
   50 A(I1)=A(I1)/BIGA
      SAVE=B(IMAX)
      B(IMAX)=B(J)
      B(J)=SAVE/BIGA
C
C        ELIMINATE NEXT VARIABLE
C
      IF(J-N) 55,70,55
   55 IQS=N*(J-1)
      DO 65 IX=JY,N
      IXJ=IQS+IX
      IT=J-IX
      DO 60 JX=JY,N
      IXJX=N*(JX-1)+IX
      JJX=IXJX+IT
   60 A(IXJX)=A(IXJX)-(A(IXJ)*A(JJX))
   65 B(IX)=B(IX)-(B(J)*A(IXJ))
C
C        BACK SOLUTION
C
   70 NY=N-1
      IT=N*N
      DO 80 J=1,NY
      IA=IT-J
      IB=N-J
      IC=N
      DO 80 K=1,J
      B(IB)=B(IB)-A(IA)*B(IC)
      IA=IA-N
   80 IC=IC-1
      RETURN
      END
```

Subroutines GELG and DGELG

These subroutines solve a system of general simultaneous linear equations by Gauss elimination. Consider the system of general simultaneous linear equations:

$$A * X = R \tag{1}$$

with an m by m coefficient matrix A and an m by n right-hand side matrix R both stored columnwise. Solution is done by means of Gauss elimination with complete pivoting. If matrix R is the identity matrix, solution X is the inverse of matrix A. The solution X is generated in matrix R. Thus, the computation of the solution requires no additional storage requirements.

Explicitly, the given system (1) is of the form:

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mm} \end{pmatrix} * \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

$$= \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ r_{m1} & r_{m2} & \cdots & r_{mn} \end{pmatrix} \tag{2}$$

The first step is to search the whole matrix A for the element of greatest absolute value, say a_{ij}, and to select it as first pivot ($p = a_{ij}$). With a_{ij}, generate the internal absolute tolerance for testing loss of significance in the following way:

$$\text{tol} = |a_{ij}| \cdot \varepsilon$$

with a given relative tolerance ε.

Suppose that pivot a_{ij} is equal to a_{11}. If it is not, interchange the first rows of matrices A and R with the ith, and the first column of matrix A with the jth, and save column interchange information by storing the difference (j-1) of pivot column index j and step counter k = 1 [interchanging column 1 with column j means interchanging of variables x_{11} with x_{lj} (l = 1, 2, ..., n)].

Now transform the elements of pivot rows in matrices A and R by multiplying with 1/p, and the other elements by adding $-a_{\nu 1}$ times the new first rows of these two matrices to the other ν rows, thus getting:

$$a_{11}^{(1)} = + \frac{a_{11}}{p} \quad (l = 2, 3, \ldots, m) \tag{4}$$

$$r_{11}^{(1)} = + \frac{r_{11}}{p} \quad (l = 1, 2, \ldots, n) \tag{5}$$

$$a_{\nu l}^{(1)} = a_{\nu l} - a_{\nu 1} \cdot a_{11}^{(1)} \quad (l = 2, 3, \ldots, m;\ \nu = 2, 3, \ldots, m) \tag{6}$$

$$r_{\nu l}^{(1)} = r_{\nu l} - a_{\nu 1} \cdot r_{11}^{(1)} \quad (l = 1, 2, \ldots, n;\ \nu = 2, 3, \ldots, m) \tag{7}$$

If column interchange information is saved in the first position of the main diagonal, the result of the first step is the two matrices:

$$A^{(1)} = \begin{pmatrix} (j-1) & a_{12}^{(1)} & a_{13}^{(1)} & \cdots & a_{1m}^{(1)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2m}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(1)} & \cdots & a_{3m}^{(1)} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & a_{m2}^{(1)} & a_{m3}^{(1)} & \cdots & a_{mm}^{(1)} \end{pmatrix} \text{ and}$$

$$R^{(1)} = \begin{pmatrix} r_{11}^{(1)} & r_{12}^{(1)} & \cdots & r_{1n}^{(1)} \\ r_{21}^{(1)} & r_{22}^{(1)} & \cdots & r_{2n}^{(1)} \\ \cdots & \cdots & \cdots & \cdots \\ r_{m1}^{(1)} & r_{m2}^{(1)} & \cdots & r_{mn}^{(1)} \end{pmatrix}$$

Now repeat this procedure m-2 times, starting at each step with the matrix $A^{(k)}$ of the step before

without first k rows and first k columns, and the matrix $R^{(k)}$ without first k rows. The total result after m-1 steps is the matrices:

$$A^{(m-1)} = \begin{pmatrix} (j_1-1) & a_{12}^{(1)} & a_{13}^{(1)} & \dots & a_{1m}^{(1)} \\ 0 & (j_2-2) & a_{23}^{(2)} & \dots & a_{2m}^{(2)} \\ 0 & 0 & (j_3-3) & \dots & a_{3m}^{(3)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & & (j_m-m) \end{pmatrix} \quad \text{and}$$

$$R^{(m-1)} = \begin{pmatrix} r_{11}^{(1)} & r_{12}^{(1)} & \dots & r_{1n}^{(1)} \\ r_{21}^{(2)} & r_{22}^{(2)} & \dots & r_{2n}^{(2)} \\ r_{31}^{(3)} & r_{32}^{(3)} & \dots & r_{3n}^{(3)} \\ \dots & \dots & \dots & \dots \\ r_{m1}^{(m)} & r_{m2}^{(m)} & \dots & r_{mn}^{(m)} \end{pmatrix}$$

Now work backward and set:

$$\left.\begin{aligned} r_{m-1,l}^{(m)} &= r_{m-1,l}^{(m-1)} - a_{m-1,m}^{(m-1)} \cdot r_{m,l}^{(m)} \quad (l = 1,2,\dots,n) \\ r_{m-2,l}^{(m)} &= r_{m-2,l}^{(m-2)} - a_{m-2,m-1}^{(m-2)} \cdot r_{m-1,l}^{(m)} \\ &\quad - a_{m-2,m}^{(m-2)} \cdot r_{m,l}^{(m)} \quad (l = 1,2,\dots,n) \\ &\dots\dots\dots\dots\dots\dots\dots\dots \\ r_{1,l}^{(m)} &= r_{1l}^{(1)} - a_{12}^{(1)} \cdot r_{2l}^{(m)} - a_{13}^{(1)} \cdot r_{3l}^{(m)} \\ &\quad - \dots - a_{1m}^{(1)} \cdot r_{ml}^{(m)} \quad (l = 1,2,\dots,n) \end{aligned}\right\} \quad (8)$$

After each step of back substitution, rows of solution matrix $X = R^{(m)}$ have to be back-interchanged according to interchange information in the corresponding main diagonal element of matrix $A^{(m-1)}$, in order to get the correct sequence of right-hand side column elements $r_{\nu l}^{(m)}$ corresponding to the sequence of left-hand side column elements $a_{j\nu}^{(j)}$.

The only case in which the above procedure can fail to give a solution occurs when at any step all elements of the rest-matrix of $A^{(k)}$ become zero, and no pivot element can be found. In this case the procedure is bypassed, and the error message ier = -1 is given. Actually, because of rounding errors, a further check of the absolute values of pivot elements is performed by the procedure. If at elimination step k this absolute value becomes less than tol (see equation 3), it is likely that matrix A is singular too. But as this is not necessarily the case, and as this test depends highly on the choice of the relative tolerance ε+ , the procedure gives only the warning ier = k-1, indicating that there is a possible loss of significance in the results computed by the algorithm++. In case of a well scaled matrix A and an appropriate choice of the relative tolerance, warning ier = k-1 may be interpreted to mean that matrix A has the rank k-1. If there is only one equation to solve (m=1), the test on loss of significance is suppressed.

+For subroutine GELG, a relative tolerance ϵ between 10^{-6} and 10^{-7} is suggested; and for subroutine DGELG, between 10^{-14} and 10^{-16}.

++For example, $\epsilon = 10^{-5}$ and warning ier = 3 mean that there is a possible loss of about five or more significant digits in the initial values of elimination step 4, and that matrix A seems to have the rank 3.

```
C
C     ..................................................................
C
C        SUBROUTINE GELG
C
C        PURPOSE
C           TO SOLVE A GENERAL SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS.
C
C        USAGE
C           CALL GELG(R,A,M,N,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           R      - THE M BY N MATRIX OF RIGHT HAND SIDES.  (DESTROYED)
C                    ON RETURN R CONTAINS THE SOLUTION OF THE EQUATIONS.
C           A      - THE M BY M COEFFICIENT MATRIX.  (DESTROYED)
C           M      - THE NUMBER OF EQUATIONS IN THE SYSTEM.
C           N      - THE NUMBER OF RIGHT HAND SIDE VECTORS.
C           EPS    - AN INPUT CONSTANT WHICH IS USED AS RELATIVE
C                    TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.
C           IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS
C                    IER=0  - NO ERROR,
C                    IER=-1 - NO RESULT BECAUSE OF M LESS THAN 1 OR
C                             PIVOT ELEMENT AT ANY ELIMINATION STEP
C                             EQUAL TO 0,
C                    IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-
C                             CANCE INDICATED AT ELIMINATION STEP K+1,
C                             WHERE PIVOT ELEMENT WAS LESS THAN OR
C                             EQUAL TO THE INTERNAL TOLERANCE EPS TIMES
C                             ABSOLUTELY GREATEST ELEMENT OF MATRIX A.
C
C        REMARKS
C           INPUT MATRICES R AND A ARE ASSUMED TO BE STORED COLUMNWISE
C           IN M*N RESP. M*M SUCCESSIVE STORAGE LOCATIONS. ON RETURN
C           SOLUTION MATRIX R IS STORED COLUMNWISE TOO.
C           THE PROCEDURE GIVES RESULTS IF THE NUMBER OF EQUATIONS M IS
C           GREATER THAN 0 AND PIVOT ELEMENTS AT ALL ELIMINATION STEPS
C           ARE DIFFERENT FROM 0. HOWEVER WARNING IER=K - IF GIVEN -
C           INDICATES POSSIBLE LOSS OF SIGNIFICANCE. IN CASE OF A WELL
C           SCALED MATRIX A AND APPROPRIATE TOLERANCE EPS, IER=K MAY BE
C           INTERPRETED THAT MATRIX A HAS THE RANK K. NO WARNING IS
C           GIVEN IN CASE M=1.
```

```
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SOLUTION IS DONE BY MEANS OF GAUSS-ELIMINATION WITH
C           COMPLETE PIVOTING.
C
C     ..................................................................
C
      SUBROUTINE GELG(R,A,M,N,EPS,IER)
C
C
      DIMENSION A(1),R(1)
      IF(M)23,23,1
C
C     SEARCH FOR GREATEST ELEMENT IN MATRIX A
    1 IER=0
      PIV=0.
      MM=M*M
      NM=N*M
      DO 3 L=1,MM
      TB=ABS(A(L))
      IF(TB-PIV)3,3,2
    2 PIV=TB
      I=L
    3 CONTINUE
      TOL=EPS*PIV
      A(I) IS PIVOT ELEMENT. PIV CONTAINS THE ABSOLUTE VALUE OF A(I).
C
C
C
C     START ELIMINATION LOOP
      LST=1
      DO 17 K=1,M
C
C     TEST ON SINGULARITY
      IF(PIV)23,23,4
    4 IF(IER)7,5,7
    5 IF(PIV-TOL)6,6,7
    6 IER=K-1
    7 PIVI=1./A(I)
      J=(I-1)/M
      I=I-J*M-K
      J=J+1-K
C     I+K IS ROW-INDEX, J+K COLUMN-INDEX OF PIVOT ELEMENT
C
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R
      DO 8 L=K,NM,M
      LL=L+I
      TB=PIVI*R(LL)
      R(LL)=R(L)
    8 R(L)=TB
C
C     IS ELIMINATION TERMINATED
      IF(K-M)9,18,18
C
C     COLUMN INTERCHANGE IN MATRIX A
    9 LEND=LST+M-K
      IF(J)12,12,10
   10 II=J*M
      DO 11 L=LST,LEND
      TB=A(L)
      LL=L+II
      A(L)=A(LL)
   11 A(LL)=TB
C
C     ROW INTERCHANGE AND PIVOT ROW REDUCTION IN MATRIX A
   12 DO 13 L=LST,MM,M
      LL=L+I
      TB=PIVI*A(LL)
      A(LL)=A(L)
   13 A(L)=TB
C
C     SAVE COLUMN INTERCHANGE INFORMATION
      A(LST)=J
C
C     ELEMENT REDUCTION AND NEXT PIVOT SEARCH
      PIV=0.
      LST=LST+1
      J=0
      DO 15 II=LST,LEND
      PIVI=-A(II)
      IST=II+M
      J=J+1
      DO 15 L=IST,MM,M
      LL=L-J
      A(L)=A(L)+PIVI*A(LL)
      TB=ABS(A(L))
      IF(TB-PIV)15,15,14
   14 PIV=TB
      I=L
   15 CONTINUE
      DO 16 L=K,NM,M
      LL=L+J
   16 R(LL)=R(LL)+PIVI*R(L)
   17 LST=LST+M
C     END OF ELIMINATION LOOP
C
C
C
C     BACK SUBSTITUTION AND BACK INTERCHANGE
   18 IF(M-1)23,22,19
   19 IST=MM+M
      LST=M+1
      DO 21 I=2,M
      II=LST-1
      IST=IST-LST
      L=IST-M
      L=A(L)+.5
      DO 21 J=II,NM,M
      TB=R(J)
      LL=J
      DO 20 K=IST,MM,M
      LL=LL+1
   20 TB=TB-A(K)*R(LL)
      K=J+L
      R(J)=R(K)
   21 R(K)=TB
   22 RETURN
C
C
C     ERROR RETURN
   23 IER=-1
      RETURN
      END
```

```
C                                                                      DELG  10
C     ..................................................................DELG  20
C                                                                      DELG  30
C        SUBROUTINE DGELG                                              DELG  40
C                                                                      DELG  50
C        PURPOSE                                                       DELG  60
C           TO SOLVE A GENERAL SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS. DELG  70
C                                                                      DELG  80
C        USAGE                                                         DELG  90
C           CALL DGELG(R,A,M,N,EPS,IER)                                DELG 100
C                                                                      DELG 110
C        DESCRIPTION OF PARAMETERS                                     DELG 120
C           R      - DOUBLE PRECISION M BY N RIGHT HAND SIDE MATRIX    DELG 130
C                    (DESTROYED). ON RETURN R CONTAINS THE SOLUTIONS   DELG 140
C                    OF THE EQUATIONS.                                 DELG 150
C           A      - DOUBLE PRECISION M BY M COEFFICIENT MATRIX        DELG 160
C                    (DESTROYED).                                      DELG 170
C           M      - THE NUMBER OF EQUATIONS IN THE SYSTEM.            DELG 180
C           N      - THE NUMBER OF RIGHT HAND SIDE VECTORS.            DELG 190
C           EPS    - SINGLE PRECISION INPUT CONSTANT WHICH IS USED AS  DELG 200
C                    RELATIVE TOLERANCE FOR TEST ON LOSS OF            DELG 210
C                    SIGNIFICANCE.                                     DELG 220
C           IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS        DELG 230
C                    IER=0  - NO ERROR.                                DELG 240
C                    IER=-1 - NO RESULT BECAUSE OF M LESS THAN 1 OR    DELG 250
C                             PIVOT ELEMENT AT ANY ELIMINATION STEP    DELG 260
C                             EQUAL TO 0.                              DELG 270
C                    IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI- DELG 280
C                             CANCE INDICATED AT ELIMINATION STEP K+1, DELG 290
C                             WHERE PIVOT ELEMENT WAS LESS THAN OR     DELG 300
C                             EQUAL TO THE INTERNAL TOLERANCE EPS TIMES DELG 310
C                             ABSOLUTELY GREATEST ELEMENT OF MATRIX A. DELG 320
C                                                                      DELG 330
C        REMARKS                                                       DELG 340
C           INPUT MATRICES R AND A ARE ASSUMED TO BE STORED COLUMNWISE DELG 350
C           IN M*N RESP. M*M SUCCESSIVE STORAGE LOCATIONS. ON RETURN   DELG 360
C           SOLUTION MATRIX R IS STORED COLUMNWISE TOO.                DELG 370
C           THE PROCEDURE GIVES RESULTS IF THE NUMBER OF EQUATIONS M IS DELG 380
C           GREATER THAN 0 AND PIVOT ELEMENTS AT ALL ELIMINATION STEPS DELG 390
C           ARE DIFFERENT FROM 0. HOWEVER WARNING IER=K - IF GIVEN -   DELG 400
C           INDICATES POSSIBLE LOSS OF SIGNIFICANCE. IN CASE OF A WELL DELG 410
C           SCALED MATRIX A AND APPROPRIATE TOLERANCE EPS, IER=K MAY BE DELG 420
C           INTERPRETED THAT MATRIX A HAS THE RANK K. NO WARNING IS    DELG 430
C           GIVEN IN CASE M=1.                                         DELG 440
C                                                                      DELG 450
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                 DELG 460
C           NONE                                                       DELG 470
C                                                                      DELG 480
C        METHOD                                                        DELG 490
C           SOLUTION IS DONE BY MEANS OF GAUSS-ELIMINATION WITH        DELG 500
C           COMPLETE PIVOTING.                                         DELG 510
C                                                                      DELG 520
C     ..................................................................DELG 530
C                                                                      DELG 540
C                                                                      DELG 550
      SUBROUTINE DGELG(R,A,M,N,EPS,IER)                                DELG 560
C                                                                      DELG 570
C                                                                      DELG 580
      DIMENSION A(1),R(1)                                              DELG 590
      DOUBLE PRECISION R,A,PIV,TB,TOL,PIVI                             DELG 600
      IF(M)23,23,1                                                     DELG 610
C                                                                      DELG 620
C     SEARCH FOR GREATEST ELEMENT IN MATRIX A                          DELG 630
    1 IER=0                                                            DELG 640
      PIV=0.D0                                                         DELG 650
      MM=M*M                                                           DELG 660
      NM=N*M                                                           DELG 670
      DO 3 L=1,MM                                                      DELG 680
      TB=DABS(A(L))                                                    DELG 690
      IF(TB-PIV)3,3,2                                                  DELG 700
    2 PIV=TB                                                           DELG 710
      I=L                                                              DELG 720
    3 CONTINUE                                                         DELG 730
      TOL=EPS*PIV                                                      DELG 740
C     A(I) IS PIVOT ELEMENT. PIV CONTAINS THE ABSOLUTE VALUE OF A(I).  DELG 750
C                                                                      DELG 760
C                                                                      DELG 770
C     START ELIMINATION LOOP                                           DELG 780
      LST=1                                                            DELG 790
      DO 17 K=1,M                                                      DELG 800
C                                                                      DELG 810
C     TEST ON SINGULARITY                                              DELG 820
      IF(PIV)23,23,4                                                   DELG 830
    4 IF(IER)7,5,7                                                     DELG 840
    5 IF(PIV-TOL)6,6,7                                                 DELG 850
    6 IER=K-1                                                          DELG 860
    7 PIVI=1.D0/A(I)                                                   DELG 870
      J=(I-1)/M                                                        DELG 880
      I=I-J*M-K                                                        DELG 890
      J=J+1-K                                                          DELG 900
C     I+K IS ROW-INDEX, J+K COLUMN-INDEX OF PIVOT ELEMENT              DELG 910
C                                                                      DELG 920
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R     DELG 930
      DO 8 L=K,NM,M                                                    DELG 940
      LL=L+I                                                           DELG 950
      TB=PIVI*R(LL)                                                    DELG 960
      R(LL)=R(L)                                                       DELG 970
    8 R(L)=TB                                                          DELG 980
C                                                                      DELG 990
C     IS ELIMINATION TERMINATED                                        DELG1000
      IF(K-M)9,18,18                                                   DELG1010
C                                                                      DELG1020
C     COLUMN INTERCHANGE IN MATRIX A                                   DELG1030
    9 LEND=LST+M-K                                                     DELG1040
      IF(J)12,12,10                                                    DELG1050
   10 II=J*M                                                           DELG1060
      DO 11 L=LST,LEND                                                 DELG1070
      TB=A(L)                                                          DELG1080
      LL=L+II                                                          DELG1090
      A(L)=A(LL)                                                       DELG1100
   11 A(LL)=TB                                                         DELG1110
C                                                                      DELG1120
C     ROW INTERCHANGE AND PIVOT ROW REDUCTION IN MATRIX A              DELG1130
   12 DO 13 L=LST,MM,M                                                 DELG1140
      LL=L+I                                                           DELG1150
      TB=PIVI*A(LL)                                                    DELG1160
      A(LL)=A(L)                                                       DELG1170
   13 A(L)=TB                                                          DELG1180
C                                                                      DELG1190
C     SAVE COLUMN INTERCHANGE INFORMATION                              DELG1200
      A(LST)=J                                                         DELG1210
C                                                                      DELG1220
C     ELEMENT REDUCTION AND NEXT PIVOT SEARCH                          DELG1230
      PIV=0.D0                                                         DELG1240
      LST=LST+1                                                        DELG1250
      J=0                                                              DELG1260
      DO 16 II=LST,LEND                                                DELG1270
      PIVI=-A(II)                                                      DELG1280
      IST=II+M                                                         DELG1290
      J=J+1
```

```
      DO 15 L=IST,MM,M
      LL=L-J
      A(L)=A(L)+PIVI*A(LL)
      TB=DABS(A(L))
      IF(TB-PIV)15,15,14
   14 PIV=TB
      I=L
   15 CONTINUE
      DO 16 L=K,MM,M
      LL=L+J
   16 A(LL)=R(LL)+PIVI*R(L)
   17 LST=LST+M
C     END OF ELIMINATION LOOP
C
C
C     BACK SUBSTITUTION AND BACK INTERCHANGE
   18 IF(M-1)23,22,19
   19 IST=MM+M
      LST=M+1
      DO 21 I=2,M
      II=LST-1
      ISF=IST-LST
      L=IST-M
      L=A(L)+.5D
      DO 21 J=II,MM,M
      TB=R(J)
      LL=J
      DO 20 K=IST,MM,M
      LL=LL+1
   20 TB=TB-A(K)*R(LL)
      K=J+L
      R(J)=R(K)
   21 R(K)=TB
   22 RETURN
C
C
C     ERROR RETURN
   23 IER=-1
      RETURN
      END
```

Subroutine RSLMC

This subroutine computes an approximate solution to a system of linear equations when the coefficient matrix has been factored into a product of two triangular matrices.

1. Mathematical background

Consider a system of linear equations

$$Ax = b \tag{1}$$

having a nonsingular coefficient matrix A. There exists a permutation P such that PA = LU, where L is lower triangular with unit diagonal, and U is upper triangular. Putting c = Pb we see that (1) is equivalent to the system

$$\begin{aligned} Ux &= y \\ Ly &= c \end{aligned} \tag{2}$$

Let $x^{(p)}$ be a trial solution of the given system. It can be improved by means of the iterative process

$$\begin{aligned} r^{(p)} &= b - Ax^{(p)} \\ Ad^{(p)} &= r^{(p)} \\ x^{(p+1)} &= x^{(p)} + d^{(p)} \end{aligned} \tag{3}$$

where the correction vector $d^{(p)}$ is computed using L and U as in (2).

If A is not too ill-conditioned, this process will give a satisfactory solution, provided that $r^{(p)}$, the residual vector, is computed with sufficient accuracy.

2. Programming considerations

This routine may be used to solve a nonsingular system of linear equations whose coefficient matrix has been factored by means of subroutine FACTR.

Let $\epsilon > 0$ be a relative precision parameter (input to RSLMC). In the following discussion we use the notation $\|x\| = \Sigma |x_i|$. The process (3) is applied to the system (1) starting with $x^{(o)} = 0$, and is stopped when one of the following situations occurs:

a. IER = 0. The components of the correction vector $d^{(p)}$ satisfy the inequalities

$$\left| d_i^{(p)} \right| \le \epsilon \quad \left| x_i^{(p)} \right| \qquad i = 1, 2, \ldots, n. \tag{4}$$

Then $x = x^{(p)}$ is returned as the solution.

b. Assume that $p > 1$, for some i the inequality (4) fails and

$$\left\| d^{(k)} \right\| \le \frac{1}{2} \left\| d^{(k-1)} \right\| \qquad k = 1, 2, \ldots, p-1$$

$$\left\| d^{(p)} \right\| > \frac{1}{2} \left\| d^{(p-1)} \right\|$$

Then:

$$\text{IER} = 1 \text{ if } \quad \left\| d^{(p)} \right\| \le \epsilon \left\| x^{(p)} \right\| ,$$

$$\text{IER} = 2 \text{ if } \quad \left\| d^{(p)} \right\| > \epsilon \left\| x^{(p)} \right\| .$$

c. IER = 3. $\left\| d^{(1)} \right\| > \frac{1}{2} \left\| d^{(o)} \right\|$. In this case the trial solution is too far from the exact solution and the process cannot converge. In general this happens when A is badly ill-conditioned.

IER = 4. U is singular.

In all cases $x = x^{(p)}$ is returned.

Note: To obtain accuracy in the computation of the residual vector, each component is computed in double-precision arithmetic.

For reference see:

(1) H. J. Bowdler, R. S. Martin, G. Peters and J. H. Wilkinson, "Solution of Real and Complex Systems of Linear Equations", Numerische Mathematik, vol. 8, no. 3, 1966.

(2) J. H. Wilkinson, The Algebraic Eigenvalue Problem. Clarendon Press, Oxford, 1965.


```
C
C
C
```

Subroutine FACTR

This subroutine determines a decomposition of a real nonsingular matrix as a product of a lower triangular matrix with unit diagonal and an upper triangular matrix.

1. Mathematical background

Let A be a real nonsingular matrix of order n. In general, A can be factored into a product of the form A = L U, where L is lower triangular with unit diagonal and U is upper triangular.

The elements of L and U may be computed recursively as follows:

$$\bar{a}_{ip} = a_{ip} - \sum_{k=1}^{p-1} l_{ik} u_{kp} \qquad i=p, p+1, \ldots, n$$

$$l_{ip} = \bar{a}_{ip} \Big/ \bar{a}_{pp} \qquad i=p, p+1, \ldots, n$$

$$u_{pp} = \bar{a}_{pp}$$

$$u_{pj} = a_{pj} - \sum_{k=1}^{p-1} l_{pk} u_{kj} \qquad j=p+1, p+2, \ldots, n$$

Using these formulas for $p=1,2,\ldots,n$, the decomposition can be carried out in n stages.

2. Programming considerations

Even if A is nonsingular and well-conditioned, the above process can fail when a leading principal submatrix of A is singular; furthermore, the process is numerically unstable whenever a leading principal submatrix is ill-conditioned.

In order to avoid these inconveniences, a technique of partial pivoting combined with an equilibration of the matrix has been built into the algorithm.

Initially, the maximum norm of each row vector is computed:

$$N_p = \max_j \left| a_{pj} \right| \qquad p=1,2,\ldots,n$$

If any $N_p = 0$, the error parameter IER is set to 3 and further calculations are bypassed. Otherwise the reciprocals $1/N_p$ are computed and stored in the vector PER.

Then, assuming that the first p-1 columns of L and p-1 rows of U have been computed, the pth stage of the factorization process is:

a. Computation of the $\bar{a}_{ip}$.

$$\bar{a}_{ip} = a_{ip} - \sum_{k=1}^{p-1} l_{ik} u_{kp} \qquad i=p,p+1,\ldots,n$$

Each term $\bar{a}_{ip}$ replaces the corresponding term of A in storage.

b. Equilibrated partial pivoting.

Let m be the smallest row subscript such that

$$(1/N_p) \left| \bar{a}_{mp} \right| = \max_{i \geq p} \left\{ (1/N_i) \left| \bar{a}_{ip} \right| \right\}$$

If $\bar{a}_{mp} = 0$, the error message IER = 3 is given and further calculations are bypassed. If $\bar{a}_{mp} \neq 0$, the mth and p^{th} rows of A are interchanged, PER(p) is moved to PER(m), and the number m is stored in PER(p). Thus the first p positions of PER contain the information on the permutations needed so far in the form of a transposition vector.

c. Computation of the pth column of L.

$$l_{ip} = \bar{a}_{ip} / \bar{a}_{pp} \qquad i=p+1, p+2, \ldots, n$$

These terms of L replace the corresponding terms of A in storage. The diagonal terms of L, which are all equal to 1, are not stored.

d. Computation of the pth row of U

$$u_{pp} = \bar{a}_{pp}$$

$$u_{pj} = a_{pj} - \sum_{k=1}^{p-1} l_{pk} u_{kj} \qquad j=p+1,\ldots,n$$

These terms replace the corresponding terms of A.

Note: Double-precision arithmetic is used to compute the inner products needed in steps (1) and (4).

For reference see:

(1) H. J. Bowdler, R. S. Martin, G. Peters, and J. H. Wilkinson, "Solution of Real and Complex Systems of Linear Equations", Numerische Mathematik, vol. 8, no. 3, 1966.
(2) J. H. Wilkinson, The Algebraic Eigenvalue Problem. Clarendon Press, Oxford, 1965.

```
C                                                                       FCTR  10
C     ..................................................................FCTR  20
C                                                                       FCTR  30
C        SUBROUTINE FACTR                                               FCTR  40
C                                                                       FCTR  50
C        PURPOSE                                                        FCTR  60
C           FACTORIZATION OF THE MATRIX A INTO A PRODUCT OF A LOWER     FCTR  70
C           TRIANGULAR MATRIX L AND AN UPPER TRIANGULAR MATRIX U.  L HASFCTR  80
C           UNIT DIAGONAL WHICH IS NOT STORED.                          FCTR  90
C                                                                       FCTR 100
C                                                                       FCTR 110
C        USAGE                                                          FCTR 120
C           CALL FACTR(A,PER,N,IA,IER)                                  FCTR 130
C                                                                       FCTR 140
C        DESCRIPTION OF PARAMETERS                                      FCTR 150
C           A       MATRIX A                                            FCTR 160
C           PER     ONE DIMENSIONAL ARRAY WHERE PERMUTATIONS OF ROWS OF FCTR 170
C                   THE MATRIX ARE STORED                               FCTR 180
C                   DIMENSION OF PER MUST BE GREATER THAN OR EQUAL TO N FCTR 190
C           N       ORDER OF THE MATRIX A                               FCTR 200
C           IA      SIZE OF THE FIRST DIMENSION ASSIGNED TO THE ARRAY A FCTR 210
C                   IN THE CALLING PROGRAM WHEN THE MATRIX IS IN DOUBLE FCTR 220
C                   SUBSCRIPTED DATA STORAGE MODE.  IA=N WHEN THE MATRIX FCTR 230
C                   IS IN SSP VECTOR STORAGE MODE.                      FCTR 240
C           IER     ERROR INDICATOR WHICH IS ZERO IF THERE IS NO ERROR, FCTR 250
C                   AND IS THREE IF THE PROCEDURE FAILS.                FCTR 260
C                                                                       FCTR 270
C        REMARKS                                                        FCTR 280
C           THE ORIGINAL MATRIX, A,IS REPLACED BY THE TRIANGULAR FACTORSFCTR 290
C                                                                       FCTR 300
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  FCTR 310
C           NONE                                                        FCTR 320
C                                                                       FCTR 330
C        METHOD                                                         FCTR 340
C           SUCCESSIVE COMPUTATION OF THE COLUMNS OF L AND THE          FCTR 350
C           CORRESPONDING ROWS OF U.                                    FCTR 360
C                                                                       FCTR 370
C        REFERENCES                                                     FCTR 380
C           J. H. WILKINSON - THE ALGEBRAIC EIGENVALUE PROBLEM -        FCTR 390
C           CLARENDON PRESS, OXFORD, 1965. H. J. BOWDLER, R. S. MARTIN, FCTR 400
C           G. PETERS, AND J. H. WILKINSON - 'SOLUTION OF REAL AND      FCTR 410
C           COMPLEX SYSTEMS OF LINEAR EQUATIONS', NUMERISCHE MATHEMATIK,FCTR 420
C           VOL. 8, NO. 3, 1966, P. 217-234.                            FCTR 430
C                                                                       FCTR 440
C     ..................................................................FCTR 450
C                                                                       FCTR 460
      SUBROUTINE FACTR(A,PER,N,IA,IER)                                  FCTR 470
      DIMENSION A(1),PER(1)                                             FCTR 480
      DOUBLE PRECISION DP                                               FCTR 490
C                                                                       FCTR 500
C        COMPUTATION OF WEIGHTS FOR EQUILIBRATION                       FCTR 510
C                                                                       FCTR 520
      DO 20 I=1,N                                                       FCTR 530
      X=0.                                                              FCTR 540
      IJ=I                                                              FCTR 550
      DO 10 J=1,N                                                       FCTR 560
      IF (ABS(A(IJ))-X)10,10,5                                          FCTR 570
    5 X=ABS(A(IJ))                                                      FCTR 580
   10 IJ=IJ+IA                                                          FCTR 590
      IF (X) 110,110,20                                                 FCTR 600
   20 PER(I)=1./X                                                       FCTR 610
      IC=0                                                              FCTR 620
      DO 100 I=1,N                                                      FCTR 630
      IM1=I-1                                                           FCTR 640
      IP1=I+1                                                           FCTR 650
      IPIVOT=I                                                          FCTR 660
      X=0.                                                              FCTR 670
C                                                                       FCTR 680
C        COMPUTATION OF THE ITH COLUMN OF L                             FCTR 690
C                                                                       FCTR 700
      DO 50 K=1,N                                                       FCTR 710
      KI=IC+K                                                           FCTR 720
      DP=A(KI)                                                          FCTR 730
      IF (I-1) 110,40,25                                                FCTR 740
   25 KJ=K                                                              FCTR 750
      DO 30 J=1,IM1                                                     FCTR 760
      IJ=IC+J                                                           FCTR 770
      DP=DP-1.D0*A(KJ)*A(IJ)                                            FCTR 780
   30 KJ=KJ+IA                                                          FCTR 790
      A(KI)=DP                                                          FCTR 800
C                                                                       FCTR 810
C        SEARCH FOR EQUILIBRATED PIVOT                                  FCTR 820
C                                                                       FCTR 830
   40 IF (X-DABS(DP)*PER(K))45,50,50                                    FCTR 840
   45 IPIVOT=K                                                          FCTR 850
      X=DABS(DP)*PER(K)                                                 FCTR 860
   50 CONTINUE                                                          FCTR 870
      IF (X)110,110,55                                                  FCTR 880
C                                                                       FCTR 890
C        PERMUTATION OF ROWS IF REQUIRED                                FCTR 900
C                                                                       FCTR 910
   55 IF (IPIVOT-I) 110,70,57                                           FCTR 920
   57 KI=IPIVOT                                                         FCTR 930
      IJ=I                                                              FCTR 940
      DO 60 J=1,N                                                       FCTR 950
      X=A(IJ)                                                           FCTR 960
      A(IJ)=A(KI)                                                       FCTR 970
      A(KI)=X                                                           FCTR 980
      KI=KI+IA                                                          FCTR 990
   60 IJ=IJ+IA                                                          FCTR1000
      PER(IPIVOT)=PER(I)                                                FCTR1010
   70 PER(I)=IPIVOT                                                     FCTR1020
      IF (I-N) 72,100,100                                               FCTR1030
   72 IJ=IC+I                                                           FCTR1040
      X=A(IJ)                                                           FCTR1050
C                                                                       FCTR1060
C        COMPUTATION OF THE ITH ROW OF U                                FCTR1070
C                                                                       FCTR1080
      KC=IC+IA                                                          FCTR1090
      DO 90 K=IP1,N                                                     FCTR1100
      KI=IC+K                                                           FCTR1110
      A(KI)=A(KI)/X                                                     FCTR1120
      IF (I-1)110,90,75                                                 FCTR1130
   75 IJ=I                                                              FCTR1140
      KI=KC+I                                                           FCTR1150
      DP=A(KI)                                                          FCTR1160
      DO 80 J=1,IM1                                                     FCTR1170
      KJ=KC+J                                                           FCTR1180
      DP=DP-1.D0*A(IJ)*A(KJ)                                            FCTR1190
   80 IJ=IJ+IA                                                          FCTR1200
      A(KI)=DP                                                          FCTR1210
   90 KC=KC+IA                                                          FCTR1220
  100 IC=IC+IA                                                          FCTR1230
      IER=0                                                             FCTR1240
      RETURN                                                            FCTR1250
  110 IER=3                                                             FCTR1260
      RETURN                                                            FCTR1270
      END
```

Subroutines MFGR and DMFGR

The following calculations are performed for a general rectangular matrix:

1. Determine rank and linearly independent rows and columns of a given matrix.
2. Express a submatrix of maximal rank as product of triangular factors.
3. Express nonbasic rows in terms of basic ones.
4. Express basic variables in terms of free ones.

1. Theoretical background

Calculation (1) is most critical. It is not claimed that MFGR will give the correct rank in all cases, due to the intrinsic difficulty caused by performing calculations with a finite number of digits.

The rank is determined using the standard Gaussian elimination technique with complete pivoting. This implies that the rows and columns of the given m by n matrix A are interchanged at each elimination step if necessary. The interchange information is recorded in two integer permutation vectors IROW and ICOL:

The ith $\left\{\begin{matrix}\text{row}\\ \text{column}\end{matrix}\right\}$ of the interchanged matrix corresponds to the $\left\{\begin{matrix}\text{IROW(I)}^{th}\ \text{row}\\ \text{ICOL(I)}^{th}\ \text{column}\end{matrix}\right\}$ in the original matrix A.

Initially IROW(J) = J and ICOL(J) = J.

The notation A^i is used for the interchanged matrix implied at the ith elimination step.†

a. First elimination step

Let a_{jk} be the absolutely greatest element of matrix A, which is found first in a columnwise scan. The internal tolerance TOL is set equal to $|EPS \cdot a_{jk}|$.

If $|a_{jk}| \leq$ TOL, further calculation is bypassed. Otherwise rows 1 and j and columns 1 and k of matrix A are interchanged, giving A^1. The same interchanges must be applied to IROW, interpreted as column vector, and to ICOL, interpreted as row vector.

A^1 is uniquely expressed as product $L^1 \cdot D^1 \cdot U^1$ by imposing the following conditions:

(I) U^1 is the n by n identity matrix except for the first row.

† Superscripts do not mean powers. They merely indicate the number of the elimination step at which the result is obtained.

(II) L^1 is the m by m identity matrix except for the first column. The first diagonal element has a value of one.

(III) D^1 is an m by n matrix with first diagonal element equal to one, while all remaining elements of the first row and column are equal to zero.

Partitioning of matrices A^1, L^1, D^1, U^1 leads to:

$$\begin{pmatrix} a^1_{11} & A^1_{12} \\ A^1_{21} & A^1_{22} \end{pmatrix} = \begin{pmatrix} 1 & \emptyset \\ L^1_{21} & I \end{pmatrix} \begin{pmatrix} 1 & \emptyset \\ \emptyset & D^1_{22} \end{pmatrix} \begin{pmatrix} u^1_{11} & U^1_{12} \\ \emptyset & I \end{pmatrix}^{\dagger}$$

or explicitly:

$$a^1_{11} = u^1_{11}$$

$$A^1_{12} = U^1_{12}$$

$$A^1_{21} = L^1_{21} \cdot u^1_{11}$$

$$A^1_{22} = L^1_{21} U^1_{12} + D^1_{22}$$

This means:

(I) The elements of the first row of U^1 are

$$u^1_{11} = a^1_{11}, \quad u^1_{12} = a^1_{12}, \ldots, \quad u^1_{1n} = a^1_{1n}$$

(II) The elements of the first column of L^1 are

$$l^1_{11} = 1, \quad l^1_{21} = a^1_{21}/a_{11}, \ldots, \quad l^1_{m1} = a^1_{m1}/a^1_{11}$$

(III) The elements of submatrix D^1_{22} of D^1 are

$$d^1_{ik} = a^1_{ik} - l^1_{i1} \cdot u^1_{1k} = a^1_{ik} - \frac{a^1_{i1} \quad a^1_{1k}}{a^1_{11}};$$

$$i = 2, \ldots, m$$
$$k = 2, \ldots, n$$

† Capital letters are used as notation for matrices or vectors throughout. $\emptyset$ means zero matrix, I means identity matrix; dimensions are implied by compatibility.

Note that it is possible to record all nontrivial entries of L^1, D^1, U^1 in the storage locations occupied by the original A, storing only:

$$\begin{pmatrix} u^1_{11} & U^1_{12} \\ L^1_{21} & D^1_{22} \end{pmatrix}$$

b. Second elimination step

Let d^1_{jk}, $j \geqq 2$, $k \geqq 2$ be the absolutely largest element of D^1_{22}. If $|d^1_{jk}| \leqq$ TOL, D^1_{22} is interpreted as being the zero matrix.

If D^1_{22} is not zero in the above sense, it may be decomposed analogously. In the compact scheme,

$$\begin{pmatrix} u^1_{11} & U^1_{12} \\ L^1_{21} & D^1_{22} \end{pmatrix}$$

rows 2 and j and columns 2 and k are interchanged, obtaining:

$$\begin{pmatrix} u^1_{11} & U^2_{12} \\ L^2_{21} & D^2_{22} \end{pmatrix}$$

The same interchanges must be performed with IROW (interpreted as column vector), with ICOL (interpreted as row vector), and with A^1 giving result A^2:

$$A^2 = \begin{pmatrix} 1 & \emptyset \\ L^2_{21} & I \end{pmatrix} \begin{pmatrix} 1 & \emptyset \\ \emptyset & D^2_{22} \end{pmatrix} \begin{pmatrix} u^1_{11} & U^2_{12} \\ \emptyset & I \end{pmatrix}$$

Now D^2_{22} may be expressed uniquely as the product of the form LDU, imposing conditions I, II, III. The result is:

$$D^2_{22} = \begin{pmatrix} 1 & \emptyset \\ L^2_{32} & I \end{pmatrix} \begin{pmatrix} 1 & \emptyset \\ \emptyset & D^2_{33} \end{pmatrix} \begin{pmatrix} u^2_{22} & U^2_{23} \\ \emptyset & I \end{pmatrix}$$

It is easily seen that $A^2 = L^2 D^2 U^2$,

with:
$$L^2 = \begin{pmatrix} 1 & \emptyset & \emptyset \\ l^2_{21} & 1 & \emptyset \\ L^2_{31} & L^2_{32} & I \end{pmatrix}$$

$$D^2 = \begin{pmatrix} 1 & 0 & \emptyset \\ 0 & 1 & \emptyset \\ \emptyset & \emptyset & D^2_{33} \end{pmatrix}$$

$$U^2 = \begin{pmatrix} u^1_{11} & u^2_{12} & U^2_{13} \\ 0 & u^2_{22} & U^2_{23} \\ \emptyset & \emptyset & I \end{pmatrix}$$

where $$L^2_{21} = \begin{pmatrix} l^2_{21} \\ L^2_{31} \end{pmatrix}, \quad U^2_{12} = (u^2_{12}, U^2_{13})$$

c. Final result of elimination process

At the next elimination step D^2_{33} is factorized, and so on. Now assume that finally $D^r_{r+1,\ r+1}$ equals zero in the sense that all its elements are absolutely less than TOL. This means that A has the rank r, and the end result is the factorization $A^r = L^r D^r U^r$:

$$A^r = \begin{pmatrix} 1 & 0 & \cdots 0 & \emptyset \\ l^2_{21} & 1 & \cdot\ 0 & \emptyset \\ \vdots & \vdots & \ddots & \vdots \\ l^r_{r1} & l^r_{r2} & \cdots 1 & \emptyset \\ L^r_{r+1,\ 1} & L_{r+1,\ 2} & \cdots L^r_{r+1,\ r} & I \end{pmatrix} \cdot$$

$$\begin{pmatrix} 1 & 0\cdots 0 & \emptyset \\ 0 & 1\cdot\ 0 & \emptyset \\ \vdots & \ddots & \vdots \\ 0 & 0\cdots 1 & \emptyset \\ \emptyset & \emptyset\cdots\emptyset & D^r_{r+1,\ r+1} \end{pmatrix} \cdot$$

$$\begin{pmatrix} u^1_{11} & u^2_{12} & \cdots & u^r_{1r} & U^r_{1,\ r+1} \\ 0 & u^2_{22} & \cdots & u^r_{2r} & U^r_{2,\ r+1} \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & u^r_{rr} & U^r_{r,\ r+1} \\ \emptyset & \emptyset & \cdots & \emptyset & I \end{pmatrix}$$

Neglecting the small elements in $D^r_{r+1,\ r+1}$ this may be written more compactly as:

$$A^r = \begin{pmatrix} L \\ LR \end{pmatrix} \begin{pmatrix} U, UR \end{pmatrix}$$

with: $$L = \begin{pmatrix} 1 & 0 \cdots 0 \\ l^2_{21} & 1 \quad 0 \\ \vdots & \ddots \\ l^r_{r1} & l^r_{r2} \cdots 1 \end{pmatrix}$$

$$LR = (L^r_{r+1,1}, L^r_{r+1,2}, \ldots, L^r_{r+1,r})$$

$$U = \begin{pmatrix} u^1_{11} & u^2_{12} \cdots u^r_{1r} \\ 0 & u^2_{22} \cdot \quad \vdots \\ \vdots & \ddots \\ 0 & 0 \cdots u^r_{rr} \end{pmatrix}$$

$$UR = \begin{pmatrix} U^r_{1,\ r+1} \\ U^r_{2,\ r+1} \\ \vdots \\ U^r_{r,\ r+1} \end{pmatrix}$$

L is of dimension r by r and unit lower triangular.
U is of dimension r by r and upper triangular.
LR is of dimension m - r by r; if the given matrix A is row regular (that is, m = r), LR is absent in the final factorization.
UR is of dimension r by n-r; if the given matrix A is column regular (that is, n = r), UR is absent in the final factorization.

d. Further calculations performed

The problem of matrix factorization arises in connection with the solution of systems of equations AX=R. Three different cases must be distinguished:

(1) r = m = n
A is nonsingular, and AX=R has a uniquely determined solution.

(2) r < m
A is not row regular; solutions of AX = R exist only if the linear combinations among the rows of A are also valid among the rows of R.

(3) $r < n$

A is not column regular; $AX = \emptyset$ has nontrivial solutions.

The cases (2) and (3) may occur combined.

The solution, if it exists, is uniquely determined if $r = n$; otherwise it contains $n - r$ free parameters. It is quite natural to ask for the linear combinations among the rows of given matrix A and for the linear forms expressing basic variables in terms of free ones. Therefore, instead of LR and UR, matrices C and H are returned, containing linear combinations and homogeneous solutions respectively.

Observe carefully that the calculated factorization belongs to the interchanged matrix A^r. Therefore $A^r X^r = R^r$ is dealt with instead of $AX = R$, where $\begin{Bmatrix} X^r \\ R^r \end{Bmatrix}$ is obtained from $\begin{Bmatrix} X \\ R \end{Bmatrix}$ using the $\begin{Bmatrix} ICOL(I)^{th} \\ IROW(J)^{th} \end{Bmatrix}$ element of $\begin{Bmatrix} X \\ R \end{Bmatrix}$ as $\begin{Bmatrix} I^{th} \\ J^{th} \end{Bmatrix}$ element of $\begin{Bmatrix} X^r \\ R^r \end{Bmatrix}$; $\begin{Bmatrix} I = 1, \ldots, n \\ J = 1, \ldots, m \end{Bmatrix}$.

Let X^r, R^r be partitioned into $\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$ and $\begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$.

Then:

$$\begin{pmatrix} L \\ LR \end{pmatrix} (U,\ UR) \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$$

or explicitly:

$$L \cdot U \cdot X_1 + L \cdot UR \cdot X_2 = R_1$$
$$LR \cdot U \cdot X_1 + LR \cdot UR \cdot X_2 = R_2$$

Since L and U are nonsingular, this implies:

$$X_1 = U^{-1} \cdot L^{-1} \cdot R_1 - U^{-1} \cdot UR \cdot X_2$$
$$LR \cdot L^{-1} R_1 = R_2$$

For the convenience of the user, LR is replaced by $LR \cdot L^{-1} = C$; UR is replaced by $-U^{-1}\ UR = H$, while L and U remain untouched. Consistency requires that $R_2 = CR_1$, and homogeneous solutions are given by $X_1 = HX_2$. In case of a consistent system of equations $A^r X^r = R^r$, the general solution is $X^r = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$ with $X_1 = U^{-1} L^{-1} R_1 + HX_2$, while the values of the free variables contained in X_2 may be chosen arbitrarily.

2. Format of Results

Subroutine MFGR returns matrices L, U, C, H, D in the storage area originally occupied by the input matrix A in the compact scheme shown in Figure 9.

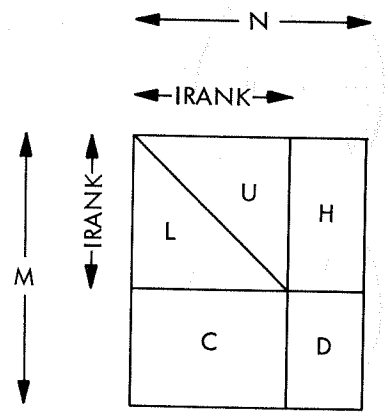


Figure 9. Format scheme (MFGR)

3. Numerical examples

a. Let $A = \begin{pmatrix} 5 & 5 \\ 1 & 1 \end{pmatrix}$, EPS = 1.0E-7

Subroutine MFGR returns as results:

L = (1), U = (5), C = (0.2), H = (-1.0)

combined in the compact scheme:

$$\begin{pmatrix} 5.0 & -1.0 \\ 0.2 & 0.6E\text{-}7 \end{pmatrix} = \left(\begin{array}{c|c} U & H \\ \hline C & D \end{array}\right)$$

IRANK = 1
ICOL = (1, 2)
IROW = (1, 2)

The value 0.6E-7 is obtained instead of the correct value 0, due to roundoff errors.

This example shows that a proper choice of EPS is somewhat greater than 10^{-d}, if d is the number of significant digits in the representation of floating point numbers.

With EPS = 0, MFGR would give the incorrect value of 2 for the rank. The choice EPS = 0 may well result in an incorrect value less than the correct rank, as shown by the following.

If the element a_{22} is slightly decreased, resulting in a matrix A of correct rank equal to 2, then subroutine MFGR gives an incorrect value of 1 for the rank with any choice of the relative tolerance EPS, even with EPS = 0.

b. Let $A = \begin{pmatrix} 1 & 3 & 5 \\ 1 & 3 & 5 \\ 1 & 3 & 1 \\ 1 & 1 & 3 \\ 1 & 3 & 5 \end{pmatrix}$, EPS = 1.0E-7

Subroutine MFGR returns as results:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0.2 & 1 & 0 \\ 0.6 & -0.3333330 & 1 \end{pmatrix}$$

$$U = \begin{pmatrix} 5 & 3 & 1 \\ 0 & 2.4 & 0.8 \\ 0 & 0 & 0.6666664 \end{pmatrix}$$

$$C = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

combined in the compact scheme:

$$\begin{pmatrix} 5 & 3 & 1 \\ 0.2 & 2.4 & 0.8 \\ 0.6 & -0.3333330 & 0.6666664 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} U \\ L \\ C \end{pmatrix}$$

IRANK = 3
ICOL = (3, 2, 1)
IROW = (1, 3, 4, 2, 5)

Linear dependencies among rows:

row (2) = 1 · row (1) + 0 · row (3) + 0 · row (4)
row (5) = 1 · row (1) + 0 · row (3) + 0 · row (4)

$$L \cdot U = \begin{pmatrix} a_{13} & a_{12} & a_{11} \\ a_{33} & a_{32} & a_{31} \\ a_{43} & a_{42} & a_{41} \end{pmatrix} = \begin{pmatrix} 5 & 3 & 1 \\ 1 & 3 & 1 \\ 3 & 1 & 1 \end{pmatrix}$$

c. Let $A = \begin{pmatrix} 1 & 1 & 3 & 3 & 1 \\ 1 & 1 & 3 & 5 & 3 \\ 1 & 3 & 1 & 5 & 5 \end{pmatrix}$, EPS = 1.0E-7

Subroutine MFGR returns as results:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0.6 & 0.2 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 5 & 1 & 3 \\ 0 & 2 & -2 \\ 0 & 0 & 1.6 \end{pmatrix}$$

$$H = \begin{pmatrix} -0.48E-7 & -0.9999992 \\ -0.2499999 & -0.2500009 \\ -0.2499999 & 0.7499991 \end{pmatrix}$$

combined in the compact scheme:

$$\begin{pmatrix} 5 & 1 & 3 & -0.48E-7 & -0.9999992 \\ 1 & 2 & -2 & -0.2499999 & -0.2500009 \\ 0.6 & 0.2 & 1.6 & -0.2499999 & 0.7499991 \end{pmatrix} =$$

$$\begin{pmatrix} U \\ L \quad H \end{pmatrix}$$

IRANK = 3
ICOL = (4, 2, 3, 1, 5)
IROW = (2, 3, 1)

Homogeneous solution of A · X = 0

$$x_4 = -0.48E-7 \cdot x_1 - 0.9999992 \cdot x_5$$
$$x_2 = -0.2499999 \cdot x_1 - 0.2500009 \cdot x_5$$
$$x_3 = -0.2499999 \cdot x_1 + 0.7499991 \cdot x_5$$

x_1 and x_5 may be chosen arbitrarily.

$$L \cdot U = \begin{pmatrix} a_{24} & a_{22} & a_{23} \\ a_{34} & a_{32} & a_{33} \\ a_{14} & a_{12} & a_{13} \end{pmatrix} = \begin{pmatrix} 5 & 1 & 3 \\ 5 & 3 & 1 \\ 3 & 1 & 3 \end{pmatrix}$$

```
C                                                                       MFGR  10
C     ..................................................................MFGR  20
C                                                                       MFGR  30
C        SUBROUTINE MFGR                                                MFGR  40
C                                                                       MFGR  50
C        PURPOSE                                                        MFGR  60
C           FOR A GIVEN M BY N MATRIX THE FOLLOWING CALCULATIONS        MFGR  70
C           ARE PERFORMED                                               MFGR  80
C           (1) DETERMINE RANK AND LINEARLY INDEPENDENT ROWS AND        MFGR  90
C               COLUMNS (BASIS).                                        MFGR 100
C           (2) FACTORIZE A SUBMATRIX OF MAXIMAL RANK.                  MFGR 110
C           (3) EXPRESS NON-BASIC ROWS IN TERMS OF BASIC ONES.          MFGR 120
C           (4) EXPRESS BASIC VARIABLES IN TERMS OF FREE ONES.          MFGR 130
C                                                                       MFGR 140
C        USAGE                                                          MFGR 150
C           CALL MFGR(A,M,N,EPS,IRANK,IROW,ICOL)                        MFGR 160
C                                                                       MFGR 170
C        DESCRIPTION OF PARAMETERS                                      MFGR 180
C           A       - GIVEN MATRIX WITH M ROWS AND N COLUMNS.           MFGR 190
C                     ON RETURN A CONTAINS THE FIVE SUBMATRICES         MFGR 200
C                     L, U, H, D, C.                                    MFGR 210
C           M       - NUMBER OF ROWS OF MATRIX A.                       MFGR 220
C           N       - NUMBER OF COLUMNS OF MATRIX A.                    MFGR 230
C           EPS     - TESTVALUE FOR ZERO AFFECTED BY ROUNDOFF NOISE.    MFGR 240
C           IRANK   - RESULTANT RANK OF GIVEN MATRIX.                   MFGR 250
C           IROW    - INTEGER VECTOR OF DIMENSION M CONTAINING THE      MFGR 260
C                     SUBSCRIPTS OF BASIC ROWS IN IROW(1),...,IROW(IRANK)MFGR 270
C           ICOL    - INTEGER VECTOR OF DIMENSION N CONTAINING THE      MFGR 280
C                     SUBSCRIPTS OF BASIC COLUMNS IN ICOL(1) UP TO      MFGR 290
C                     ICOL(IRANK).                                      MFGR 300
C                                                                       MFGR 310
C        REMARKS                                                        MFGR 320
C           THE LEFT HAND TRIANGULAR FACTOR IS NORMALIZED SUCH THAT     MFGR 330
C           THE DIAGONAL CONTAINS ALL ONES THUS ALLOWING TO STORE ONLY  MFGR 340
C           THE SUBDIAGONAL PART.                                       MFGR 350
C                                                                       MFGR 360
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MFGR 370
C           NONE                                                        MFGR 380
C                                                                       MFGR 390
C        METHOD                                                         MFGR 400
C           GAUSSIAN ELIMINATION TECHNIQUE IS USED FOR CALCULATION      MFGR 410
C           OF THE TRIANGULAR FACTORS OF A GIVEN MATRIX.                MFGR 420
C           COMPLETE PIVOTING IS BUILT IN.                              MFGR 430
C           IN CASE OF A SINGULAR MATRIX ONLY THE TRIANGULAR FACTORS    MFGR 440
C           OF A SUBMATRIX OF MAXIMAL RANK ARE RETAINED.                MFGR 450
C           THE REMAINING PARTS OF THE RESULTANT MATRIX GIVE THE        MFGR 460
C           DEPENDENCIES OF ROWS AND THE SOLUTION OF THE HOMOGENEOUS    MFGR 470
C           MATRIX EQUATION A*X=0.                                      MFGR 480
C                                                                       MFGR 490
C     ..................................................................MFGR 500
C                                                                       MFGR 510
      SUBROUTINE MFGR(A,M,N,EPS,IRANK,IROW,ICOL)                        MFGR 520
C                                                                       MFGR 530
C        DIMENSIONED DUMMY VARIABLES                                    MFGR 540
      DIMENSION A(1),IROW(1),ICOL(1)                                    MFGR 550
C                                                                       MFGR 560
C        TEST OF SPECIFIED DIMENSIONS                                   MFGR 570
      IF(M)2,2,1                                                        MFGR 580
    1 IF(N)2,2,4                                                        MFGR 590
    2 IRANK=-1                                                          MFGR 600
    3 RETURN                                                            MFGR 610
C        RETURN IN CASE OF FORMAL ERRORS                                MFGR 620
```

```
C
C
C        INITIALIZE COLUMN INDEX VECTOR
C        SEARCH FIRST PIVOT ELEMENT
    4 IRANK=0
      PIV=0.
      JJ=0
      DO 6 J=1,N
      ICOL(J)=J
      DO 5 I=1,M
      JJ=JJ+1
      HOLD=A(JJ)
      IF(ABS(PIV)-ABS(HOLD))5,6,6
    5 PIV=HOLD
      IR=I
      IC=J
    6 CONTINUE
C
C        INITIALIZE ROW INDEX VECTOR
      DO 7 I=1,M
    7 IROW(I)=I
C
C        SET UP INTERNAL TOLERANCE
      TOL=ABS(EPS*PIV)
C
C        INITIALIZE ELIMINATION LOOP
      NM=N*M
      DO 19 NCOL=M,NM,M
C
C        TEST FOR FEASIBILITY OF PIVOT ELEMENT
    8 IF(ABS(PIV)-TOL)20,20,9
C
C        UPDATE RANK
    9 IRANK=IRANK+1
C
C        INTERCHANGE ROWS IF NECESSARY
      JJ=IR-IRANK
      IF(JJ)12,12,10
   10 DO 11 J=IRANK,NM,M
      I=J+JJ
      SAVE=A(J)
      A(J)=A(I)
   11 A(I)=SAVE
C
C        UPDATE ROW INDEX VECTOR
      JJ=IROW(IR)
      IROW(IR)=IROW(IRANK)
      IROW(IRANK)=JJ
C
C        INTERCHANGE COLUMNS IF NECESSARY
   12 JJ=(IC-IRANK)*M
      IF(JJ)15,15,13
   13 KK=NCOL
      DO 14 J=1,M
      I=KK+JJ
      SAVE=A(KK)
      A(KK)=A(I)
      KK=KK-1
   14 A(I)=SAVE
C
C        UPDATE COLUMN INDEX VECTOR
      JJ=ICOL(IC)
      ICOL(IC)=ICOL(IRANK)
      ICOL(IRANK)=JJ
   15 KK=IRANK+1
      MM=IRANK-M
      LL=NCOL+MM
C
C        TEST FOR LAST ROW
      IF(MM)16,25,25
C
C        TRANSFORM CURRENT SUBMATRIX AND SEARCH NEXT PIVOT
   16 JJ=LL
      SAVE=PIV
      PIV=0.
      DO 19 J=KK,M
      JJ=JJ+1
      HOLD=A(JJ)/SAVE
      A(JJ)=HOLD
      L=J-IRANK
C
C        TEST FOR LAST COLUMN
      IF(IRANK-N)17,19,19
   17 II=JJ
      DO 19 I=KK,N
      II=II+M
      MM=II-L
      A(II)=A(II)-HOLD*A(MM)
      IF(ABS(A(II))-ABS(PIV))19,19,18
   18 PIV=A(II)
      IR=J
      IC=I
   19 CONTINUE
C
C        SET UP MATRIX EXPRESSING ROW DEPENDENCIES
   20 IF(IRANK-1)3,25,21
   21 IR=LL
      DO 24 J=2,IRANK
      II=J-1
      IR=IR-M
      JJ=LL
      DO 23 I=KK,M
      HOLD=0.
      JJ=JJ+1
      MM=JJ
      IC=IR
      DO 22 L=1,II
      HOLD=HOLD+A(MM)*A(IC)
      IC=IC-1
   22 MM=MM-M
   23 A(MM)=A(MM)-HOLD
   24 CONTINUE
C
C        TEST FOR COLUMN REGULARITY
   25 IF(N-IRANK)3,3,26
C
C        SET UP MATRIX EXPRESSING BASIC VARIABLES IN TERMS OF FREE
C        PARAMETERS (HOMOGENEOUS SOLUTION).
   26 IR=LL
      KK=LL+M
      DO 30 J=1,IRANK
      DO 29 I=KK,NM,M
      JJ=IR
      LL=I
      HOLD=0.
      II=J
   27 II=II-1
      IF(II)29,29,28
   28 HOLD=HOLD-A(JJ)*A(LL)
      JJ=JJ-M
      LL=LL-1
      GOTO 27
   29 A(LL)=(HOLD-A(LL))/A(JJ)
   30 IR=IR-1
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DMFGR
C
C        PURPOSE
C           FOR A GIVEN M BY N MATRIX THE FOLLOWING CALCULATIONS
C           ARE PERFORMED
C           (1) DETERMINE RANK AND LINEARLY INDEPENDENT ROWS AND
C               COLUMNS (BASIS).
C           (2) FACTORIZE A SUBMATRIX OF MAXIMAL RANK.
C           (3) EXPRESS NON-BASIC ROWS IN TERMS OF BASIC ONES.
C           (4) EXPRESS BASIC VARIABLES IN TERMS OF FREE ONES.
C
C        USAGE
C           CALL DMFGR(A,M,N,EPS,IRANK,IROW,ICOL)
C
C        DESCRIPTION OF PARAMETERS
C           A      - DOUBLE PRECISION GIVEN MATRIX WITH M ROWS
C                    AND N COLUMNS.
C                    ON RETURN A CONTAINS THE TRIANGULAR FACTORS
C                    OF A SUBMATRIX OF MAXIMAL RANK.
C           M      - NUMBER OF ROWS OF MATRIX A.
C           N      - NUMBER OF COLUMNS OF MATRIX A.
C           EPS    - SINGLE PRECISION TESTVALUE FOR ZERO AFFECTED BY
C                    ROUNDOFF NOISE.
C           IRANK  - RESULTANT RANK OF GIVEN MATRIX.
C           IROW   - INTEGER VECTOR OF DIMENSION M CONTAINING THE
C                    SUBSCRIPTS OF BASIC ROWS IN IROW(1),...,IROW(IRANK)
C           ICOL   - INTEGER VECTOR OF DIMENSION N CONTAINING THE
C                    SUBSCRIPTS OF BASIC COLUMNS IN ICOL(1) UP TO
C                    ICOL(IRANK).
C
C        REMARKS
C           THE LEFT HAND TRIANGULAR FACTOR IS NORMALIZED SUCH THAT
C           THE DIAGONAL CONTAINS ALL ONES THUS ALLOWING TO STORE ONLY
C           THE SUBDIAGONAL PART.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           GAUSSIAN ELIMINATION TECHNIQUE IS USED FOR CALCULATION
C           OF THE TRIANGULAR FACTORS OF A GIVEN MATRIX.
C           COMPLETE PIVOTING IS BUILT IN.
C           IN CASE OF A SINGULAR MATRIX ONLY THE TRIANGULAR FACTORS
C           OF A SUBMATRIX OF MAXIMAL RANK ARE RETAINED.
C           THE REMAINING PARTS OF THE RESULTANT MATRIX GIVE THE
C           DEPENDENCIES OF ROWS AND THE SOLUTION OF THE HOMOGENEOUS
C           MATRIX EQUATION A*X=0.
C
C     ..................................................................
C
      SUBROUTINE DMFGR(A,M,N,EPS,IRANK,IROW,ICOL)
C
C        DIMENSIONED DUMMY VARIABLES
      DIMENSION A(1),IROW(1),ICOL(1)
      DOUBLE PRECISION A,PIV,HOLD,SAVE
C
C        TEST OF SPECIFIED DIMENSIONS
      IF(M)2,2,1
    1 IF(N)2,2,4
    2 IRANK=-1
    3 RETURN
C        RETURN IN CASE OF FORMAL ERRORS
C
C
C        INITIALIZE COLUMN INDEX VECTOR
C        SEARCH FIRST PIVOT ELEMENT
    4 IRANK=0
      PIV=0.0D
      JJ=0
      DO 6 J=1,N
      ICOL(J)=J
      DO 6 I=1,M
      JJ=JJ+1
      HOLD=A(JJ)
      IF(DABS(PIV)-DABS(HOLD))5,6,6
    5 PIV=HOLD
      IR=I
      IC=J
    6 CONTINUE
C        INITIALIZE ROW INDEX VECTOR
      DO 7 I=1,M
    7 IROW(I)=I
C
C        SET UP INTERNAL TOLERANCE
      TOL=ABS(EPS*SNGL(PIV))
C
C        INITIALIZE ELIMINATION LOOP
      NM=N*M
      DO 19 NCOL=M,NM,M
C
C        TEST FOR FEASIBILITY OF PIVOT ELEMENT
    8 IF(ABS(SNGL(PIV))-TOL)20,20,9
C
C        UPDATE RANK
    9 IRANK=IRANK+1
C
C        INTERCHANGE ROWS IF NECESSARY
      JJ=IR-IRANK
      IF(JJ)12,12,10
   10 DO 11 J=IRANK,NM,M
      I=J+JJ
      SAVE=A(J)
      A(J)=A(I)
   11 A(I)=SAVE
C
C        UPDATE ROW INDEX VECTOR
      JJ=IROW(IR)
      IROW(IR)=IROW(IRANK)
      IROW(IRANK)=JJ
C
C        INTERCHANGE COLUMNS IF NECESSARY
   12 JJ=(IC-IRANK)*M
```

```
      IF(JJ)15,15,13
   13 KK=NCOL
      DO 14 J=1,M
      I=KK+JJ
      SAVE=A(KK)
      A(KK)=A(I)
      KK=KK-1
   14 A(I)=SAVE
C
C        UPDATE COLUMN INDEX VECTOR
      JJ=ICOL(IC)
      ICOL(IC)=ICOL(IRANK)
      ICOL(IRANK)=JJ
   15 KK=IRANK+1
      MM=IRANK-M
      LL=NCOL+MM
C
C        TEST FOR LAST ROW
      IF(MM)16,25,25
C
C        TRANSFORM CURRENT SUBMATRIX AND SEARCH NEXT PIVOT
   16 JJ=LL
      SAVE=PIV
      PIV=0.D0
      DO 19 J=KK,M
      JJ=JJ+1
      HOLD=A(JJ)/SAVE
      A(JJ)=HOLD
      L=J-IRANK
C
C        TEST FOR LAST COLUMN
      IF(IRANK-N)17,19,19
   17 II=JJ
      DO 19 I=KK,N
      II=II+M
      MM=II-L
      A(II)=A(II)-HOLD*A(MM)
      IF(DABS(A(II))-DABS(PIV))19,19,18
   18 PIV=A(II)
      IR=J
      IC=I
   19 CONTINUE
C
C        SET UP MATRIX EXPRESSING ROW DEPENDENCIES
   20 IF(IRANK-1)3,25,21
   21 IR=LL
      DO 24 J=2,IRANK
      II=J-1
      IR=IR-M
      JJ=LL
      DO 23 I=KK,M
      HOLD=0.D0
      JJ=JJ+1
      MM=JJ
      IC=IR
      DO 22 L=1,II
      HOLD=HOLD+A(MM)*A(IC)
      IC=IC-1
   22 MM=MM-M
   23 A(MM)=A(MM)-HOLD
   24 CONTINUE
C
C        TEST FOR COLUMN REGULARITY
   25 IF(N-IRANK)3,3,26
C
C        SET UP MATRIX EXPRESSING BASIC VARIABLES IN TERMS OF FREE
C        PARAMETERS (HOMOGENEOUS SOLUTION).
   26 IR=LL
      KK=LL+M
      DO 30 J=1,IRANK
      DO 29 I=KK,NM,M
      JJ=IR
      LL=I
      HOLD=0.D0
      II=J
   27 II=II-1
      IF(II)28,29,28
   28 HOLD=HOLD-A(JJ)*A(LL)
      JJ=JJ-M
      LL=LL-1
      GOTO 27
   29 A(LL)=(HOLD-A(LL))/A(JJ)
   30 IR=IR-1
      RETURN
      END
```

Subroutines GELS and DGELS

These subroutines solve a system of simultaneous linear equations with symmetric coefficient matrix by Gauss elimination. Consider the system of simultaneous linear equations

$$A * X = R \tag{1}$$

with symmetric m by m coefficient matrix, the upper triangular part of which is stored by column in m * (m+1)/2 successive storage locations, and an m by n right-hand side matrix R stored by column in m * n successive storage locations. Solution is done by Gauss elimination with pivoting in the main diagonal of matrix A. If matrix R is the identity matrix, solution X is the inverse of matrix A. Solution matrix X is placed in positions of the right-hand side matrix R and is stored by column also. Thus, the computation of the solution requires no extra m by n array of storage. Only an auxiliary storage array named AUX with (m-1) storage locations is necessary.

Explicitly, the given system (1) is of the form:+

$$\begin{pmatrix} \underline{a_{11}} & \underline{a_{12}} & \underline{a_{13}} & \cdots & \underline{a_{1m}} \\ a_{21} & \underline{a_{22}} & \underline{a_{23}} & \cdots & \underline{a_{2m}} \\ a_{31} & a_{32} & \underline{a_{33}} & \cdots & \underline{a_{3m}} \\ a_{m1} & a_{m2} & a_{m3} & \cdots & \underline{a_{mm}} \end{pmatrix} * \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ x_{31} & x_{32} & \cdots & x_{3n} \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

$$= \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ r_{31} & r_{32} & \cdots & r_{3n} \\ r_{m1} & r_{m2} & \cdots & r_{mn} \end{pmatrix} \tag{2}$$

The first step is to search the main diagonal of matrix A for the element of greatest absolute value, say a_{jj}, and to select it as first pivot ($p = a_{jj}$). The reason for pivoting only in the main diagonal of A is that rest-matrices of $A^{(k)}$

+Note that subroutines GELS and DGELS require only the upper triangular part of matrix A; that is, the elements a_{11}; a_{12}, a_{22}; a_{13}, a_{23}, a_{33};...; a_{1m}, a_{2m},..., a_{mm}. These elements are underlined in formula (2).

$k = 1, 2, \ldots, m-1$) must remain symmetrical during the whole algorithm. With a_{jj}, generate the internal absolute tolerance for testing usefulness of the symmetric algorithm in the following way:

$$\text{tol} = |a_{jj}| \cdot \varepsilon \tag{3}$$

with a given relative tolerance ε.

Suppose that pivot element a_{jj} is equal to a_{11}. If it is not, interchange the first rows of matrices A and R with the jth and the first column of matrix A with the jth, and save column interchange information by storing the difference (j-1) of pivot column index j and step counter k = 1 [interchanging column 1 with column j means interchanging variables x_{11} with x_{1j} $(1 = 1, 2, \ldots, n)$].

Now transform the elements of pivot rows in matrices A and R by division with p, and the other elements by adding $-a_{\nu 1}$ times the new first rows of these two matrices to the other ν rows, thus getting:++

$$a_{1l}^{(1)} = \frac{a_{1l}}{p} \quad (l = 1, 2, \ldots, m) \tag{4}$$

$$r_{1l}^{(1)} = \frac{r_{1l}}{p} \quad (l = 1, 2, \ldots, n) \tag{5}$$

$$a_{\nu l}^{(1)} = a_{\nu l} - a_{\nu 1} \cdot a_{1l}^{(1)} \quad (l = 2, 3, \ldots, m;\ \nu = 2, 3, \ldots, m) \tag{6}$$

$$r_{\nu l}^{(1)} = r_{\nu l} - a_{\nu 1} \cdot r_{1l}^{(1)} \quad (l = 1, 2, \ldots, n;\ \nu = 2, 3, \ldots, m) \tag{7}$$

As column interchange information is saved in the first position of the main diagonal, the result of the first step is the two matrices:

$$A^{(1)} = \begin{pmatrix} \underline{(j-1)} & \underline{a_{12}^{(1)}} & \underline{a_{13}^{(1)}} & \cdots & \underline{a_{1m}^{(1)}} \\ 0 & \underline{a_{22}^{(1)}} & \underline{a_{23}^{(1)}} & \cdots & \underline{a_{2m}^{(1)}} \\ 0 & a_{32}^{(1)} & \underline{a_{33}^{(1)}} & \cdots & \underline{a_{3m}^{(1)}} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & a_{m2}^{(1)} & a_{m3}^{(1)} & \cdots & \underline{a_{mm}^{(1)}} \end{pmatrix} \quad \text{and}$$

$$R^{(1)} = \begin{pmatrix} r_{11}^{(1)} & r_{12}^{(1)} & \cdots & r_{1n}^{(1)} \\ r_{21}^{(1)} & r_{22}^{(1)} & \cdots & r_{2n}^{(1)} \\ r_{31}^{(1)} & r_{32}^{(1)} & \cdots & r_{3n}^{(1)} \\ \cdots & \cdots & \cdots & \cdots \\ r_{m1}^{(1)} & r_{m2}^{(1)} & \cdots & r_{mn}^{(1)} \end{pmatrix}$$

It is easily seen from equations (4) - (7) that the rest of matrix $A^{(1)}$ -- that is, matrix $A^{(1)}$ without the first row and first column -- is symmetrical and that actually only the underlined elements must be calculated and stored. Therefore, the range of index l in formula (6) reduces to $l = \nu, \nu + 1, \ldots, m$.

Now repeat this procedure m-2 times, starting at each step with the matrix $A^{(k)}$ of the step before without the first k rows and first k columns, and the matrix $R^{(k)}$ without the first k rows. The total result after m-1 steps is the matrices:

$$A^{(m-1)} = \begin{pmatrix} (j_1-1) & a_{12}^{(1)} & a_{13}^{(1)} & \cdots & a_{1m}^{(1)} \\ 0 & (j_2-2) & a_{23}^{(2)} & \cdots & a_{2m}^{(2)} \\ 0 & 0 & (j_3-3) & \cdots & a_{3m}^{(3)} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & (j_m-m) \end{pmatrix} \quad \text{and}$$

$$R^{(m-1)} = \begin{pmatrix} r_{11}^{(1)} & r_{12}^{(1)} & \cdots & r_{1n}^{(1)} \\ r_{21}^{(2)} & r_{22}^{(2)} & \cdots & r_{2n}^{(2)} \\ r_{31}^{(3)} & r_{32}^{(3)} & \cdots & r_{3n}^{(3)} \\ \cdots & \cdots & \cdots & \cdots \\ r_{m1}^{(m)} & r_{m2}^{(m)} & \cdots & r_{mn}^{(m)} \end{pmatrix}$$

++ Note that transformation of pivot row in matrix A destroys pivot column, which is, due to symmetry, stored in the same locations. As pivot column is used unchanged for transformation of rest of A and R, it has to be saved in auxiliary array AUX before transforming pivot row.

Now work backward and set:

$$
\left.
\begin{aligned}
r^{(m)}_{m-1,l} &= r^{(m-1)}_{m-1,l} - a^{(m-1)}_{m-1,m} \cdot r^{(m)}_{ml} \quad (l = 1,2,\ldots,n) \\
r^{(m)}_{m-2,l} &= r^{(m-2)}_{m-2,l} - a^{(m-2)}_{m-2,m-1} \cdot r^{(m)}_{m-1,l} \\
&\ldots\ldots\ldots \\
r^{(m)}_{11} &= r^{(1)}_{11} - a^{(1)}_{12} \cdot r^{(m)}_{21} - a^{(1)}_{13} \cdot r^{(m)}_{31} \\
&\quad - a^{(m-2)}_{m-2,m} \cdot r^{(m)}_{ml} \quad (l = 1,2,\ldots,n) \\
&\quad - \ldots - a^{(1)}_{1m} \cdot r^{(m)}_{ml} \quad (l = 1,2,\ldots,n)
\end{aligned}
\right\} \quad (8)
$$

After each step of back substitution, rows of solution matrix $X = R^{(m)}$ have to be back-interchanged according to interchange information in the corresponding main diagonal element of matrix $A^{(m-1)}$, in order to get the correct sequence of right-hand side column elements corresponding to the sequence of left-hand side row elements.

The only case in which the procedure described above can fail to give a solution occurs when at any step all elements in the main diagonal of the rest-matrix of $A^{(k)}$ become zero, and no pivot element can be found. In this case, the procedure is bypassed and the error message ier = -1 is given. This may -- but does not necessarily -- mean, that matrix A is singular. Possibly subroutines GELG or DGELG (which are working with complete pivoting) will be able to find a solution in cases where subroutines GELS or DGELS fail. Actually, because of rounding errors, a further check of the absolute values of pivot elements is performed by the procedure. If at elimination step k this absolute value becomes less than tol (see equation 3), it is likely that there was loss of significance in the computation of the diagonal elements. But as this may not necessarily be the case, and as this test depends highly on the choice of the relative tolerance ε+, the procedure gives only the warning ier = k-1, which indicates that there is a possible loss of significance in the results computed by the algorithm.++But here it is also possible that subroutines GELG or DGELG will give better results. If there is only one equation to solve (m = 1), the test on loss of significance is suppressed.

+For subroutine GELS, a relative tolerance between 10^{-6} and 10^{-7} is suggested; and for subroutine DGELS, between 10^{-14} and 10^{-16}.

++For example, $\varepsilon = 10^{-5}$ and warning ier = 3 mean that there is a possible loss of about five or more significant digits in the initial values of elimination step 4.

```
C                                                                       GELS  10
C     ......................................................................GELS  20
C                                                                       GELS  30
C     SUBROUTINE GELS                                                   GELS  40
C                                                                       GELS  50
C     PURPOSE                                                           GELS  60
C        TO SOLVE A SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS WITH        GELS  70
C        SYMMETRIC COEFFICIENT MATRIX UPPER TRIANGULAR PART OF WHICH    GELS  80
C        IS ASSUMED TO BE STORED COLUMNWISE.                            GELS  90
C                                                                       GELS 100
C     USAGE                                                             GELS 110
C        CALL GELS(R,A,M,N,EPS,IER,AUX)                                 GELS 120
C                                                                       GELS 130
C     DESCRIPTION OF PARAMETERS                                         GELS 140
C        R      - M BY N RIGHT HAND SIDE MATRIX.  (DESTROYED)          GELS 150
C                 ON RETURN R CONTAINS THE SOLUTION OF THE EQUATIONS.GELS 160
C        A      - UPPER TRIANGULAR PART OF THE SYMMETRIC               GELS 170
C                 M BY M COEFFICIENT MATRIX.  (DESTROYED)               GELS 180
C        M      - THE NUMBER OF EQUATIONS IN THE SYSTEM.               GELS 190
C        N      - THE NUMBER OF RIGHT HAND SIDE VECTORS.               GELS 200
C        EPS    - AN INPUT CONSTANT WHICH IS USED AS RELATIVE          GELS 210
C                 TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.           GELS 220
C        IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS           GELS 230
C                 IER=0  - NO ERROR,                                    GELS 240
C                 IER=-1 - NO RESULT BECAUSE OF M LESS THAN 1 OR        GELS 250
C                          PIVOT ELEMENT AT ANY ELIMINATION STEP        GELS 260
C                          EQUAL TO 0.                                  GELS 270
C                 IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-     GELS 280
C                          CANCE INDICATED AT ELIMINATION STEP K+1,     GELS 290
C                          WHERE PIVOT ELEMENT WAS LESS THAN OR         GELS 300
C                          EQUAL TO THE INTERNAL TOLERANCE EPS TIMES    GELS 310
C                          ABSOLUTELY GREATEST MAIN DIAGONAL            GELS 320
C                          ELEMENT OF MATRIX A.                         GELS 330
C        AUX    - AN AUXILIARY STORAGE ARRAY WITH DIMENSION M-1.       GELS 340
C                                                                       GELS 350
C     REMARKS                                                           GELS 360
C        UPPER TRIANGULAR PART OF MATRIX A IS ASSUMED TO BE STORED      GELS 370
C        COLUMNWISE IN M*(M+1)/2 SUCCESSIVE STORAGE LOCATIONS, RIGHT    GELS 380
C        HAND SIDE MATRIX R COLUMNWISE IN N*M SUCCESSIVE STORAGE        GELS 390
C        LOCATIONS. ON RETURN SOLUTION MATRIX R IS STORED COLUMNWISE    GELS 400
C        TOO.                                                           GELS 410
C        THE PROCEDURE GIVES RESULTS IF THE NUMBER OF EQUATIONS M IS    GELS 420
C        GREATER THAN 0 AND PIVOT ELEMENTS AT ALL ELIMINATION STEPS     GELS 430
C        ARE DIFFERENT FROM 0. HOWEVER WARNING IER=K - IF GIVEN -       GELS 440
C        INDICATES POSSIBLE LOSS OF SIGNIFICANCE. IN CASE OF A WELL     GELS 450
C        SCALED MATRIX A AND APPROPRIATE TOLERANCE EPS, IER=K MAY BE    GELS 460
C        INTERPRETED THAT MATRIX A HAS THE RANK K. NO WARNING IS        GELS 470
C        GIVEN IN CASE M=1.                                             GELS 480
C        ERROR PARAMETER IER=-1 DOES NOT NECESSARILY MEAN THAT          GELS 490
C        MATRIX A IS SINGULAR, AS ONLY MAIN DIAGONAL ELEMENTS           GELS 500
C        ARE USED AS PIVOT ELEMENTS. POSSIBLY SUBROUTINE GELG (WHICH    GELS 510
C        WORKS WITH TOTAL PIVOTING) WOULD BE ABLE TO FIND A SOLUTION.GELS 520
C                                                                       GELS 530
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     GELS 540
C        NONE                                                           GELS 550
C                                                                       GELS 560
C                                                                       GELS 570
C     METHOD                                                            GELS 580
C        SOLUTION IS DONE BY MEANS OF GAUSS-ELIMINATION WITH            GELS 590
C        PIVOTING IN MAIN DIAGONAL, IN ORDER TO PRESERVE                GELS 600
C        SYMMETRY IN REMAINING COEFFICIENT MATRICES.                    GELS 610
C                                                                       GELS 620
C     ......................................................................GELS 630
C                                                                       GELS 640
      SUBROUTINE GELS(R,A,M,N,EPS,IER,AUX)                              GELS 650
C                                                                       GELS 660
C                                                                       GELS 670
      DIMENSION A(1),R(1),AUX(1)                                        GELS 680
      IF(M)24,24,1                                                      GELS 690
C                                                                       GELS 700
C     SEARCH FOR GREATEST MAIN DIAGONAL ELEMENT                         GELS 710
    1 IER=0                                                             GELS 720
      PIV=0.                                                            GELS 730
      L=0                                                               GELS 740
      DO 3 K=1,M                                                        GELS 750
      L=L+K                                                             GELS 760
      TB=ABS(A(L))                                                      GELS 770
      IF(TB-PIV)3,3,2                                                   GELS 780
    2 PIV=TB                                                            GELS 790
      I=L                                                               GELS 800
      J=K                                                               GELS 810
    3 CONTINUE                                                          GELS 820
      TOL=EPS*PIV                                                       GELS 830
C     MAIN DIAGONAL ELEMENT A(I)=A(J,J) IS FIRST PIVOT ELEMENT.         GELS 840
C     PIV CONTAINS THE ABSOLUTE VALUE OF A(I).                          GELS 850
C                                                                       GELS 860
C                                                                       GELS 870
C     START ELIMINATION LOOP                                            GELS 880
      LST=0                                                             GELS 890
      NM=N*M                                                            GELS 900
      LEND=M-1                                                          GELS 910
      DO 18 K=1,M                                                       GELS 920
C                                                                       GELS 930
C     TEST ON USEFULNESS OF SYMMETRIC ALGORITHM                         GELS 940
      IF(PIV)24,24,4                                                    GELS 950
    4 IF(IER)7,5,7                                                      GELS 960
    5 IF(PIV-TOL)6,6,7                                                  GELS 970
    6 IER=K-1                                                           GELS 980
    7 LT=J-K                                                            GELS 990
      LST=LST+K                                                         GELS1000
C                                                                       GELS1010
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R      GELS1020
      PIVI=1./A(I)                                                      GELS1030
      DO 8 L=K,NM,M                                                     GELS1040
      LL=L+LT                                                           GELS1050
      TB=PIVI*R(LL)                                                     GELS1060
      R(LL)=R(L)                                                        GELS1070
    8 R(L)=TB                                                           GELS1080
C                                                                       GELS1090
C     IS ELIMINATION TERMINATED                                         GELS1100
      IF(K-M)9,19,19                                                    GELS1110
C                                                                       GELS1120
C     ROW AND COLUMN INTERCHANGE AND PIVOT ROW REDUCTION IN MATRIX A.   GELS1130
C     ELEMENTS OF PIVOT COLUMN ARE SAVED IN AUXILIARY VECTOR AUX.       GELS1140
    9 LR=LST+(LT*(K+J-1))/2                                             GELS1150
      LL=LR                                                             GELS1160
      L=LST                                                             GELS1170
      DO 14 II=K,LEND                                                   GELS1180
      L=L+II                                                            GELS1190
      LL=LL+1                                                           GELS1200
      IF(L-LR)12,10,11                                                  GELS1210
   10 A(LL)=A(LST)                                                      GELS1220
      TB=A(L)                                                           GELS1230
      GO TO 13                                                          GELS1240
   11 LL=L+LT                                                           GELS1250
   12 TB=A(LL)                                                          GELS1260
      A(LL)=A(L)                                                        GELS1270
   13 AUX(II)=TB                                                        GELS1280
   14 A(L)=PIVI*TB                                                      GELS1290
C
```

```
C     SAVE COLUMN INTERCHANGE INFORMATION
      A(LST)=LT
C
C     ELEMENT REDUCTION AND SEARCH FOR NEXT PIVOT
      PIV=0.
      LLST=LST
      LT=0
      DO 18 II=K,LEND
      PIVI=-AUX(II)
      LL=LLST
      LT=LT+1
      DO 15 LLD=II,LEND
      LL=LL+LLD
      L=LL+LT
   15 A(L)=A(L)+PIVI*A(LL)
      LLST=LLST+II
      LR=LLST+LT
      TB=ABS(A(LR))
      IF(TB-PIV)17,17,16
   16 PIV=TB
      I=LR
      J=II+1
   17 DO 18 LR=K,NM,M
      LL=LR+LT
   18 R(LL)=R(LL)+PIVI*R(LR)
C     END OF ELIMINATION LOOP
C
C
C     BACK SUBSTITUTION AND BACK INTERCHANGE
   19 IF(LEND)24,23,20
   20 II=M
      DO 22 I=2,M
      LST=LST-II
      II=II-1
      L=A(LST)+.5
      DO 22 J=II,NM,M
      TB=R(J)
      LL=J
      K=LST
      DO 21 LT=II,LEND
      LL=LL+1
      K=K+LT
   21 TB=TB-A(K)*R(LL)
      K=J+L
      R(J)=R(K)
   22 R(K)=TB
   23 RETURN
C
C
C     ERROR RETURN
   24 IER=-1
      RETURN
      END
```

```
C
C     ******************************************************************
C
C        SUBROUTINE DGELS
C
C        PURPOSE
C           TO SOLVE A SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS WITH
C           SYMMETRIC COEFFICIENT MATRIX UPPER TRIANGULAR PART OF WHICH
C           IS ASSUMED TO BE STORED COLUMNWISE.
C
C        USAGE
C           CALL DGELS(R,A,M,N,EPS,IER,AUX)
C
C        DESCRIPTION OF PARAMETERS
C           R      - DOUBLE PRECISION M BY N RIGHT HAND SIDE MATRIX
C                    (DESTROYED). ON RETURN R CONTAINS THE SOLUTION OF
C                    THE EQUATIONS.
C           A      - UPPER TRIANGULAR PART OF THE SYMMETRIC DOUBLE
C                    PRECISION M BY M COEFFICIENT MATRIX.  (DESTROYED)
C           M      - THE NUMBER OF EQUATIONS IN THE SYSTEM.
C           N      - THE NUMBER OF RIGHT HAND SIDE VECTORS.
C           EPS    - SINGLE PRECISION INPUT CONSTANT WHICH IS USED AS
C                    RELATIVE TOLERANCE FOR TEST ON LOSS OF
C                    SIGNIFICANCE.
C           IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS
C                    IER=0  - NO ERROR,
C                    IER=-1 - NO RESULT BECAUSE OF M LESS THAN 1 OR
C                             PIVOT ELEMENT AT ANY ELIMINATION STEP
C                             EQUAL TO 0,
C                    IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-
C                             CANCE INDICATED AT ELIMINATION STEP K+1,
C                             WHERE PIVOT ELEMENT WAS LESS THAN OR
C                             EQUAL TO THE INTERNAL TOLERANCE EPS TIMES
C                             ABSOLUTELY GREATEST MAIN DIAGONAL
C                             ELEMENT OF MATRIX A.
C           AUX    - DOUBLE PRECISION AUXILIARY STORAGE ARRAY
C                    WITH DIMENSION M-1.
C
C        REMARKS
C           UPPER TRIANGULAR PART OF MATRIX A IS ASSUMED TO BE STORED
C           COLUMNWISE IN M*(M+1)/2 SUCCESSIVE STORAGE LOCATIONS, RIGHT
C           HAND SIDE MATRIX R COLUMNWISE IN N*M SUCCESSIVE STORAGE
C           LOCATIONS. ON RETURN SOLUTION MATRIX R IS STORED COLUMNWISE
C           TOO.
C           THE PROCEDURE GIVES RESULTS IF THE NUMBER OF EQUATIONS M IS
C           GREATER THAN 0 AND PIVOT ELEMENTS AT ALL ELIMINATION STEPS
C           ARE DIFFERENT FROM 0. HOWEVER WARNING IER=K - IF GIVEN -
C           INDICATES POSSIBLE LOSS OF SIGNIFICANCE. IN CASE OF A WELL
C           SCALED MATRIX A AND APPROPRIATE TOLERANCE EPS, IER=K MAY BE
C           INTERPRETED THAT MATRIX A HAS THE RANK K. NO WARNING IS
C           GIVEN IN CASE M=1.
C           ERROR PARAMETER IER=-1 DOES NOT NECESSARILY MEAN THAT
C           MATRIX A IS SINGULAR, AS ONLY MAIN DIAGONAL ELEMENTS
C           ARE USED AS PIVOT ELEMENTS. POSSIBLY SUBROUTINE DGELG (WHICH
C           WORKS WITH TOTAL PIVOTING) WOULD BE ABLE TO FIND A SOLUTION.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SOLUTION IS DONE BY MEANS OF GAUSS-ELIMINATION WITH
C           PIVOTING IN MAIN DIAGONAL, IN ORDER TO PRESERVE
C           SYMMETRY IN REMAINING COEFFICIENT MATRICES.
C
C     ******************************************************************
C
      SUBROUTINE DGELS(R,A,M,N,EPS,IER,AUX)
C
      DIMENSION A(1),R(1),AUX(1)
      DOUBLE PRECISION R,A,AUX,PIV,TB,TOL,PIVI
      IF(M)24,24,1
C
C     SEARCH FOR GREATEST MAIN DIAGONAL ELEMENT
    1 IER=0
      PIV=0.D0
      L=0
      DO 3 K=1,M
      L=L+K
      TB=DABS(A(L))
      IF(TB-PIV)3,3,2
    2 PIV=TB
      I=L
      J=K
    3 CONTINUE
      TOL=EPS*PIV
C     MAIN DIAGONAL ELEMENT A(I)=A(J,J) IS FIRST PIVOT ELEMENT.
C     PIV CONTAINS THE ABSOLUTE VALUE OF A(I).
C
C
C     START ELIMINATION LOOP
      LST=0
      NM=N*M
      LEND=M-1
      DO 18 K=1,M
C
C     TEST ON USEFULNESS OF SYMMETRIC ALGORITHM
      IF(PIV)24,24,4
    4 IF(IER)7,5,7
    5 IF(PIV-TOL)6,6,7
    6 IER=K-1
    7 LT=J-K
      LST=LST+K
C
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R
      PIVI=1.D0/A(I)
      DO 3 L=K,NM,M
      LL=L+LT
      TB=PIVI*R(LL)
      R(LL)=R(L)
    8 R(L)=TB
C
C     IS ELIMINATION TERMINATED
      IF(K-M)9,19,19
C
C     ROW AND COLUMN INTERCHANGE AND PIVOT ROW REDUCTION IN MATRIX A.
C     ELEMENTS OF PIVOT COLUMN ARE SAVED IN AUXILIARY VECTOR AUX.
    9 LR=LST+(LT+(K+J-1))/2
      LL=LR
      L=LST
      DO 14 II=K,LEND
      L=L+II
      LL=LL+1
      IF(L-LR)12,10,11
   10 A(LL)=A(LST)
      TB=A(L)
      GO TO 15
   11 LL=L+LT
   12 TB=A(LL)
      A(LL)=A(L)
   15 AUX(II)=TB
   14 A(L)=PIVI*TB
C
C     SAVE COLUMN INTERCHANGE INFORMATION
      A(LST)=LT
C
C     ELEMENT REDUCTION AND SEARCH FOR NEXT PIVOT
      PIV=0.D0
      LLST=LST
      LT=0
      DO 18 II=K,LEND
      PIVI=-AUX(II)
      LL=LLST
      LT=LT+1
      DO 15 LLD=II,LEND
      LL=LL+LLD
      L=LL+LT
   15 A(L)=A(L)+PIVI*A(LL)
      LLST=LLST+II
      LR=LLST+LT
      TB=DABS(A(LR))
      IF(TB-PIV)17,17,16
   16 PIV=TB
      I=LR
      J=II+1
   17 DO 18 LR=K,NM,M
      LL=LR+LT
   18 R(LL)=R(LL)+PIVI*R(LR)
C     END OF ELIMINATION LOOP
C
C
C     BACK SUBSTITUTION AND BACK INTERCHANGE
   19 IF(LEND)24,23,20
   20 II=M
      DO 22 I=2,M
      LST=LST-II
      II=II-1
      L=A(LST)+.500
      DO 22 J=II,NM,M
      TB=R(J)
      LL=J
      K=LST
      DO 21 LT=II,LEND
      LL=LL+1
      K=K+LT
   21 TB=TB-A(K)*R(LL)
      K=J+L
      R(J)=R(K)
   22 R(K)=TB
   23 RETURN
C
C
C     ERROR RETURN
   24 IER=-1
      RETURN
      END
```

Subroutines GELB and DGELB

These subroutines solve a system of simultaneous linear equations with a coefficient matrix of band structure by Gauss elimination. Consider the system of simultaneous linear equations

$$A * X = R \qquad (1)$$

with an m by m coefficient matrix A of band structure and an m by n right-hand side matrix R. Solution is done by means of Gauss elimination with column pivoting in matrix A. If R is the identity matrix, solution X is the inverse of matrix A. Solution matrix X is placed in positions of right-hand side matrix R and is stored in the same way -- that is, columnwise in m*n successive storage locations. Thus, the computation of the solution requires no extra m by n array of storage.

As storing matrix A by row results in a much more effective inner loop than storing by column, the procedure assumes that band matrix A is stored by row in successive storage locations, the number of which is equal to the number of elements appearing in the main diagonal and upper and lower codiagonals. The elements to be handed to the procedure by the user are underlined in the following explicit writing of formula (1) in the special case of a 7 by 7 matrix A with three upper and two lower codiagonals:

$$\begin{bmatrix} a_{\underline{11}} & a_{\underline{12}} & a_{\underline{13}} & a_{\underline{14}} & 0 & 0 & 0 \\ a_{\underline{21}} & a_{\underline{22}} & a_{\underline{23}} & a_{\underline{24}} & a_{\underline{25}} & 0 & 0 \\ a_{\underline{31}} & a_{\underline{32}} & a_{\underline{33}} & a_{\underline{34}} & a_{\underline{35}} & a_{\underline{36}} & 0 \\ 0 & a_{\underline{42}} & a_{\underline{43}} & a_{\underline{44}} & a_{\underline{45}} & a_{\underline{46}} & a_{\underline{47}} \\ 0 & 0 & a_{\underline{53}} & a_{\underline{54}} & a_{\underline{55}} & a_{\underline{56}} & a_{\underline{57}} \\ 0 & 0 & 0 & a_{\underline{64}} & a_{\underline{65}} & a_{\underline{66}} & a_{\underline{67}} \\ 0 & 0 & 0 & 0 & a_{\underline{75}} & a_{\underline{76}} & a_{\underline{77}} \end{bmatrix} * \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ x_{31} & x_{32} & \cdots & x_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{71} & x_{72} & \cdots & x_{7n} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ r_{31} & r_{32} & \cdots & r_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ r_{71} & r_{72} & \cdots & r_{7n} \end{pmatrix} \qquad (2)$$

Immediately following the matrix elements there must be some auxiliary storage locations used by the procedure, such that in the general case of an m by m matrix A with "mud" upper and "mld" lower codiagonals the total number of storage locations of array A is:+

$$ma = m \cdot mc - \frac{(mc-mld-1)(mc-mld)}{2} \qquad (3)$$

with mc = min (m, 1 + mud + mld). (4)

That means, in the special case of formula (2), that the 33 underlined elements of matrix A must be stored by row in the first 33 successive locations of array A, which consists of 36 storage locations.

In the first part of the procedure, matrix A is searched for its absolutely greatest element, say a_{ij}, and the absolute tolerance for testing loss of significance

$$tol = \varepsilon \cdot |a_{ij}| \qquad (5)$$

is generated with a given relative tolerance ε++. Close together with this search, the elements of matrix A are moved backward, and a triangular area of zeros is inserted in the last positions of the first (mc-mud-1) rows, according to storage requirements of the Gauss algorithm for band matrices. After these manipulations, the special matrix A of formula (2) will occupy the 36 positions of array A by row as shown in Figure 10 (the underlined elements are the main diagonal elements of matrix A).

In order to make the description of the band-algorithm easier, the elements of array A are renamed according to their arrangement in Figure 10,

$$\begin{array}{cccccc}
\underline{a_{11}} & a_{12} & a_{13} & a_{14} & 0 & 0 \\
a_{21} & \underline{a_{22}} & a_{23} & a_{24} & a_{25} & 0 \\
a_{31} & a_{32} & \underline{a_{33}} & a_{34} & a_{35} & a_{36} \\
a_{42} & a_{43} & \underline{a_{44}} & a_{45} & a_{46} & a_{47} \\
a_{53} & a_{54} & \underline{a_{55}} & a_{56} & a_{57} & \\
a_{64} & a_{65} & \underline{a_{66}} & a_{67} & & \\
a_{75} & a_{76} & \underline{a_{77}} & & &
\end{array}$$

Figure 10. Array A

thus generally giving the array shown in Figure 11 (main diagonal elements are underlined).

Each of the first $mr = m-(mc - mld - 1)$ rows in this array contains mc elements; in the following rows the number of elements decreases steadily by 1 until the last row, which contains mld + 1 elements.

Now Gauss elimination can start. In the first step, search the first column of matrix A that is, the elements of the first column of array A with row indexes from 1 up to 1 + mld) for the absolutely greatest element, say a_{j1} ($1 \leq j \leq 1 + mld$), and select it as pivot element ($p = a_{j1}$). Assume that j = 1. If it is not, interchange the first rows in array A and matrix R with the jth rows. Now, transform the elements of the pivot rows (that is, the first rows) of array A and matrix R by division with p, and the elements of the following mld rows by adding $-a_{\nu 1}$ times the first rows to the νth rows ($\nu = 2, 3, \ldots, 1 + mld$). At the same time, shift back the elements of the νth rows in array A by one position and insert a zero in the last position of each transformed row. This transformation is described by the following set of formulas:

$$a_{1l}^{(1)} = \frac{a_{1l}}{p} \quad (l = 1, 2, \ldots, mc) \tag{6}$$

$$r_{1l}^{(1)} = \frac{r_{1l}}{p} \quad (l = 1, 2, \ldots, n) \tag{7}$$

$$\left.\begin{aligned} a_{\nu,l-1}^{(1)} &= a_{\nu 1} - a_{\nu 1} \cdot a_{1l}^{(1)} \\ &\qquad (l = 2, 3, \ldots, mc) \\ a_{\nu,mc}^{(1)} &= 0 \end{aligned}\right\} \quad (\nu = 2, 3, \ldots, 1+mld) \tag{8}$$

$$\begin{array}{cccc}
\underline{a_{11}} & a_{12} & a_{13} \cdots\cdots a_{1,mc-1} & a_{1,mc} \\
a_{21} & \underline{a_{22}} & a_{23} \cdots\cdots a_{2,mc-1} & a_{2,mc} \\
a_{31} & a_{32} & \underline{a_{33}} \cdots\cdots a_{3,mc-1} & a_{3,mc} \\
\cdots & \cdots & \cdots & \cdots \\
a_{mr,1} & a_{mr,2} & a_{mr,3} \cdots a_{mr,mc-1} & a_{mr,mc} \\
a_{mr+1,1} & a_{mr+1,2} & a_{mr+1,3} \cdots a_{mr+1,mc-1} & \\
\cdots & \cdots & \cdots & \\
a_{m1} & a_{m2} & a_{m3} \cdots \underline{a_{m,mld+1}} &
\end{array}$$

Figure 11. Array A renamed

$$r_{\nu 1}^{(1)} = r_{\nu 1} - a_{\nu 1} \cdot r_{11}^{(1)} \quad (l = 1, 2, \ldots, n;\ \nu = 2, 3, \ldots, 1+mld) \tag{9}$$

Rows with row indexes greater than 1 + mld remain unchanged; nevertheless, they are marked with the superscript 1. The result of the first step is shown in Figure 12 (main diagonal elements are underlined).

$$R^{(1)} = \begin{pmatrix} r_{11}^{(1)} & r_{12}^{(1)} & \cdots & r_{1n}^{(1)} \\ r_{21}^{(1)} & r_{22}^{(1)} & \cdots & r_{2n}^{(1)} \\ r_{31}^{(1)} & r_{32}^{(1)} & \cdots & r_{3n}^{(1)} \\ & \vdots & & \\ r_{m1}^{(1)} & r_{m2}^{(1)} & \cdots & r_{mn}^{(1)} \end{pmatrix}$$

This procedure is now repeated (m-1) times, starting at the kth step with array $A^{(k-1)}$ and matrix $R^{(k-1)}$.

+Note that the user must provide this number of storage locations of array A by the dimension statement in his main program.

++In case of a matrix A without lower codiagonals, tol is set equal to zero, whether ε is zero or not.

$$
\begin{array}{ccccc}
\underline{1} & a^{(1)}_{12} & a^{(1)}_{13} & \cdots\ a^{(1)}_{1,mc-1} & a^{(1)}_{1,mc} \\
\underline{a^{(1)}_{21}} & a^{(1)}_{22} & a^{(1)}_{23} & \cdots\ a^{(1)}_{2,mc-1} & a^{(1)}_{2,mc} \\
a^{(1)}_{31} & \underline{a^{(1)}_{32}} & a^{(1)}_{33} & \cdots\ a^{(1)}_{3,mc-1} & a^{(1)}_{3,mc} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
a^{(1)}_{mr,1} & a^{(1)}_{mr,2} & a^{(1)}_{mr,3} & \cdots\ a^{(1)}_{mr,mc-1} & a^{(1)}_{mr,mc} \\
a^{(1)}_{mr+1,1} & a^{(1)}_{mr+1,2} & a^{(1)}_{mr+1,3} & \cdots\ a^{(1)}_{mr+1,mc-1} & \\
\cdots & \cdots & \cdots & \cdots & \\
a^{(1)}_{m1} & a^{(1)}_{m2} & a^{(1)}_{m3} & \cdots\ \underline{a^{(1)}_{m,mld+1}} &
\end{array}
$$

Figure 12. Array $A^{(1)}$

$$
\begin{array}{ccccc}
1 & a^{(1)}_{12} & a^{(1)}_{13} & \cdots\ a^{(1)}_{1,mc-1} & a^{(1)}_{1,mc} \\
1 & a^{(2)}_{22} & a^{(2)}_{23} & \cdots\ a^{(2)}_{2,mc-1} & a^{(2)}_{2,mc} \\
1 & a^{(3)}_{32} & a^{(3)}_{33} & \cdots\ a^{(3)}_{3,mc-1} & a^{(3)}_{3,mc} \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
1 & a^{(mr)}_{mr,2} & a^{(mr)}_{mr,3} & \cdots\ \cdots\ 0 \cdots 0 \cdots 0 & \\
\cdots & \cdots & \cdots & \cdots & \\
1 & 0 \cdots 0 \cdots 0 & & &
\end{array}
$$

mld zeros

mld zeros

Figure 13. Array $A^{(m)}$

The rows which must be worked through at the kth step are the rows with indexes $i = k, k+1, \ldots, ilr = \min(m, k + mld)$. As soon as ilr becomes greater than mr, the number of elements in any row of array A which have to be transformed decreases by one in each step k. The total results after m steps are shown in Figure 13.

$$
R^{(m)} = \begin{bmatrix}
r^{(1)}_{11} & r^{(1)}_{12} & \cdots & r^{(1)}_{1n} \\
r^{(2)}_{21} & r^{(2)}_{22} & \cdots & r^{(2)}_{2n} \\
r^{(3)}_{31} & r^{(3)}_{32} & \cdots & r^{(3)}_{3n} \\
 & \vdots & & \\
r^{(mr)}_{mr,1} & r^{(mr)}_{mr,2} & \cdots & r^{(mr)}_{mr,n} \\
 & \vdots & & \\
r^{(m)}_{m1} & r^{(m)}_{m2} & \cdots & r^{(m)}_{mn}
\end{bmatrix}
$$

Now, working backward, generate solution matrix X in the locations of right-hand side matrix R by means of the following equations:

$$
\left.
\begin{aligned}
r^{(m)}_{m-1,l} &= r^{(m-1)}_{m-1,l} - a^{(m-1)}_{m-1,2} \cdot r^{(m)}_{m,l} \qquad (l = 1, 2, \ldots, n) \\
r^{(m)}_{m-2,l} &= r^{(m-2)}_{m-2,l} - a^{(m-2)}_{m-2,2} \cdot r^{(m)}_{m-1,l} \\
&\quad - a^{(m-2)}_{m-2,3} \cdot r^{(m)}_{m,l} \qquad (l = 1, 2, \ldots, n) \\
&\cdots\cdots\cdots\cdots \\
r^{(m)}_{m-mc+1,l} &= r^{(m-mc+1)}_{m-mc+1,l} - a^{(m-mc+1)}_{m-mc+1,2} \\
&\quad \cdot r^{(m)}_{m-mc+2,l} - \cdots - a^{(m-mc+1)}_{m-mc-1,mc} \\
&\quad \cdot r^{(m)}_{m,l} \qquad (l = 1, 2, \ldots, n) \\
r^{(m)}_{11} &= r^{(1)}_{11} - a^{(1)}_{12} \cdot r^{(m)}_{21} - a^{(1)}_{13} \cdot r^{(m)}_{31} \\
&\quad - \cdots - a^{(1)}_{1,mc} \cdot r^{(m)}_{mc,l} \qquad (l = 1, 2, \ldots, n)
\end{aligned}
\right\} \quad (10)
$$

Note that in the first mc-1 equations of formula (10) the number of terms to be subtracted increases steadily by 1 until this number has reached mc-1. From the (mc-1)st equation up to the last equation the number of terms to be subtracted remains constant.

The only case in which the procedure described above can fail to give a solution occurs when at any step (say the kth step) all elements in the kth column of the transformed matrix A [that is, the elements in the first column of array A with row indexes k up to ilr = min (m, k + mld)] become zero, and no pivot element can be found. In this case, the procedure is bypassed and the error message ier = -1 is given. Actually, because of rounding errors, a further check of the absolute values of pivot elements is performed by the procedure. If at elimination step k this absolute value becomes less than tol (see equation 5 and second footnote), it is likely that matrix A is singular, too. But, as this may not necessarily be the case and as this test depends highly on the choice of the relative tolerance ϵ +, the procedure gives only the warning ier = k-1, which indicates that there is a possible loss of significance in the results computed by the algorithm. ++ In case of a well scaled matrix A and an appropriate choice of the relative tolerance, warning ier = k-1 may be interpreted as meaning that the rank of A is less than m but not less than k - 1. If there are no lower codiagonals (mld = 0), the test on loss of significance is suppressed.

+For subroutine GELB, a relative tolerance between 10^{-6} and 10^{-7} is suggested; and for subroutine DGELB, between 10^{-14} and 10^{-16}.

++For example, $\epsilon = 10^{-5}$ and warning ier = 3 mean that there is a possible loss of about 5 or more significant digits in the initial values of elimination step 4, and that matrix A seems to have a rank less than m but not less than 3.

```
C                                                                       GELB  10
C  ......................................................................GELB  20
C                                                                       GELB  30
C     SUBROUTINE GELB                                                   GELB  40
C                                                                       GELB  50
C     PURPOSE                                                           GELB  60
C        TO SOLVE A SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS WITH A      GELB  70
C        COEFFICIENT MATRIX OF BAND STRUCTURE.                          GELB  80
C                                                                       GELB  90
C     USAGE                                                             GELB 100
C        CALL GELB(R,A,M,N,MUD,MLD,EPS,IER)                             GELB 110
C                                                                       GELB 120
C     DESCRIPTION OF PARAMETERS                                         GELB 130
C        R      - M BY N RIGHT HAND SIDE MATRIX (DESTROYED).            GELB 140
C                 ON RETURN R CONTAINS THE SOLUTION OF THE EQUATIONS.GELB 150
C        A      - M BY M COEFFICIENT MATRIX WITH BAND STRUCTURE        GELB 160
C                 (DESTROYED).                                          GELB 170
C        M      - THE NUMBER OF EQUATIONS IN THE SYSTEM.               GELB 180
C        N      - THE NUMBER OF RIGHT HAND SIDE VECTORS.               GELB 190
C        MUD    - THE NUMBER OF UPPER CODIAGONALS (THAT MEANS          GELB 200
C                 CODIAGONALS ABOVE MAIN DIAGONAL).                     GELB 210
C        MLD    - THE NUMBER OF LOWER CODIAGONALS (THAT MEANS          GELB 220
C                 CODIAGONALS BELOW MAIN DIAGONAL).                     GELB 230
C        EPS    - AN INPUT CONSTANT WHICH IS USED AS RELATIVE          GELB 240
C                 TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.           GELB 250
C        IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS           GELB 260
C                 IER=0  - NO ERROR,                                    GELB 270
C                 IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARAME-     GELB 280
C                          TERS M,MUD,MLD OR BECAUSE OF PIVOT ELEMENTGELB 290
C                          AT ANY ELIMINATION STEP EQUAL TO 0.          GELB 300
C                 IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-     GELB 310
C                          CANCE INDICATED AT ELIMINATION STEP K+1,     GELB 320
C                          WHERE PIVOT ELEMENT WAS LESS THAN OR         GELB 330
C                          EQUAL TO THE INTERNAL TOLERANCE EPS TIMES GELB 340
C                          ABSOLUTELY GREATEST ELEMENT OF MATRIX A.     GELB 350
C                                                                       GELB 360
C     REMARKS                                                           GELB 370
C        BAND MATRIX A IS ASSUMED TO BE STORED ROWWISE IN THE FIRST     GELB 380
C        MC SUCCESSIVE STORAGE LOCATIONS OF TOTALLY NEEDED MA           GELB 390
C        STORAGE LOCATIONS, WHERE                                       GELB 400
C          MA=M*MC-ML*(ML+1)/2     AND     MC=MA-MU*(MU+1)/2     WITH  GELB 410
C          MC=MIN(M,1+MUD+MLD),  ML=MC-1-MLD,  MU=MC-1-MUD.             GELB 420
C        RIGHT HAND SIDE MATRIX R IS ASSUMED TO BE STORED COLUMNWISE    GELB 430
C        IN N*M SUCCESSIVE STORAGE LOCATIONS. ON RETURN SOLUTION        GELB 440
C        MATRIX R IS STORED COLUMNWISE TOO.                             GELB 450
C        INPUT PARAMETERS M, MUD, MLD SHOULD SATISFY THE FOLLOWING      GELB 460
C        RESTRICTIONS        MUD NOT LESS THAN ZERO                     GELB 470
C                            MLD NOT LESS THAN ZERO                     GELB 480
C                            MUD+MLD NOT GREATER THAN 2*M-2.            GELB 490
C        NO ACTION BESIDES ERROR MESSAGE IER=-1 TAKES PLACE IF THESE    GELB 500
C        RESTRICTIONS ARE NOT SATISFIED.                                GELB 510
C        THE PROCEDURE GIVES RESULTS IF THE RESTRICTIONS ON INPUT       GELB 520
C        PARAMETERS ARE SATISFIED AND IF PIVOT ELEMENTS AT ALL          GELB 530
C        ELIMINATION STEPS ARE DIFFERENT FROM 0. HOWEVER WARNING        GELB 540
C        IER=K - IF GIVEN - INDICATES POSSIBLE LOSS OF SIGNIFICANCE.    GELB 550
C        IN CASE OF A WELL SCALED MATRIX A AND APPROPRIATE TOLERANCE    GELB 560
C        EPS, IER=K MAY BE INTERPRETED THAT MATRIX A HAS THE RANK K.    GELB 570
C        NO WARNING IS GIVEN IF MATRIX A HAS NO LOWER CODIAGONAL.       GELB 580
C                                                                       GELB 590
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     GELB 600
C        NONE                                                           GELB 610
C                                                                       GELB 620
C     METHOD                                                            GELB 630
C        SOLUTION IS DONE BY MEANS OF GAUSS ELIMINATION WITH            GELB 640
C        COLUMN PIVOTING ONLY, IN ORDER TO PRESERVE BAND STRUCTURE      GELB 650
C        IN REMAINING COEFFICIENT MATRICES.                             GELB 660
C                                                                       GELB 670
C  ......................................................................GELB 680
C                                                                       GELB 690
      SUBROUTINE GELB(R,A,M,N,MUD,MLD,EPS,IER)                          GELB 700
C                                                                       GELB 710
C                                                                       GELB 720
      DIMENSION R(1),A(1)                                               GELB 730
C                                                                       GELB 740
C     TEST ON WRONG INPUT PARAMETERS                                    GELB 750
      IF(MLD)47,1,1                                                     GELB 760
    1 IF(MUD)47,2,2                                                     GELB 770
    2 MC=1+MLD+MUD                                                      GELB 780
      IF(MC+1-M)3,3,47                                                  GELB 790
C                                                                       GELB 800
C     PREPARE INTEGER PARAMETERS                                        GELB 810
C        MC=NUMBER OF COLUMNS IN MATRIX A                               GELB 820
C        MU=NUMBER OF ZEROS TO BE INSERTED IN FIRST ROW OF MATRIX A     GELB 830
C        ML=NUMBER OF MISSING ELEMENTS IN LAST ROW OF MATRIX A          GELB 840
C        MR=INDEX OF LAST ROW IN MATRIX A WITH MC ELEMENTS              GELB 850
C        MZ=TOTAL NUMBER OF ZEROS TO BE INSERTED IN MATRIX A            GELB 860
C        MA=TOTAL NUMBER OF STORAGE LOCATIONS NECESSARY FOR MATRIX A    GELB 870
C        NM=NUMBER OF ELEMENTS IN MATRIX R                              GELB 880
    3 IF(MC-M)5,5,4                                                     GELB 890
    4 MC=M                                                              GELB 900
    5 MU=MC-MUD-1                                                       GELB 910
      ML=MC-MLD-1                                                       GELB 920
      MR=M-ML                                                           GELB 930
      MZ=(MU*(MU+1))/2                                                  GELB 940
      MA=M*MC-(ML*(ML+1))/2                                             GELB 950
      NM=N*M                                                            GELB 960
C                                                                       GELB 970
C     MOVE ELEMENTS BACKWARD AND SEARCH FOR ABSOLUTELY GREATEST ELEMENT GELB 980
C     (NOT NECESSARY IN CASE OF A MATRIX WITHOUT LOWER CODIAGONALS)     GELB 990
      IER=0                                                             GELB1000
      PIV=0.                                                            GELB1010
      IF(MLD)14,14,6                                                    GELB1020
    6 JJ=MA                                                             GELB1030
      J=MA-MZ                                                           GELB1040
      KST=J                                                             GELB1050
      DO 9 K=1,KST                                                      GELB1060
      TB=A(J)                                                           GELB1070
      A(JJ)=TB                                                          GELB1080
      TB=ABS(TB)                                                        GELB1090
      IF(TB-PIV)8,8,7                                                   GELB1100
    7 PIV=TB                                                            GELB1110
    8 J=J-1                                                             GELB1120
    9 JJ=JJ-1                                                           GELB1130
C                                                                       GELB1140
C     INSERT ZEROS IN FIRST MU ROWS (NOT NECESSARY IN CASE MZ=0)        GELB1150
      IF(MZ)14,14,10                                                    GELB1160
   10 JJ=1                                                              GELB1170
      J=1+MZ                                                            GELB1180
      IC=1+MUD                                                          GELB1190
      DO 13 I=1,MU                                                      GELB1200
      DO 12 K=1,MC                                                      GELB1210
      A(JJ)=0.                                                          GELB1220
      IF(K-IC)11,11,12                                                  GELB1230
   11 A(JJ)=A(J)                                                        GELB1240
      J=J+1                                                             GELB1250
   12 JJ=JJ+1                                                           GELB1260
   13 IC=IC+1                                                           GELB1270
C                                                                       GELB1280
```

```
C     GENERATE TEST VALUE FOR SINGULARITY
   14 TOL=EPS*PIV
C
C
C     START DECOMPOSITION LOOP
      KST=1
      IDST=MC
      IC=MC-1
      DO 38 K=1,M
      IF(K-MR-1)16,16,15
   15 IDST=IDST-1
   16 ID=IDST
      ILR=K+MLD
      IF(ILR-M)18,18,17
   17 ILR=M
   18 II=KST
C
C     PIVOT SEARCH IN FIRST COLUMN (ROW INDEXES FROM I=K UP TO I=ILR)
      PIV=0.
      DO 22 I=K,ILR
      TB=ABS(A(II))
      IF(TB-PIV)20,20,19
   19 PIV=TB
      J=I
      JJ=II
   20 IF(I-MR)22,22,21
   21 ID=ID-1
   22 II=II+ID
C
C     TEST ON SINGULARITY
      IF(PIV)47,47,23
   23 IF(IER)26,24,26
   24 IF(PIV-TOL)25,25,26
   25 IER=K-1
   26 PIV=1./A(JJ)
C
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R
      ID=J-K
      DO 27 I=K,NM,M
      II=I+ID
      TB=PIV*R(II)
      R(II)=R(I)
   27 R(I)=TB
C
C     PIVOT ROW REDUCTION AND ROW INTERCHANGE IN COEFFICIENT MATRIX A
      II=KST
      J=JJ+IC
      DO 28 I=JJ,J
      TB=PIV*A(II)
      A(I)=A(II)
      A(II)=TB
   28 II=II+1
C
C     ELEMENT REDUCTION
      IF(K-ILR)29,34,34
   29 ID=KST
      II=K+1
      MU=KST+1
      MZ=KST+IC
      DO 33 I=II,ILR
C
C     IN MATRIX A
      ID=ID+MC
      JJ=I-MR-1
      IF(JJ)31,31,30
   30 ID=ID-JJ
   31 PIV=-A(ID)
      J=ID+1
      DO 32 JJ=MU,MZ
      A(J-1)=A(JJ)+PIV*A(JJ)
   32 J=J+1
      A(J-1)=0.
C
C     IN MATRIX R
      J=K
      DO 33 JJ=I,NM,M
      R(JJ)=R(JJ)+PIV*R(J)
   33 J=J+M
   34 KST=KST+MC
      IF(ILR-MR)36,35,35
   35 IC=IC-1
   36 ID=K-MR
      IF(ID)38,38,37
   37 KST=KST-ID
   38 CONTINUE
C     END OF DECOMPOSITION LOOP
C
C
C     BACK SUBSTITUTION
      IF(MC-1)46,46,39
   39 IC=2
      KST=MA+ML-MC+2
      II=M
      DO 45 I=2,M
      KST=KST-MC
      II=II-1
      J=II-MR
      IF(J)41,41,40
   40 KST=KST+J
   41 DO 43 J=II,NM,M
      TB=R(J)
      MZ=KST+IC-2
      ID=J
      DO 42 JJ=KST,MZ
      ID=ID+1
   42 TB=TB-A(JJ)*R(ID)
   43 R(J)=TB
      IF(IC-MC)44,45,45
   44 IC=IC+1
   45 CONTINUE
   46 RETURN
C
C
C     ERROR RETURN
   47 IER=-1
      RETURN
      END
```

```
C
C
C
C
C     SUBROUTINE DGELB
C
C     PURPOSE
C        TO SOLVE A SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS WITH A
C        COEFFICIENT MATRIX OF BAND STRUCTURE.
C
C     USAGE
C        CALL DGELB(R,A,M,N,MUD,MLD,EPS,IER)
C
C     DESCRIPTION OF PARAMETERS
C        R     - DOUBLE PRECISION M BY N RIGHT HAND SIDE MATRIX
C                (DESTROYED). ON RETURN R CONTAINS THE SOLUTION
C                OF THE EQUATIONS.
C        A     - DOUBLE PRECISION M BY M COEFFICIENT MATRIX WITH
C                BAND STRUCTURE (DESTROYED).
C        M     - THE NUMBER OF EQUATIONS IN THE SYSTEM.
C        N     - THE NUMBER OF RIGHT HAND SIDE VECTORS.
C        MUD   - THE NUMBER OF UPPER CODIAGONALS (THAT MEANS
C                CODIAGONALS ABOVE MAIN DIAGONAL).
C        MLD   - THE NUMBER OF LOWER CODIAGONALS (THAT MEANS
C                CODIAGONALS BELOW MAIN DIAGONAL).
C        EPS   - SINGLE PRECISION INPUT CONSTANT WHICH IS USED AS
C                RELATIVE TOLERANCE FOR TEST ON LOSS OF
C                SIGNIFICANCE.
C        IER   - RESULTING ERROR PARAMETER CODED AS FOLLOWS
C                IER=0  - NO ERROR,
C                IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARAME-
C                         TERS M,MUD,MLD OR BECAUSE OF PIVOT ELEMENT
C                         AT ANY ELIMINATION STEP EQUAL TO C,
C                IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-
C                         CANCE INDICATED AT ELIMINATION STEP K+1,
C                         WHERE PIVOT ELEMENT WAS LESS THAN OR
C                         EQUAL TO THE INTERNAL TOLERANCE EPS TIMES
C                         ABSOLUTELY GREATEST ELEMENT OF MATRIX A.
C
C     REMARKS
C        BAND MATRIX A IS ASSUMED TO BE STORED ROWWISE IN THE FIRST
C        ME SUCCESSIVE STORAGE LOCATIONS OF TOTALLY NEEDED MA
C        STORAGE LOCATIONS, WHERE
C          MA=M*MC-ML*(ML+1)/2     AND     ME=MA-MU*(MU+1)/2     WITH
C          MC=MIN(M,1+MUD+MLD),  ML=MC-1-MLD,  MU=MC-1-MUD.
C        RIGHT HAND SIDE MATRIX R IS ASSUMED TO BE STORED COLUMNWISE
C        IN N*M SUCCESSIVE STORAGE LOCATIONS. ON RETURN SOLUTION
C        MATRIX R IS STORED COLUMNWISE TOO.
C        INPUT PARAMETERS M, MUD, MLD SHOULD SATISFY THE FOLLOWING
C        RESTRICTIONS          MUD NOT LESS THAN ZERO
C                              MLD NOT LESS THAN ZERO
C                              MUD+MLD NOT GREATER THAN 2*M-2.
C        NO ACTION BESIDES ERROR MESSAGE IER=-1 TAKES PLACE IF THESE
C        RESTRICTIONS ARE NOT SATISFIED.
C        THE PROCEDURE GIVES RESULTS IF THE RESTRICTIONS ON INPUT
C        PARAMETERS ARE SATISFIED AND IF PIVOT ELEMENTS AT ALL
C        ELIMINATION STEPS ARE DIFFERENT FROM C. HOWEVER WARNING
C        IER=K - IF GIVEN - INDICATES POSSIBLE LOSS OF SIGNIFICANCE.
C        IN CASE OF A WELL SCALED MATRIX A AND APPROPRIATE TOLERANCE
C        EPS, IER=K MAY BE INTERPRETED THAT MATRIX A HAS THE RANK K.
C        NO WARNING IS GIVEN IF MATRIX A HAS NO LOWER CODIAGONAL.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        NONE
C
C     METHOD
C        SOLUTION IS DONE BY MEANS OF GAUSS ELIMINATION WITH
C        COLUMN PIVOTING ONLY, IN ORDER TO PRESERVE BAND STRUCTURE
C        IN REMAINING COEFFICIENT MATRICES.
C
C     ..................................................................
C
      SUBROUTINE DGELB(R,A,M,N,MUD,MLD,EPS,IER)
C
C
      DIMENSION R(1),A(1)
      DOUBLE PRECISION R,A,PIV,TB,TOL
C
C     TEST ON WRONG INPUT PARAMETERS
      IF(MLD)47,1,1
    1 IF(MUD)47,2,2
    2 MC=1+MLD+MUD
      IF(MC+1-M-M)3,3,47
C
C     PREPARE INTEGER PARAMETERS
C        MC=NUMBER OF COLUMNS IN MATRIX A
C        MU=NUMBER OF ZEROS TO BE INSERTED IN FIRST ROW OF MATRIX A
C        ML=NUMBER OF MISSING ELEMENTS IN LAST ROW OF MATRIX A
C        MR=INDEX OF LAST ROW IN MATRIX A WITH MC ELEMENTS
C        MZ=TOTAL NUMBER OF ZEROS TO BE INSERTED IN MATRIX A
C        MA=TOTAL NUMBER OF STORAGE LOCATIONS NECESSARY FOR MATRIX A
C        NM=NUMBER OF ELEMENTS IN MATRIX R
    3 IF(MC-M)5,5,4
    4 MC=M
    5 MU=MC-MUD-1
      ML=MC-MLD-1
      MR=M-ML
      MZ=(MU*(MU+1))/2
      MA=M*MC-(ML*(ML+1))/2
      NM=N*M
C
C     MOVE ELEMENTS BACKWARD AND SEARCH FOR ABSOLUTELY GREATEST ELEMENT
C     (NOT NECESSARY IN CASE OF A MATRIX WITHOUT LOWER CODIAGONALS)
      IER=0
      PIV=0.D0
      IF(MLD)14,14,6
    6 JJ=MA
      J=MA-MZ
      KST=J
      DO 9 K=1,KST
      TB=A(J)
      A(JJ)=TB
      TB=DABS(TB)
      IF(TB-PIV)8,8,7
    7 PIV=TB
    8 J=J-1
    9 JJ=JJ-1
C
C     INSERT ZEROS IN FIRST MU ROWS (NOT NECESSARY IN CASE MZ=0)
      IF(MZ)14,14,10
   10 JJ=1
      J=1+MZ
      IC=1+MUD
      DO 13 I=1,MU
      DO 12 K=1,MC
      A(JJ)=0.D0
      IF(K-IC)11,11,12
   11 A(JJ)=A(J)
      J=J+1
   12 JJ=JJ+1
```

```
13 IC=IC+1

   GENERATE TEST VALUE FOR SINGULARITY
14 TOL=EPS*PIV


   START DECOMPOSITION LOOP
   KST=1
   IDST=MC
   IC=MC-1
   DO 38 K=1,M
   IF(K-MR-1)16,16,15
15 IDST=IDST-1
16 ID=IDST
   ILR=K+MLD
   IF(ILR-M)18,18,17
17 ILR=M
18 II=KST

   PIVOT SEARCH IN FIRST COLUMN (ROW INDEXES FROM I=K UP TO I=ILR)
   PIV=0.0D
   DO 22 I=K,ILR
   TB=DABS(A(II))
   IF(TB-PIV)20,20,19
19 PIV=TB
   J=I
   JJ=II
20 IF(I-MR)22,22,21
21 ID=ID-1
22 II=II+ID

   TEST ON SINGULARITY
   IF(PIV)47,47,23
23 IF(ILR)26,24,26
24 IF(PIV-TOL)25,25,26
25 IER=K-1
26 PIV=1.0D/A(JJ)

   PIVOT ROW REDUCTION AND ROW INTERCHANGE IN RIGHT HAND SIDE R
   ID=J-K
   DO 27 I=K,NM,M
   II=I+ID
   TB=PIV*R(II)
   R(II)=R(I)
27 R(I)=TB

   PIVOT ROW REDUCTION AND ROW INTERCHANGE IN COEFFICIENT MATRIX A
   II=KST
   J=JJ+IC
   DO 28 I=JJ,J
   TB=PIV*A(I)
   A(I)=A(II)
   A(II)=TB
29 II=II+1

   ELEMENT REDUCTION
   IF(K-ILR)29,34,34
29 ID=KST
   II=K+1
   MD=KST+1
   MZ=KST+IC
   DO 33 I=II,ILR

   IN MATRIX A
   ID=ID+MC
   JJ=I-MR-1
   IF(JJ)31,31,30
30 ID=ID-JJ
31 PIV=-A(ID)
   J=ID+1
   DO 32 JJ=MD,MZ
   A(J-1)=A(J)+PIV*A(JJ)
32 J=J+1
   A(J-1)=0.0D

   IN MATRIX R
   J=K
   DO 33 JJ=I,NM,M
   R(JJ)=R(JJ)+PIV*R(J)
33 J=J+M
34 KST=KST+MC
   IF(ILR-MR)36,35,35
35 IC=IC-1
36 ID=K-MR
   IF(ID)38,38,37
37 KST=KST-ID
38 CONTINUE
   END OF DECOMPOSITION LOOP


   BACK SUBSTITUTION
   IF(MC-1)46,46,39
39 IC=2
   KST=MA+ML-MC+2
   II=M
   DO 45 I=2,M
   KST=KST-MC
   II=II-1
   J=II-MR
   IF(J)41,41,40
40 KST=KST+J
41 DO 43 J=II,NM,M
   TB=R(J)
   MZ=KST+IC-2
   ID=J
   DO 42 JJ=KST,MZ
   ID=ID+1
42 TB=TB-A(JJ)*R(ID)
43 R(J)=TB
   IF(IC-MC)44,45,45
44 IC=IC+1
45 CONTINUE
46 RETURN


   ERROR RETURN
47 IER=-1
   RETURN
   END
```

Subroutines MTDS and DMTDS

Given a general matrix A and a nonsingular upper triangular matrix T, these subroutines will perform one of the following six operations, depending on the value of an input parameter IOP:

IOP = 1: A is replaced by $T^{-1}A$.
IOP = -1: A is replaced by AT^{-1}.
IOP = 2: A is replaced by $(T^{-1})^T A$.
IOP = -2: A is replaced by $A(T^{-1})^T$.
IOP = 3: A is replaced by $(T^T T)^{-1}A$.
IOP = -3: A is replaced by $A(T^T T)^{-1}$.

1. Mathematical background

a. Calculation of $X = T^{-1}A$ is done using backward substitution to obtain X from $TX = A$.

b. Calculation of $Y = (T^{-1})^T A$ is done using forward substitution to obtain Y from $T^T Y = A$.

c. Calculation of $Z = (T^T T)^{-1}A$ is done by first solving $T^T Y = A$ and then solving $TZ = Y$.

The remaining three operations are reducible to the above three.

2. Applications

Subroutines MTDS and DMTDS may be used to compute the solution of a system of equations $BX = A$ with symmetric positive definite coefficient matrix B. The first step towards the solution is the triangular factorization of B by means of subroutine MFSD or DMFSD. The second step, which may be repeated for different sets of right-hand sides A, is the calculation of $(T^T T)^{-1}A$ by means of MTDS or DMTDS with IOP = 3. Another useful application is the computation of the product $A^T B^{-1} A$ with symmetric positive definite B and arbitrary A in only three steps and without additional storage requirements:

a. Replace B by T where $B = T^T T$.
b. Replace A by $C = (T^T)^{-1}A$.
c. Replace B by $C^T C$.

```
C
C     ..................................................................
C
C        SUBROUTINE MTDS
C
C        PURPOSE
C           MULTIPLY A GENERAL MATRIX A ON THE LEFT OR RIGHT BY
C           INVERSE(T),INVERSE(TRANSPOSE(T)) OR INVERSE(TRANSPOSE(T*T))
C           THE TRIANGULAR MATRIX T IS STORED COLUMNWISE IN COMPRESSED
C           FORM, I.E. UPPER TRIANGULAR PART ONLY.
C
C        USAGE
C           CALL MTDS(A,M,N,T,IOP,IER)
C
C        DESCRIPTION OF PARAMETERS
C           A     - GIVEN GENERAL MATRIX WHITH M ROWS AND N COLUMNS.
C           M     - NUMBER OF ROWS OF MATRIX A
C           N     - NUMBER OF COLUMNS OF MATRIX A
C           T     - GIVEN TRIANGULAR MATRIX STORED COLUMNWISE UPPER
C                   TRIANGULAR PART ONLY. ITS NUMBER OF ROWS AND
C                   COLUMNS K IS IMPLIED BY COMPATIBILITY.
C                   K = M IF IOP IS POSITIVE,
C                   K = N IF IOP IS NEGATIVE.
C                   T OCCUPIES K*(K+1)/2 STORAGE POSITIONS.
C           IOP   - INPUT VARIABLE FOR SELECTION OF OPERATION
C                   IOP = 1 - A IS REPLACED BY INVERSE(T)*A
C                   IOP =-1 - A IS REPLACED BY A*INVERSE(T)
C                   IOP = 2 - A IS REPLACED BY INVERSE(TRANSPOSE(T))*A
C                   IOP =-2 - A IS REPLACED BY A*INVERSE(TRANSPOSE(T))
C                   IOP = 3 - A IS REPLACED BY INVERSE(TRANSPOSE(T)*T)*A
C                   IOP =-3 - A IS REPLACED BY A*INVERSE(TRANSPOSE(T)*T)
C           IER   - RESULTING ERROR PARAMETER
C                   IER =-1 MEANS M AND N ARE NOT BOTH POSITIVE
C                           AND/OR IOP IS ILLEGAL
C                   IER = 0 MEANS OPERATION WAS SUCCESSFUL
C                   IER = 1 MEANS TRIANGULAR MATRIX T IS SINGULAR
C
C        REMARKS
C           SUBROUTINE MTDS MAY BE USED TO CALCULATE THE SOLUTION OF
C           A SYSTEM OF EQUATIONS WITH SYMMETRIC POSITIVE DEFINITE
C           COEFFICIENT MATRIX. THE FIRST STEP TOWARDS THE SOLUTION
C           IS TRIANGULAR FACTORIZATION BY MEANS OF MFSD. THE SECOND
C           STEP IS APPLICATION OF MTDS.
C           SUBROUTINES MFSD AND MTDS MAY BE USED IN ORDER TO CALCULATE
C           THE PRODUCT TRANSPOSE(A)*INVERSE(B)*A WITH GIVEN SYMMETRIC
C           POSITIVE DEFINITE B AND GIVEN A EFFICIENTLY IN THREE STEPS
C           1) TRIANGULAR FACTORIZATION OF B (B=TRANSPOSE(T)*T)
C           2) MULTIPLICATION OF A ON THE LEFT BY INVERSE(TRANSPOSE(T))
C              A IS REPLACED BY C=INVERSE(TRANSPOSE(T))*A
C           3) CALCULATION OF THE RESULT FORMING TRANSPOSE(C)*C
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           CALCULATION OF X = INVERSE(T)*A IS DONE USING BACKWARD
C           SUBSTITUTION TO OBTAIN X FROM T*X = A.
C           CALCULATION OF Y = INVERSE(TRANSPOSE(T))*A IS DONE USING
C           FORWARD SUBSTITUTION TO OBTAIN Y FROM TRANSPOSE(T)*Y = A.
C           CALCULATION OF Z = INVERSE(TRANSPOSE(T)*T)*A IS DONE
C           SOLVING FIRST TRANSPOSE(T)*Y = A AND THEN T*Z = Y, IE.
C           USING THE ABOVE TWO STEPS IN REVERSE ORDER
C
C     ..................................................................
C
      SUBROUTINE MTDS(A,M,N,T,IOP,IER)
C
C
      DIMENSION A(1),T(1)
      DOUBLE PRECISION DSUM
C
C        TEST OF DIMENSION
      IF(M)2,2,1
    1 IF(N)2,2,4
C
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSIONS
    2 IER=-1
      RETURN
C
C        ERROR RETURN IN CASE OF SINGULAR MATRIX T
    3 IER=1
      RETURN
C
C        INITIALIZE DIVISION PROCESS
    4 MN=M*N
      MM=M*(M+1)/2
      MM1=M-1
      IER=0
      ICS=M
      IRS=1
      IMEND=M
C
C        TEST OF SPECIFIED OPERATION
      IF(IOP)5,2,6
    5 MM=M*(M+1)/2
      MM1=N-1
      IRS=M
      ICS=1
      IMEND=MN-M+1
      MN=M
    6 IOPF=MOD(IOP+3,3)
      IF(IABS(IOP)-3)7,7,2
    7 IF(IOPF-1)8,18,8
C
C        INITIALIZE SOLUTION OF TRANSPOSE(T)*X = A
    8 MEND=1
      LLD=IRS
      MSTA=1
      MDEL=1
      MX=1
      LD=1
      LX=0
C
C        TEST FOR NONZERO DIAGONAL TERM IN T
    9 IF(T(MSTA))10,3,10
   10 DO 11 I=MEND,MN,ICS
   11 A(I)=A(I)/DPLE(T(MSTA))
C
C        IS M EQUAL 1
      IF(MM1)12,15,12
   12 DO 14 J=1,MM1
      MSTA=MSTA+MDEL
      MDEL=MDEL+MX
      DO 14 I=MEND,MN,ICS
      DSUM=0.D0
      L=MSTA
      LDX=LD
      LL=I
      DO 13 K=1,J
      DSUM=DSUM-T(L)*A(LL)
      LL=LL+LLD
      L=L+LDX
   13 LDX=LDX+LX
      IF(T(L))14,3,14
   14 A(LL)=(DSUM+A(LL))/T(L)
C
C        TEST END OF OPERATION
   15 IF(IER)16,17,16
   16 IER=0
      RETURN
   17 IF(IOPF)18,18,16
C
C        INITIALIZE SOLUTION OF T*X = A
   18 IER=1
      MEND=IMEND
      MN=M*M
      LLD=-IRS
      MSTA=MM
      MDEL=-1
      MX=0
      LD=-MM1
      LX=1
      GOTO 9
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DMTDS
C
C        PURPOSE
C           MULTIPLY A GENERAL MATRIX A ON THE LEFT OR RIGHT BY
C           INVERSE(T),INVERSE(TRANSPOSE(T)) OR INVERSE(TRANSPOSE(T*T))
C           THE TRIANGULAR MATRIX T IS STORED COLUMNWISE IN COMPRESSED
C           FORM, I.E. UPPER TRIANGULAR PART ONLY.
C
C        USAGE
C           CALL DMTDS(A,M,N,T,IOP,IER)
C
C        DESCRIPTION OF PARAMETERS
C           A     - GIVEN GENERAL MATRIX WITH  M ROWS AND N COLUMNS.
C                   A MUST BE OF DOUBLE PRECISION
C           M     - NUMBER OF ROWS OF MATRIX A
C           N     - NUMBER OF COLUMNS OF MATRIX A
C           T     - GIVEN TRIANGULAR MATRIX STORED COLUMNWISE UPPER
C                   TRIANGULAR PART ONLY. ITS NUMBER OF ROWS AND
C                   COLUMNS K IS IMPLIED BY COMPATIBILITY.
C                   K = M IF IOP IS POSITIVE.
C                   K = N IF IOP IS NEGATIVE.
C                   T OCCUPIES K*(K+1)/2 STORAGE POSITIONS.
C                   T MUST BE OF DOUBLE PRECISION
C           IOP   - INPUT VARIABLE FOR SELECTION OF OPERATION
C                   IOP = 1 - A IS REPLACED BY INVERSE(T)*A
C                   IOP =-1 - A IS REPLACED BY A*INVERSE(T)
C                   IOP = 2 - A IS REPLACED BY INVERSE(TRANSPOSE(T))*A
C                   IOP =-2 - A IS REPLACED BY A*INVERSE(TRANSPOSE(T))
C                   IOP = 3 - A IS REPLACED BY INVERSE(TRANSPOSE(T)*T)*A
C                   IOP =-3 - A IS REPLACED BY A*INVERSE(TRANSPOSE(T)*T)
C           IER   - RESULTING ERROR PARAMETER
C                   IER =-1 MEANS M AND N ARE NOT BOTH POSITIVE
C                           AND/OR IOP IS ILLEGAL
C                   IER = 0 MEANS OPERATION WAS SUCCESSFUL
C                   IER = 1 MEANS TRIANGULAR MATRIX T IS SINGULAR
C
C        REMARKS
C           SUBROUTINE DMTDS MAY BE USED TO CALCULATE THE SOLUTION OF
C           A SYSTEM OF EQUATIONS WITH SYMMETRIC POSITIVE DEFINITE
C           COEFFICIENT MATRIX. THE FIRST STEP TOWARDS THE SOLUTION
C           IS TRIANGULAR FACTORIZATION BY MEANS OF DMFSD, THE SECOND
C           STEP IS APPLICATION OF DMTDS.
C           SUBROUTINES DMFSD AND DMTDS MAY BE USED IN ORDER TO
C           CALCULATE THE PRODUCT TRANSPOSE(A)*INVERSE(B)*A WITH GIVEN
C           SYMMETRIC POSITIVE DEFINITE B AND GIVEN A IN THREE STEPS
C           1) TRIANGULAR FACTORIZATION OF B (B=TRANSPOSE(T)*T)
C           2) MULTIPLICATION OF A ON THE LEFT BY INVERSE(TRANSPOSE(T))
C              A IS REPLACED BY C=INVERSE(TRANSPOSE(T))*A
C           3) CALCULATION OF THE RESULT FORMING TRANSPOSE(C)*C
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           CALCULATION OF X = INVERSE(T)*A IS DONE USING BACKWARD
C           SUBSTITUTION TO OBTAIN X FROM T*X = A.
C           CALCULATION OF Y = INVERSE(TRANSPOSE(T))*A IS DONE USING
C           FORWARD SUBSTITUTION TO OBTAIN Y FROM TRANSPOSE(T)*Y = A.
C           CALCULATION OF Z = INVERSE(TRANSPOSE(T)*T)*A IS DONE
C           SOLVING FIRST TRANSPOSE(T)*Y = A AND THEN T*Z = Y, IE.
C           USING THE ABOVE TWO STEPS IN REVERSE ORDER
```

```
C     ..................................................................
C
      SUBROUTINE DMTDS(A,M,N,T,IOP,IER)
C
C
      DIMENSION A(1),T(1)
      DOUBLE PRECISION DSUM,A,T
C
C        TEST OF DIMENSION
      IF(M)2,2,1
    1 IF(N)2,2,4
C
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSIONS
    2 IER=-1
      RETURN
C
C        ERROR RETURN IN CASE OF SINGULAR MATRIX T
    3 IER=1
      RETURN
C
C        INITIALIZE DIVISION PROCESS
    4 MN=M*N
      MM=M*(M+1)/2
      MM1=M-1
      IER=0
      ICS=M
      IRS=1
      IMEND=M
C
C        TEST SPECIFIED OPERATION
      IF(IOP)5,2,6
    5 MM=N*(N+1)/2
      MM1=N-1
      IRS=M
      ICS=1
      IMEND=MN-M+1
      MN=M
    6 IOPE=MOD(IOP+3,3)
      IF(IABS(IOP)-3)7,7,2
    7 IF(IOPE-1)8,18,8
C
C        INITIALIZE SOLUTION OF TRANSPOSE(T)*X = A
    8 MEND=1
      LLD=IRS
      MSTA=1
      MDEL=1
      MX=1
      LD=1
      LX=0
C
C        TEST FOR NONZERO DIAGONAL TERM IN T
    9 IF(T(MSTA))10,3,10
   10 DO 11 I=MEND,MN,ICS
   11 A(I)=A(I)/T(MSTA)
C
C        IS M EQUAL 1
      IF(MM1)2,15,12
   12 DO 14 J=1,MM1
      MSTA=MSTA+MDEL
      MDEL=MDEL+MX
      DO 14 I=MEND,MN,ICS
      DSUM=0.D0
      L=MSTA
      LDX=LD
      LL=I
      DO 13 K=1,J
      DSUM=DSUM-T(L)*A(LL)
      LL=LL+LLD
      L=L+LDX
   13 LDX=LDX+LX
      IF(T(L))14,3,14
   14 A(LL)=(DSUM+A(LL))/T(L)
C
C        TEST END OF OPERATION
   15 IF(IER)16,17,16
   16 IER=0
      RETURN
   17 IF(IOPE)18,18,16
C
C        INITIALIZE SOLUTION OF T*X = A
   18 IER=1
      MEND=IMEND
      MN=M*N
      LLD=-IRS
      MSTA=MM
      MDEL=-1
      MX=0
      LD=-MM1
      LX=1
      GOTO 9
      END
```

Subroutines MLSS and DMLSS

These subroutines will compute the least-squares solution of minimal length of a system of linear equations

$$AX = R \tag{1}$$

with symmetric positive semidefinite coefficient matrix A. That is, they will determine that vector X of smallest Euclidean length $\|X\|$ for which $\|AX - R\| =$ minimum.

The matrix A is not used directly as input. Instead, MLSS expects the matrices U, T, and T_u and the permutation P that would be obtained if subroutine MFSS were used with A as input matrix (for details see MFSS under "Final result of elimination").

The given vector R is replaced by the solution vector X.

1. Mathematical background

Instead of (1), solve the system of equations:

$$(PAP^T)PX = PR \tag{2}$$

Any solution PX of (2) determines the solution P^{-1} (PX) of (1); therefore it suffices to consider (2).

Using the decomposition:

$$PAP^T = \begin{bmatrix} I & O \\ U^T & I \end{bmatrix} \begin{bmatrix} A_s & O \\ O & O \end{bmatrix} \begin{bmatrix} I & U \\ O & I \end{bmatrix} \tag{3}$$

where $A_s = (T)^T T$ (see MFSS), it is easy to see that (2) is equivalent to the system:

$$A_s(X_1 + UX_2) = R_1 \tag{4}$$

$$OX_2 = R_2 - U^T R_1$$

where $PX = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$ and $PR = \begin{bmatrix} R_1 \\ R_2 \end{bmatrix}$. If rank(A) = n, formula (3) reduces to $PAP^T = A_s$, and (4) is

$$A_s X = R \tag{4'}$$

In solving (2), or equivalently (4), three cases must be distinguished:

a. Rank(A) = n. Then the unique solution PX is given by $PX = A_s^{-1} R$, which is computed easily using the triangular decomposition $A_s = (T)^T T$.

b. Rank(A) = r < n and $R_2 = U^T R_1$ (a compatible system). Then the solution of minimal length is:

$$X_1 = (I + UU^T)^{-1} T^{-1} (T^{-1})^T R_1$$

$$X_2 = U^T X_1$$

Using the identity $(I + UU^T)^{-1} = I - U(I + U^T U)^{-1} U^T$ and the triangular factorization of $(I + U^T U) = T_u^T T_u$, X_1 may be computed by the following sequence of operations:

$$X_1 = T^{-1}(T^T)^{-1} R_1$$

$$X_2 = T_u^{-1}(T_u^{-1})^T U^T X_1$$

$$X_1 = X_1 - UX_2$$

c. Rank(A) = r< n and $R_2 \neq U^T R_1$ (an incompatible system). Then the least-squares solution of minimal length is:

$$X_1 = (I + UU^T)^{-1} T^{-1} (T^T)^{-1} (I + UU^T)^{-1} (R_1 + UR_2)$$

$$X_2 = U^T X_1$$

X_1 may be computed using the following sequence of operations:

$$X_1 = R_1 + UR_2$$

$$X_2 = -T_u^{-1}(T_u^{-1})^T U^T X_1$$

$$X_1 = T^{-1}(T^T)^{-1}(X_1 + UX_2)$$

$$X_2 = -T_u^{-1}(T_u^{-1})^T U^T X_1$$

$$X_1 = X_1 + UX_2$$

2. Programming considerations

Cases a, b and c above are easily combined into one sequence of operations:

(1) $X = PR$ (interchange given right-hand side)
(2) $X_1 = X_1 + U X_2$
(3) $X_2 = -U^T X_1$
(4) $X_2 = (T_u)^{-1} ((T_u)^T)^{-1} X_2$
(5) $X_1 = X_1 + U X_2$
(6) $X_1 = T^{-1}(T^T)^{-1} X_1$
(7) $X_2 = -U^T X_1$
(8) $X_2 = (T_u)^{-1} ((T_u)^T)^{-1} X_2$
(9) $X_1 = X_1 + U X_2$
(10) $X_2 = (U^T) X_1$
(11) $X = P^{-1} X$ (reinterchange calculated solution)

In case a, the only steps performed are (1), (6), (11).
In case b, only steps (1) and (6) up to (11) are performed.
In case c, all steps are performed.

```
C                                                                       MLSS  10
C     ******************************************************************MLSS  20
C                                                                       MLSS  30
C        SUBROUTINE MLSS                                                MLSS  40
C                                                                       MLSS  50
C        PURPOSE                                                        MLSS  60
C           SUBROUTINE MLSS IS THE SECOND STEP IN THE PROCEDURE FOR     MLSS  70
C           CALCULATING THE LEAST SQUARES SOLUTION OF MINIMAL LENGTH    MLSS  80
C           OF A SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS WITH SYMMETRIC MLSS  90
C           POSITIVE SEMI-DEFINITE COEFFICIENT MATRIX.                  MLSS 100
C                                                                       MLSS 110
C        USAGE                                                          MLSS 120
C           CALL MLSS(A,N,IRANK,TRAC,INC,RHS,IER)                       MLSS 130
C                                                                       MLSS 140
C        DESCRIPTION OF PARAMETERS                                      MLSS 150
C           A    - COEFFICIENT MATRIX IN FACTORED FORM AS GENERATED     MLSS 160
C                  BY SUBROUTINE MFSS FROM INITIALLY GIVEN SYMMETRIC    MLSS 170
C                  COEFFICIENT MATRIX A STORED IN N*(N+1)/2 LOCATIONS   MLSS 180
C                  A REMAINS UNCHANGED                                  MLSS 190
C           N    - DIMENSION OF COEFFICIENT MATRIX                      MLSS 200
C           IRANK - RANK OF COEFFICIENT MATRIX, CALCULATED BY MEANS OF  MLSS 210
C                  SUBROUTINE MFSS                                      MLSS 220
C           TRAC - VECTOR OF DIMENSION N CONTAINING THE                 MLSS 230
C                  SUBSCRIPTS OF PIVOT ROWS AND COLUMNS, I.E. THE       MLSS 240
C                  PRODUCT REPRESENTATION IN TRANSPOSITIONS OF THE      MLSS 250
C                  PERMUTATION WHICH WAS APPLIED TO ROWS AND COLUMNS    MLSS 260
C                  OF A IN THE FACTORIZATION PROCESS                    MLSS 270
C                  TRAC IS A RESULTANT ARRAY OF SUBROUTINE MFSS         MLSS 280
C           INC  - INPUT VARIABLE WHICH SHOULD CONTAIN THE VALUE ZERO   MLSS 290
C                  IF THE SYSTEM OF SIMULTANEOUS EQUATIONS IS KNOWN     MLSS 300
C                  TO BE COMPATIBLE AND A NONZERO VALUE OTHERWISE       MLSS 310
C           RHS  - VECTOR OF DIMENSION N CONTAINING THE RIGHT HAND SIDEMLSS 320
C                  ON RETURN RHS CONTAINS THE MINIMAL LENGTH SOLUTION   MLSS 330
C           IER  - RESULTANT ERROR PARAMETER                            MLSS 340
C                  IER = 0 MEANS NO ERRORS                              MLSS 350
C                  IER =-1 MEANS N AND/OR IRANK IS NOT POSITIVE AND/OR  MLSS 360
C                          IRANK IS GREATER THAN N                      MLSS 370
C                  IER = 1 MEANS THE FACTORIZATION CONTAINED IN A HAS   MLSS 380
C                          ZERO DIVISORS AND/OR TRAC CONTAINS           MLSS 390
C                          VALUES OUTSIDE THE FEASIBLE RANGE 1 UP TO N  MLSS 400
C                                                                       MLSS 410
C        REMARKS                                                        MLSS 420
C           THE MINIMAL LENGTH SOLUTION IS PRODUCED IN THE STORAGE      MLSS 430
C           LOCATIONS OCCUPIED BY THE RIGHT HAND SIDE.                  MLSS 440
C           SUBROUTINE MLSS DOES TAKE CARE OF THE PERMUTATION           MLSS 450
C           WHICH WAS APPLIED TO ROWS AND COLUMNS OF A.                 MLSS 460
C           OPERATION IS BYPASSED IN CASE OF A NON POSITIVE VALUE       MLSS 470
C           OF IRANK                                                    MLSS 480
C                                                                       MLSS 490
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MLSS 500
C           NONE                                                        MLSS 510
C                                                                       MLSS 520
C        METHOD                                                         MLSS 530
C           LET T, U, TU BE THE COMPONENTS OF THE FACTORIZATION OF A,   MLSS 540
C           AND LET THE RIGHT HAND SIDE BE PARTITIONED INTO A FIRST     MLSS 550
C           PART X1 OF DIMENSION IRANK AND A SECOND PART X2 OF DIMENSIONMLSS 560
C           N-IRANK. THEN THE FOLLOWING OPERATIONS ARE APPLIED IN       MLSS 570
C           SEQUENCE                                                    MLSS 580
C           (1) INTERCHANGE RIGHT HAND SIDE                             MLSS 590
C           (2) X1 = X1 + U * X2                                        MLSS 600
C           (3) X2 =-TRANSPOSE(U) * X1                                  MLSS 610
C           (4) X2 = INVERSE(TU) * INVERSE(TRANSPOSE(TU)) * X2          MLSS 620
C           (5) X1 = X1 + U * X2                                        MLSS 630
C           (6) X1 = INVERSE(T) * INVERSE(TRANSPOSE(T)) * X1            MLSS 640
C           (7) X2 =-TRANSPOSE(U) * X1                                  MLSS 650
C           (8) X2 = INVERSE(TU) * INVERSE(TRANSPOSE(TU)) * X2          MLSS 660
C           (9) X1 = X1 + U * X2                                        MLSS 670
C           (10)X2 = TRANSPOSE(U) * X1                                  MLSS 680
C           (11) REINTERCHANGE CALCULATED SOLUTION                      MLSS 690
C           IF THE SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS IS SPECIFIED MLSS 700
C           TO BE COMPATIBLE THEN STEPS (2), (3), (4) AND (5) ARE       MLSS 710
C           CANCELLED.                                                  MLSS 720
C           IF THE COEFFICIENT MATRIX HAS RANK N, THEN THE ONLY STEPS   MLSS 730
C           PERFORMED ARE (1), (6) AND (11).                            MLSS 740
C                                                                       MLSS 750
C     ******************************************************************MLSS 760
C                                                                       MLSS 770
      SUBROUTINE MLSS(A,N,IRANK,TRAC,INC,RHS,IER)                       MLSS 780
C                                                                       MLSS 790
C                                                                       MLSS 800
C        DIMENSIONED DUMMY VARIABLES                                    MLSS 810
      DIMENSION A(1),TRAC(1),RHS(1)                                     MLSS 820
      DOUBLE PRECISION SUM                                              MLSS 830
C                                                                       MLSS 840
C        TEST OF SPECIFIED DIMENSIONS                                   MLSS 850
      IDEF=N-IRANK                                                      MLSS 860
      IF(N)33,33,1                                                      MLSS 870
    1 IF(IRANK)33,33,2                                                  MLSS 880
    2 IF(IDEF)33,3,3                                                    MLSS 890
C                                                                       MLSS 900
C        CALCULATE AUXILIARY VALUES                                     MLSS 910
    3 ITE=IRANK*(IRANK+1)/2                                             MLSS 920
      IX2=IRANK+1                                                       MLSS 930
      NP1=N+1                                                           MLSS 940
      IER=0                                                             MLSS 950
C                                                                       MLSS 960
C        INTERCHANGE RIGHT HAND SIDE                                    MLSS 970
      JJ=1                                                              MLSS 980
      II=1                                                              MLSS 990
    4 DO 6 I=1,N                                                        MLSS1000
      J=TRAC(II)                                                        MLSS1010
      IF(J)31,31,5                                                      MLSS1020
    5 HOLD=RHS(II)                                                      MLSS1030
      RHS(II)=RHS(J)                                                    MLSS1040
      RHS(J)=HOLD                                                       MLSS1050
    6 II=II+JJ                                                          MLSS1060
      IF(JJ)32,7,7                                                      MLSS1070
C                                                                       MLSS1080
C        PERFORM STEP 2 IF NECESSARY                                    MLSS1090
    7 ISW=1                                                             MLSS1100
      IF(INC*IDEF)8,28,8                                                MLSS1110
C                                                                       MLSS1120
C        CALCULATE X1 = X1 + U * X2                                     MLSS1130
    8 ISTA=ITE                                                          MLSS1140
      DO 10 I=1,IRANK                                                   MLSS1150
      ISTA=ISTA+1                                                       MLSS1160
      JJ=ISTA                                                           MLSS1170
      SUM=0.D0                                                          MLSS1180
      DO 9 J=IX2,N                                                      MLSS1190
      SUM=SUM+A(JJ)*RHS(J)                                              MLSS1200
    9 JJ=JJ+J                                                           MLSS1210
   10 RHS(I)=RHS(I)+SUM                                                 MLSS1220
      GOTO(11,28,11),ISW                                                MLSS1230
C                                                                       MLSS1240
C        CALCULATE X2 = TRANSPOSE(U) * X1                               MLSS1250
   11 ISTA=ITE                                                          MLSS1260
      DO 15 I=IX2,N                                                     MLSS1270
      JJ=ISTA                                                           MLSS1280
      SUM=0.D0                                                          MLSS1290
      DO 12 J=1,IRANK                                                   MLSS1300
      JJ=JJ+1                                                           MLSS1310
   12 SUM=SUM+A(JJ)*RHS(J)                                              MLSS1320
      GOTO(13,13,14),ISW                                                MLSS1330
   13 SUM=-SUM                                                          MLSS1340
   14 RHS(I)=SUM                                                        MLSS1350
   15 ISTA=ISTA+I                                                       MLSS1360
      GOTO(16,29,30),ISW                                                MLSS1370
C                                                                       MLSS1380
C        INITIALIZE STEP (4) OR STEP (6)                                MLSS1390
   16 ISTA=IX2                                                          MLSS1400
      IEND=N                                                            MLSS1410
      JJ=ITE+ISTA                                                       MLSS1420
C                                                                       MLSS1430
C        DIVISION OF X1 BY TRANSPOSE OF TRIANGULAR MATRIX               MLSS1440
   17 SUM=0.D0                                                          MLSS1450
      DO 20 I=ISTA,IEND                                                 MLSS1460
      IF(A(JJ))18,31,18                                                 MLSS1470
   18 RHS(I)=(RHS(I)-SUM)/A(JJ)                                         MLSS1480
      IF(I-IEND)19,21,21                                                MLSS1490
   19 JJ=JJ+ISTA                                                        MLSS1500
      SUM=0.D0                                                          MLSS1510
      DO 20 J=ISTA,I                                                    MLSS1520
      SUM=SUM+A(JJ)*RHS(J)                                              MLSS1530
   20 JJ=JJ+1                                                           MLSS1540
C                                                                       MLSS1550
C        DIVISION OF X1 BY TRIANGULAR MATRIX                            MLSS1560
   21 SUM=0.D0                                                          MLSS1570
      II=IEND                                                           MLSS1580
      DO 24 I=ISTA,IEND                                                 MLSS1590
      RHS(II)=(RHS(II)-SUM)/A(JJ)                                       MLSS1600
      IF(II-ISTA)25,25,22                                               MLSS1610
   22 KK=JJ-1                                                           MLSS1620
      SUM=0.D0                                                          MLSS1630
      DO 23 J=II,IEND                                                   MLSS1640
      SUM=SUM+A(KK)*RHS(J)                                              MLSS1650
   23 KK=KK+J                                                           MLSS1660
      JJ=JJ-II                                                          MLSS1670
   24 II=II-1                                                           MLSS1680
   25 IF(IDEF)26,30,26                                                  MLSS1690
   26 GOTO(27,11,8),ISW                                                 MLSS1700
C                                                                       MLSS1710
C        PERFORM STEP (5)                                               MLSS1720
   27 ISW=2                                                             MLSS1730
      GOTO 8                                                            MLSS1740
C                                                                       MLSS1750
C        PERFORM STEP (6)                                               MLSS1760
   28 ISTA=1                                                            MLSS1770
      IEND=IRANK                                                        MLSS1780
      JJ=1                                                              MLSS1790
      ISW=2                                                             MLSS1800
      GOTO 17                                                           MLSS1810
C                                                                       MLSS1820
C        PERFORM STEP (8)                                               MLSS1830
   29 ISW=3                                                             MLSS1840
      GOTO 16                                                           MLSS1850
C                                                                       MLSS1860
C        REINTERCHANGE CALCULATED SOLUTION                              MLSS1870
   30 II=N                                                              MLSS1880
      JJ=-1                                                             MLSS1890
      GOTO 4                                                            MLSS1900
C                                                                       MLSS1910
C        ERROR RETURN IN CASE OF ZERO DIVISOR                           MLSS1920
   31 IER=1                                                             MLSS1930
   32 RETURN                                                            MLSS1940
C                                                                       MLSS1950
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSION                      MLSS1960
   33 IER=-1                                                            MLSS1970
      RETURN                                                            MLSS1980
      END                                                               MLSS1990
```

```
C                                                                       DMLS  10
C     ******************************************************************DMLS  20
C                                                                       DMLS  30
C        SUBROUTINE DMLSS                                               DMLS  40
C                                                                       DMLS  50
C        PURPOSE                                                        DMLS  60
C           SUBROUTINE DMLSS IS THE SECOND STEP IN THE PROCEDURE FOR    DMLS  70
C           CALCULATING THE LEAST SQUARES SOLUTION OF MINIMAL LENGTH    DMLS  80
C           OF A SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS WITH SYMMETRIC DMLS  90
C           POSITIVE SEMI-DEFINITE COEFFICIENT MATRIX.                  DMLS 100
C                                                                       DMLS 110
C        USAGE                                                          DMLS 120
C           CALL DMLSS(A,N,IRANK,TRAC,INC,RHS,IER)                      DMLS 130
C                                                                       DMLS 140
C        DESCRIPTION OF PARAMETERS                                      DMLS 150
C           A    - COEFFICIENT MATRIX IN FACTORED FORM AS GENERATED     DMLS 160
C                  BY SUBROUTINE MFSS FROM INITIALLY GIVEN SYMMETRIC    DMLS 170
C                  COEFFICIENT MATRIX A STORED IN N*(N+1)/2 LOCATIONS   DMLS 180
C                  A REMAINS UNCHANGED                                  DMLS 190
C                  A MUST BE OF DOUBLE PRECISION                        DMLS 200
C           N    - DIMENSION OF COEFFICIENT MATRIX                      DMLS 210
C           IRANK - RANK OF COEFFICIENT MATRIX, CALCULATED BY MEANS OF  DMLS 220
C                  SUBROUTINE DMFSS                                     DMLS 230
C           TRAC - VECTOR OF DIMENSION N CONTAINING THE                 DMLS 240
C                  SUBSCRIPTS OF PIVOT ROWS AND COLUMNS, I.E. THE       DMLS 250
C                  PRODUCT REPRESENTATION IN TRANSPOSITIONS OF THE      DMLS 260
C                  PERMUTATION WHICH WAS APPLIED TO ROWS AND COLUMNS    DMLS 270
C                  OF A IN THE FACTORIZATION PROCESS                    DMLS 280
C                  TRAC IS A RESULTANT ARRAY OF SUBROUTINE MFSS         DMLS 290
C                  TRAC MUST BE OF DOUBLE PRECISION                     DMLS 300
C           INC  - INPUT VARIABLE WHICH SHOULD CONTAIN THE VALUE ZERO   DMLS 310
C                  IF THE SYSTEM OF SIMULTANEOUS EQUATIONS IS KNOWN     DMLS 320
C                  TO BE COMPATIBLE AND A NONZERO VALUE OTHERWISE       DMLS 330
C           RHS  - VECTOR OF DIMENSION N CONTAINING THE RIGHT HAND SIDEDMLS 340
C                  ON RETURN RHS CONTAINS THE MINIMAL LENGTH SOLUTION   DMLS 350
C                  RHS MUST BE OF DOUBLE PRECISION                      DMLS 360
C           IER  - RESULTANT ERROR PARAMETER                            DMLS 370
C                  IER = 0 MEANS NO ERRORS                              DMLS 380
C                  IER =-1 MEANS N AND/OR IRANK IS NOT POSITIVE AND/OR DMLS 390
C                          IRANK IS GREATER THAN N                      DMLS 400
C                  IER = 1 MEANS THE FACTORIZATION CONTAINED IN A HAS  DMLS 410
C                          ZERO DIVISORS AND/OR TRAC CONTAINS           DMLS 420
C                          VALUES OUTSIDE THE FEASIBLE RANGE 1 UP TO N DMLS 430
C                                                                       DMLS 440
C        REMARKS                                                        DMLS 450
C           THE MINIMAL LENGTH SOLUTION IS PRODUCED IN THE STORAGE      DMLS 460
C           LOCATIONS OCCUPIED BY THE RIGHT HAND SIDE.                  DMLS 470
C           SUBROUTINE DMLSS DOES TAKE CARE OF THE PERMUTATION          DMLS 480
C           WHICH WAS APPLIED TO ROWS AND COLUMNS OF A.                 DMLS 490
C           OPERATION IS BYPASSED IN CASE OF A NON POSITIVE VALUE       DMLS 500
C           OF IRANK                                                    DMLS 510
C                                                                       DMLS 520
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DMLS 530
C           NONE                                                        DMLS 540
```

```
C
C        METHOD
C           LET T, U, TU BE THE COMPONENTS OF THE FACTORIZATION OF A,
C           AND LET THE RIGHT HAND SIDE BE PARTITIONED INTO A FIRST
C           PART X1 OF DIMENSION IRANK AND A SECOND PART X2 OF DIMENSION
C           N-IRANK. THEN THE FOLLOWING OPERATIONS ARE APPLIED IN
C           SEQUENCE
C           (1) INTERCHANGE RIGHT HAND SIDE
C           (2) X1 = X1 + U * X2
C           (3) X2 =-TRANSPOSE(U) * X1
C           (4) X2 = INVERSE(TU) * INVERSE(TRANSPOSE(TU)) * X2
C           (5) X1 = X1 + U * X2
C           (6) X1 = INVERSE(T) * INVERSE(TRANSPOSE(T)) * X1
C           (7) X2 =-TRANSPOSE(U) * X1
C           (8) X2 = INVERSE(TU) * INVERSE(TRANSPOSE(TU)) * X2
C           (9) X1 = X1 + U * X2
C           (10)X2 = TRANSPOSE(U) * X1
C           (11) REINTERCHANGE CALCULATED SOLUTION
C           IF THE SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS IS SPECIFIED
C           TO BE COMPATIBLE THEN STEPS (2), (3), (4) AND (5) ARE
C           CANCELLED.
C           IF THE COEFFICIENT MATRIX HAS RANK N, THEN THE ONLY STEPS
C           PERFORMED ARE (1), (6) AND (11).
C
C     ..................................................................
C
      SUBROUTINE DMLSS(A,N,IRANK,TRAC,INC,RHS,IER)
C
C
C        DIMENSIONED DUMMY VARIABLES
      DIMENSION A(1),TRAC(1),RHS(1)
      DOUBLE PRECISION SUM,A,RHS,TRAC,HOLD
C
C        TEST OF SPECIFIED DIMENSIONS
      IDEF=N-IRANK
      IF(N)33,33,1
    1 IF(IRANK)33,33,2
    2 IF(IDEF)33,3,3
C
C        CALCULATE AUXILIARY VALUES
    3 ITE=IRANK*(IRANK+1)/2
      IX2=IRANK+1
      NP1=N+1
      IER=0
C
C        INTERCHANGE RIGHT HAND SIDE
      JJ=1
      II=1
    4 DO 6 I=1,N
      J=TRAC(II)
      IF(J)31,31,5
    5 HOLD=RHS(II)
      RHS(II)=RHS(J)
      RHS(J)=HOLD
    6 II=II+JJ
      IF(JJ)32,7,7
C
C        PERFORM STEP 2 IF NECESSARY
    7 ISW=1
      IF(INC*IDEF)8,28,8
C
C        CALCULATE X1 = X1 + U * X2
    8 ISTA=ITE
      DO 10 I=1,IRANK
      ISTA=ISTA+1
      JJ=ISTA
      SUM=0.D0
      DO 9 J=IX2,N
      SUM=SUM+A(JJ)*RHS(J)
    9 JJ=JJ+J
   10 RHS(I)=RHS(I)+SUM
      GOTO(11,29,11),ISW
C
C        CALCULATE X2 = TRANSPOSE(U) * X1
   11 ISTA=ITE
      DO 15 I=IX2,N
      JJ=ISTA
      SUM=0.D0
      DO 12 J=1,IRANK
      JJ=JJ+1
   12 SUM=SUM+A(JJ)*RHS(J)
      GOTO(13,13,14),ISW
   13 SUM=-SUM
   14 RHS(I)=SUM
   15 ISTA=ISTA+I
      GOTO(16,29,30),ISW
C
C        INITIALIZE STEP (4) OR STEP (8)
   16 ISTA=IX2
      IEND=N
      JJ=ITE+ISTA
C
C        DIVISION OF X1 BY TRANSPOSE OF TRIANGULAR MATRIX
   17 SUM=0.D0
      DO 20 I=ISTA,IEND
      IF(A(JJ))18,31,18
   18 RHS(I)=(RHS(I)-SUM)/A(JJ)
      IF(I-IEND)19,21,21
   19 JJ=JJ+ISTA
      SUM=0.D0
      DO 20 J=ISTA,I
      SUM=SUM+A(JJ)*RHS(J)
   20 JJ=JJ+1
C
C        DIVISION OF X1 BY TRIANGULAR MATRIX
   21 SUM=0.D0
      II=IEND
      DO 24 I=ISTA,IEND
      RHS(II)=(RHS(II)-SUM)/A(JJ)
      IF(II-ISTA)25,25,22
   22 KK=JJ-1
      SUM=0.D0
      DO 23 J=II,IEND
      SUM=SUM+A(KK)*RHS(J)
   23 KK=KK+J
      JJ=JJ-II
   24 II=II-1
   25 IF(IDEF)26,30,26
   26 GOTO(27,11,8),ISW
C
C        PERFORM STEP (5)
   27 ISW=2
      GOTO 8
C
C        PERFORM STEP (6)
   28 ISTA=1
      IEND=IRANK
      JJ=1
      ISW=2
      GOTO 17
C
C        PERFORM STEP (8)
   29 ISW=3
      GOTO 16
C
C        REINTERCHANGE CALCULATED SOLUTION
   30 II=N
      JJ=-1
      GOTO 4
C
C        ERROR RETURN IN CASE OF ZERO DIVISOR
   31 IER=1
   32 RETURN
C
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSION
   33 IER=-1
      RETURN
      END
```

Subroutines MCHB and DMCHB

These subroutines will compute a triangular factorization

$$A = (T_u)^T T_u \tag{1}$$

of an m by m symmetric positive definite band matrix A. They may also be used to replace a given n by n matrix R by $A^{-1}R$, $T_u^{-1}R$, or $(T_u^{-1})^T R$, depending on the value of an input parameter IOP, as shown in Table 3.

1. Mathematical background

The upper triangular matrix T_u is computed using the following recursive formulas:

$$t_{kj} = (1/t_{kk})\left(a_{kj} - \sum_{i=1}^{k-1} t_{ik}t_{ij}\right) \quad k = 1, \ldots, m \quad j = k, k+1, \ldots, m \tag{2}$$

(Any symbol $\sum_{i=j}^{k} x_i$ is to be interpreted as zero if $j > k$.)

In the special case $j = k$, equation (2) may be written:

$$t_{kk} = \sqrt{a_{kk} - \sum_{i=1}^{k-1} t_{ik}^2}, \quad k = 1, 2, \ldots, m \tag{3}$$

When A is a symmetric band matrix with mud upper codiagonals, all elements a_{kj} are equal to zero if $j > k + mud$. From formulas (2) and (3) it is easy to see that the same is true for t_{kj}:

$$t_{kj} = 0 \text{ if } j > k + mud \tag{4}$$

Therefore formulas (2) and (3) reduce to:

$$t_{kj} = (1/t_{kk})\left(a_{kj} - \sum_{i=i_o}^{k-1} t_{ik}t_{ij}\right) \quad k = 1, 2, \ldots, m \quad j = k+1, \ldots, \min(m, k+mud) \tag{5}$$

$$i_o = \max(1, j-mud)$$

$$t_{kk} = \sqrt{a_{kk} - \sum_{i=i_o}^{k-1} t_{ik}^2} \quad k = 1, 2, \ldots, m \tag{6}$$

$$i_o = \max(1, k-mud)$$

Computing $(T_u^{-1})^T$ is equivalent to solving for Y in the system of equations:

$$T_u^T Y = R \tag{7}$$

The solution of (7) is computed using the following recursive scheme:

$$y_{kj} = (1/t_{kk})\left(r_{kj} - \sum_{i=i_o}^{k-1} t_{ik}y_{ij}\right) \quad k = 1, 2, \ldots, m \quad j = 1, 2, \ldots, n \tag{8}$$

$$i_o = \max(1, k-mud)$$

After each y_{kj} is computed, it may be stored in the location of r_{kj}.

Analogously, computing $T_u^{-1}R$ is the same as solving for X in the system of equations:

$$T_u X = R \tag{9}$$

The solution of (9) is given using a similar recursive scheme:

$$x_{kj} = (1/t_{kk})\left(r_{kj} - \sum_{i=k+1}^{i_o} t_{ki}x_{ij}\right) \quad k = m, m-1, \ldots, 1 \quad j = 1, 2, \ldots, n \tag{10}$$

$$i_o = \min(m, k+mud)$$

$A^{-1}R$ is computed by first computing $S = (T_u^T)^{-1}R$ in the locations of R, and then computing $T_u^{-1}S$, again in the locations of R. If $R = I$, this process replaces R with the inverse A^{-1} of A. Note that in general A^{-1} is no longer a band matrix.

2. Programming considerations

The input matrix A is assumed to be stored in compressed form, that is, main diagonal and mud upper codiagonals rowwise in successive storage locations. Therefore, the total storage requirement for matrix A is m + mud (2m-mud-1)/2 storage locations. The general m by n matrix R is assumed to be stored columnwise in nm successive storage locations.

The operations performed by the subroutines depend on the actual value of the decision parameter IOP.

Table 3 shows input and output depending on the value of IOP.

Table 3. Dependence of Input and Output on IOP

	IOP	0	1	-1	2	-2	3	-3
INPUT	A in compressed form	A	A	T_u	A	T_u	A	T_u
	R column-wise	irrelevant	R	R	R	R	R	R
OUTPUT	A in compressed form	T_u	T_u	T_u	T_u	T_u	T_u	T_u
	R column-wise	unchanged	$A^{-1}R$	$A^{-1}R$	$T_u^{-1}R$	$T_u^{-1}R$	$(T_u^{-1})^T R$	$(T_u^{-1})^T R$

If IOP is not equal to one of the values listed in Table 3, no action except error message IER = -1 takes place. In this case A and R remain unchanged. The same is true if mud (the number of upper codiagonals of A) is negative or if m (the number of rows of A) is less than 1+mud.

There is one way in which the factorization described above may fail to give a solution, which is easily seen from equation (6). If at any step the radicand of equation (6) is not positive, the procedure is bypassed, and the error message IER = -1 is given. The interpretation may be that matrix A is not positive-definite; however, failure is also possible due to roundoff errors.

Backsubstitutions may fail too, as is easily seen from equations (8) and (10). If any main diagonal element of T_u is equal to zero (this is only possible if T_u is given by input), the procedure is bypassed, and the error message IER = -1 is given.

A further test on possible loss of significance in the factorization part is performed by the subroutines. If at the kth factorization step the radicand of equation (6) is not greater than $|\mathrm{EPS}.a_{kk}|$, with the relative tolerance EPS given by input, the subroutines give the message IER = k-1.

For subroutine MCHB a relative tolerance between 10^{-6} and 10^{-7} is suggested, and for subroutine DMCHB, between 10^{-14} and 10^{-16}. The message IER = k-1, in case EPS = 10^{-D}, means that there are about D or more significant digits lost in the computation of the radicand in formula (6) at the kth factorization step. The interpretation may be that there is loss of significance in the factorized matrix from the kth row on, while the results of rows 1 up to k-1 may be better. Loss of significance in the lower part of the factorized matrix may affect the results of matrix divisions.

For reference see H. Rutishauser, "Algorithmus 1 - Lineares Gleichungs-system mit symmetrischer positiv-definiter Bandmatrix nach Cholesky-Computing," Archives for Electronic Computing, vol. 1, iss. 1 (1966), pp. 77-78.

```
C                                                                       MCHB  10
C ......................................................................MCHB  20
C                                                                       MCHB  30
C     SUBROUTINE MCHB                                                   MCHB  40
C                                                                       MCHB  50
C     PURPOSE                                                           MCHB  60
C        FOR A GIVEN POSITIVE-DEFINITE M BY M MATRIX A WITH SYMMETRICMCHB  70
C        BAND STRUCTURE AND - IF NECESSARY - A GIVEN GENERAL M BY N  MCHB  80
C        MATRIX R, THE FOLLOWING CALCULATIONS (DEPENDENT ON THE      MCHB  90
C        VALUE OF THE DECISION PARAMETER IOP) ARE PERFORMED          MCHB 100
C        (1) MATRIX A IS FACTORIZED (IF IOP IS NOT NEGATIVE), THAT   MCHB 110
C            MEANS BAND MATRIX TU WITH UPPER CODIAGONALS ONLY IS     MCHB 120
C            GENERATED ON THE LOCATIONS OF A SUCH THAT               MCHB 130
C            TRANSPOSE(TU)*TU=A.                                     MCHB 140
C        (2) MATRIX R IS MULTIPLIED ON THE LEFT BY INVERSE(TU)       MCHB 150
C            AND/OR INVERSE(TRANSPOSE(TU)) AND THE RESULT IS STORED  MCHB 160
C            IN THE LOCATIONS OF R.                                  MCHB 170
C        THIS SUBROUTINE ESPECIALLY CAN BE USED TO SOLVE THE SYSTEM  MCHB 180
C        OF SIMULTANEOUS LINEAR EQUATIONS A*X=R WITH POSITIVE-       MCHB 190
C        DEFINITE COEFFICIENT MATRIX A OF SYMMETRIC BAND STRUCTURE.  MCHB 200
C                                                                    MCHB 210
C     USAGE                                                          MCHB 220
C        CALL MCHB (R,A,M,N,MUD,IOP,EPS,IER)                         MCHB 230
C                                                                    MCHB 240
C     DESCRIPTION OF PARAMETERS                                      MCHB 250
C        R      - INPUT IN CASES IOP=-3,-2,-1,1,2,3  M BY N RIGHT   MCHB 260
C                       HAND SIDE MATRIX,                            MCHB 270
C                       IN CASE IOP=0  IRRELEVANT.                   MCHB 280
C                 OUTPUT IN CASES IOP=1,-1  INVERSE(A)*R,            MCHB 290
C                        IN CASES IOP=2,-2  INVERSE(TU)*R,           MCHB 300
C                        IN CASES IOP=3,-3  INVERSE(TRANSPOSE(TU))*R,MCHB 310
C                        IN CASE  IOP=0     UNCHANGED.               MCHB 320
C        A      - INPUT IN CASES IOP=0,1,2,3 M BY M POSITIVE-DEFINITEMCHB 330
C                       COEFFICIENT MATRIX OF SYMMETRIC BAND STRUC-  MCHB 340
C                       TURE STORED IN COMPRESSED FORM (SEE REMARKS),MCHB 350
C                       IN CASES IOP=-1,-2,-3  M BY M BAND MATRIX TU MCHB 360
C                       WITH UPPER CODIAGONALS ONLY, STORED IN       MCHB 370
C                       COMPRESSED FORM (SEE REMARKS).               MCHB 380
C                 OUTPUT IN ALL CASES  BAND MATRIX TU WITH UPPER     MCHB 390
C                       CODIAGONALS ONLY, STORED IN COMPRESSED FORM MCHB 400
C                       (THAT MEANS UNCHANGED IF IOP=-1,-2,-3).      MCHB 410
C        M      - INPUT VALUE SPECIFYING THE NUMBER OF ROWS AND     MCHB 420
C                 COLUMNS OF A AND THE NUMBER OF ROWS OF R.          MCHB 430
C        N      - INPUT VALUE SPECIFYING THE NUMBER OF COLUMNS OF R MCHB 440
C                 (IRRELEVANT IN CASE IOP=0).                        MCHB 450
C        MUD    - INPUT VALUE SPECIFYING THE NUMBER OF UPPER        MCHB 460
C                 CODIAGONALS OF A.                                  MCHB 470
C        IOP    - ONE OF THE VALUES -3,-2,-1,0,1,2,3 GIVEN AS INPUT MCHB 480
C                 AND USED AS DECISION PARAMETER.                    MCHB 490
C        EPS    - INPUT VALUE USED AS RELATIVE TOLERANCE FOR TEST ON MCHB 500
C                 LOSS OF SIGNIFICANT DIGITS.                        MCHB 510
C        IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS        MCHB 520
C                 IER=0  - NO ERROR,                                 MCHB 530
C                 IER=-1 - NO RESULT BECAUSE OF WRONG INPUT          MCHB 540
C                          PARAMETERS M,MUD,IOP (SEE REMARKS),       MCHB 550
C                          OR BECAUSE OF A NONPOSITIVE RADICAND AT   MCHB 560
C                          SOME FACTORIZATION STEP,                  MCHB 570
C                          OR BECAUSE OF A ZERO DIAGONAL ELEMENT     MCHB 580
C                          AT SOME DIVISION STEP.                    MCHB 590
C                 IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-  MCHB 600
C                          CANCE INDICATED AT FACTORIZATION STEP K+1MCHB 610
C                          WHERE RADICAND WAS NO LONGER GREATER      MCHB 620
C                          THAN EPS*A(K+1,K+1).                      MCHB 630
C                                                                    MCHB 640
C     REMARKS                                                        MCHB 650
C        UPPER PART OF SYMMETRIC BAND MATRIX A CONSISTING OF MAIN    MCHB 660
C        DIAGONAL AND MUD UPPER CODIAGONALS (RESP. BAND MATRIX TU    MCHB 670
C        CONSISTING OF MAIN DIAGONAL AND MUD UPPER CODIAGONALS)      MCHB 680
C        IS ASSUMED TO BE STORED IN COMPRESSED FORM, I.E. ROWWISE    MCHB 690
C        IN TOTALLY NEEDED M*MUD*(2M-MUD-1)/2 SUCCESSIVE STORAGE     MCHB 700
C        LOCATIONS. ON RETURN UPPER BAND FACTOR TU (ON THE LOCATIONS MCHB 710
C        OF A) IS STORED IN THE SAME WAY.                            MCHB 720
C        RIGHT HAND SIDE MATRIX R IS ASSUMED TO BE STORED COLUMNWISE MCHB 730
C        IN N*M SUCCESSIVE STORAGE LOCATIONS. ON RETURN RESULT MATRIXMCHB 740
C        INVERSE(A)*R OR INVERSE(TU)*R OR INVERSE(TRANSPOSE(TU))*R  MCHB 750
C        IS STORED COLUMNWISE (ON THE LOCATIONS OF R.                MCHB 760
C        INPUT PARAMETERS M, MUD, IOP SHOULD SATISFY THE FOLLOWING   MCHB 770
C        RESTRICTIONS        MUD NOT LESS THAN ZERO,                 MCHB 780
C                            1+MUD NOT GREATER THAN M,               MCHB 790
C                            ABS(IOP) NOT GREATER THAN 3.            MCHB 800
C        NO ACTION BESIDES ERROR MESSAGE IER=-1 TAKES PLACE IF THESE MCHB 810
C        RESTRICTIONS ARE NOT SATISFIED.                             MCHB 820
C        THE PROCEDURE GIVES RESULTS IF THE RESTRICTIONS ON INPUT    MCHB 830
C        PARAMETERS ARE SATISFIED, IF RADICANDS AT ALL FACTORIZATION MCHB 840
C        STEPS ARE POSITIVE AND/OR IF ALL DIAGONAL ELEMENTS OF       MCHB 850
C        UPPER BAND FACTOR TU ARE NONZERO.                           MCHB 860
C                                                                    MCHB 870
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MCHB 880
C        NONE                                                        MCHB 890
C                                                                    MCHB 900
C     METHOD                                                         MCHB 910
C        FACTORIZATION IS DONE USING CHOLESKY-S SQUARE-ROOT METHOD,  MCHB 920
C        WHICH GENERATES THE UPPER BAND MATRIX TU SUCH THAT          MCHB 930
C        TRANSPOSE(TU)*TU=A. TU IS RETURNED AS RESULT ON THE         MCHB 940
C        LOCATIONS OF A. FURTHER, DEPENDENT ON THE ACTUAL VALUE OF   MCHB 950
C        IOP, DIVISION OF R BY TRANSPOSE(TU) AND/OR TU IS PERFORMED  MCHB 960
C        AND THE RESULT IS RETURNED ON THE LOCATIONS OF R.           MCHB 970
C        FOR REFERENCE, SEE H. RUTISHAUSER, ALGORITHMUS 1 - LINEARES MCHB 980
C        GLEICHUNGSSYSTEM MIT SYMMETRISCHER POSITIV-DEFINITER        MCHB 990
C        BANDMATRIX NACH CHOLESKY - , COMPUTING (ARCHIVES FOR        MCHB1000
C        ELECTRONIC COMPUTING), VOL.1, ISS.1 (1966), PP.77-78.       MCHB1010
C                                                                    MCHB1020
C ......................................................................MCHB1030
C                                                                    MCHB1040
C     SUBROUTINE MCHB(R,A,M,N,MUD,IOP,EPS,IER)                       MCHB1050
C                                                                    MCHB1060
C                                                                    MCHB1070
C     DIMENSION R(1),A(1)                                            MCHB1080
C     DOUBLE PRECISION TOL,SUM,PIV                                   MCHB1090
C                                                                    MCHB1100
C        TEST ON WRONG INPUT PARAMETERS                              MCHB1110
C     IF(IABS(IOP)-3)1,1,43                                          MCHB1120
C   1 IF(MUD)43,2,2                                                  MCHB1130
C   2 MC=MUD+1                                                       MCHB1140
C     IF(M-MC)43,3,3                                                 MCHB1150
C   3 MR=M-MUD                                                       MCHB1160
C     IER=0                                                          MCHB1170
C                                                                    MCHB1180
C        MC IS THE MAXIMUM NUMBER OF ELEMENTS IN THE ROWS OF ARRAY A MCHB1190
C        MR IS THE INDEX OF THE LAST ROW IN ARRAY A WITH MC ELEMENTS MCHB1200
C                                                                    MCHB1210
C     ******************************************************************MCHB1220
C                                                                    MCHB1230
C        START FACTORIZATION OF MATRIX A                             MCHB1240
C     IF(IOP)24,4,4                                                  MCHB1250
C   4 IEND=0                                                         MCHB1260
C     LLOST=MUD                                                      MCHB1270
C     DO 23 K=1,M                                                    MCHB1280
C     IST=IEND+1                                                     MCHB1290
```

```
      IEND=IST+MUD                                                      MCHB1300
      J=K-MR                                                            MCHB1310
      IF(J)6,4,5                                                        MCHB1320
    5 IEND=IEND-J                                                       MCHB1330
    6 IF(J-1)8,8,7                                                      MCHB1340
    7 LLDST=LLDST-1                                                     MCHB1350
    8 LMAX=MUD                                                          MCHB1360
      J=MC-K                                                            MCHB1370
      IF(J)10,10,9                                                      MCHB1380
    9 LMAX=LMAX-J                                                       MCHB1390
   10 ID=0                                                              MCHB1400
      TOL=A(IST)*EPS                                                    MCHB1410
C                                                                       MCHB1420
C        START FACTORIZATION-LOOP OVER K-TH ROW                         MCHB1430
      DO 23 I=IST,IEND                                                  MCHB1440
      SUM=0.D0                                                          MCHB1450
      IF(LMAX)14,14,11                                                  MCHB1460
C                                                                       MCHB1470
C        PREPARE INNER LOOP                                             MCHB1480
   11 LL=IST                                                            MCHB1490
      LLD=LLDST                                                         MCHB1500
C                                                                       MCHB1510
C        START INNER LOOP                                               MCHB1520
      DO 13 L=1,LMAX                                                    MCHB1530
      LL=LL-LLD                                                         MCHB1540
      LLL=LL+ID                                                         MCHB1550
      SUM=SUM+A(LL)*A(LLL)                                              MCHB1560
      IF(LLD-MUD)12,13,13                                               MCHB1570
   12 LLD=LLD+1                                                         MCHB1580
   13 CONTINUE                                                          MCHB1590
C        END OF INNER LOOP                                              MCHB1600
C                                                                       MCHB1610
C        TRANSFORM ELEMENT A(I)                                         MCHB1620
   14 SUM=DBLE(A(I))-SUM                                                MCHB1630
      IF(I-IST)15,15,20                                                 MCHB1640
C                                                                       MCHB1650
C        A(I) IS DIAGONAL ELEMENT. ERROR TEST.                          MCHB1660
   15 IF(SUM)43,43,16                                                   MCHB1670
C                                                                       MCHB1680
C        TEST ON LOSS OF SIGNIFICANT DIGITS AND WARNING                 MCHB1690
   16 IF(SUM-TOL)17,17,19                                               MCHB1700
   17 IF(IEP)18,18,19                                                   MCHB1710
   18 IER=K-1                                                           MCHB1720
C                                                                       MCHB1730
C        COMPUTATION OF PIVOT ELEMENT                                   MCHB1740
   19 PIV=DSQRT(SUM)                                                    MCHB1750
      A(I)=PIV                                                          MCHB1760
      PIV=1.D0/PIV                                                      MCHB1770
      GO TO 21                                                          MCHB1780
C                                                                       MCHB1790
C        A(I) IS NOT DIAGONAL ELEMENT                                   MCHB1800
   20 A(I)=SUM*PIV                                                      MCHB1810
C                                                                       MCHB1820
C        UPDATE ID AND LMAX                                             MCHB1830
   21 ID=ID+1                                                           MCHB1840
      IF(ID-JJ)23,22,22                                                 MCHB1850
   22 LMAX=LMAX-1                                                       MCHB1860
   23 CONTINUE                                                          MCHB1870
C                                                                       MCHB1880
C        END OF FACTORIZATION-LOOP OVER K-TH ROW                        MCHB1890
C        END OF FACTORIZATION OF MATRIX A                               MCHB1900
C                                                                       MCHB1910
C     ******************************************************************MCHB1920
C                                                                       MCHB1930
C        PREPARE MATRIX DIVISIONS                                       MCHB1940
      IF(IOP)24,44,24                                                   MCHB1950
   24 ID=N*M                                                            MCHB1960
      IEND=IABS(IOP)-2                                                  MCHB1970
      IF(IEND)25,35,25                                                  MCHB1980
C                                                                       MCHB1990
C     ******************************************************************MCHB2000
C                                                                       MCHB2010
C        START DIVISION BY TRANSPOSE OF MATRIX TU (TU IS STORED IN      MCHB2020
C        LOCATIONS OF A)                                                MCHB2030
   25 IST=1                                                             MCHB2040
      LMAX=0                                                            MCHB2050
      J=-MR                                                             MCHB2060
      LLDST=MUD                                                         MCHB2070
      DO 34 K=1,N                                                       MCHB2080
      PIV=A(IST)                                                        MCHB2090
      IF(PIV)26,43,26                                                   MCHB2100
   26 PIV=1.D0/PIV                                                      MCHB2110
C                                                                       MCHB2120
C        START BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R           MCHB2130
      DO 30 I=K,ID,M                                                    MCHB2140
      SUM=0.D0                                                          MCHB2150
      IF(LMAX)30,30,27                                                  MCHB2160
C                                                                       MCHB2170
C        PREPARE INNER LOOP                                             MCHB2180
   27 LL=IST                                                            MCHB2190
      LLL=I                                                             MCHB2200
      LLD=LLDST                                                         MCHB2210
C                                                                       MCHB2220
C        START INNER LOOP                                               MCHB2230
      DO 29 L=1,LMAX                                                    MCHB2240
      LL=LL-LLD                                                         MCHB2250
      LLL=LLL-1                                                         MCHB2260
      SUM=SUM+A(LL)*R(LLL)                                              MCHB2270
      IF(LLD-MUD)28,29,29                                               MCHB2280
   28 LLD=LLD+1                                                         MCHB2290
   29 CONTINUE                                                          MCHB2300
C        END OF INNER LOOP                                              MCHB2310
C                                                                       MCHB2320
C        TRANSFORM ELEMENT R(I)                                         MCHB2330
   30 R(I)=PIV*(DBLE(R(I))-SUM)                                         MCHB2340
C        END OF BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R          MCHB2350
C                                                                       MCHB2360
C        UPDATE PARAMETERS LMAX, IST AND LLDST                          MCHB2370
      IF(MC-K)32,32,31                                                  MCHB2380
   31 LMAX=K                                                            MCHB2390
   32 IST=IST+MC                                                        MCHB2400
      J=J+1                                                             MCHB2410
      IF(J)34,34,33                                                     MCHB2420
   33 IST=IST-J                                                         MCHB2430
      LLDST=LLDST-1                                                     MCHB2440
   34 CONTINUE                                                          MCHB2450
C                                                                       MCHB2460
C        END OF DIVISION BY TRANSPOSE OF MATRIX TU                      MCHB2470
C                                                                       MCHB2480
C     ******************************************************************MCHB2490
C                                                                       MCHB2500
C        START DIVISION BY MATRIX TU (TU IS STORED ON LOCATIONS OF A)   MCHB2510
      IF(IEND)35,35,44                                                  MCHB2520
   35 IST=M*(MUD*(M*M-MC))/2+1                                          MCHB2530
      LMAX=0                                                            MCHB2540
      K=M                                                               MCHB2550
   36 IEND=IST-1                                                        MCHB2560
      IST=IEND-LMAX                                                     MCHB2570
      PIV=A(IST)                                                        MCHB2580
      IF(PIV)37,43,37                                                   MCHB2590
   37 PIV=1.D0/PIV                                                      MCHB2600
      L=IST+1                                                           MCHB2610
C                                                                       MCHB2620
C        START BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R           MCHB2630
      DO 40 I=K,ID,M                                                    MCHB2640
      SUM=0.D0                                                          MCHB2650
      IF(LMAX)40,40,38                                                  MCHB2660
   38 LLL=I                                                             MCHB2670
C                                                                       MCHB2680
C        START INNER LOOP                                               MCHB2690
      DO 39 LL=L,IEND                                                   MCHB2700
      LLL=LLL+1                                                         MCHB2710
   39 SUM=SUM+A(LL)*R(LLL)                                              MCHB2720
C        END OF INNER LOOP                                              MCHB2730
C                                                                       MCHB2740
C        TRANSFORM ELEMENT R(I)                                         MCHB2750
   40 R(I)=PIV*(DBLE(R(I))-SUM)                                         MCHB2760
C        END OF BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R          MCHB2770
C                                                                       MCHB2780
C        UPDATE PARAMETERS LMAX AND K                                   MCHB2790
      IF(K-MR)42,42,41                                                  MCHB2800
   41 LMAX=LMAX+1                                                       MCHB2810
   42 K=K-1                                                             MCHB2820
      IF(K)44,44,36                                                     MCHB2830
C                                                                       MCHB2840
C        END OF DIVISION BY MATRIX TU                                   MCHB2850
C                                                                       MCHB2860
C     ******************************************************************MCHB2870
C                                                                       MCHB2880
C        ERROR EXIT IN CASE OF WRONG INPUT PARAMETERS OR PIVOT ELEMENT  MCHB2890
C        LESS THAN OR EQUAL TO ZERO                                     MCHB2900
   43 IER=-1                                                            MCHB2910
   44 RETURN                                                            MCHB2920
      END                                                               MCHB2930
                                                                        MCHB2940
```

```
C                                                                       DMCH  10
C     ******************************************************************DMCH  20
C                                                                       DMCH  30
C        SUBROUTINE DMCHB                                               DMCH  40
C                                                                       DMCH  50
C        PURPOSE                                                        DMCH  60
C           FOR A GIVEN POSITIVE-DEFINITE M BY M MATRIX A WITH SYMMETRICDMCH  70
C           BAND STRUCTURE AND - IF NECESSARY - A GIVEN GENERAL M BY N  DMCH  80
C           MATRIX R, THE FOLLOWING CALCULATIONS (DEPENDENT ON THE      DMCH  90
C           VALUE OF THE DECISION PARAMETER IOP) ARE PERFORMED          DMCH 100
C           (1) MATRIX A IS FACTORIZED (IF IOP IS NOT NEGATIVE), THAT   DMCH 110
C               MEANS BAND MATRIX TU WITH UPPER CODIAGONALS ONLY IS     DMCH 120
C               GENERATED ON THE LOCATIONS OF A SUCH THAT               DMCH 130
C               TRANSPOSE(TU)*TU=A.                                     DMCH 140
C           (2) MATRIX R IS MULTIPLIED ON THE LEFT BY INVERSE(TU)       DMCH 150
C               AND/OR INVERSE(TRANSPOSE(TU)) AND THE RESULT IS STORED  DMCH 160
C               IN THE LOCATIONS OF R.                                  DMCH 170
C           THIS SUBROUTINE ESPECIALLY CAN BE USED TO SOLVE THE SYSTEM  DMCH 180
C           OF SIMULTANEOUS LINEAR EQUATIONS A*X=R WITH POSITIVE-       DMCH 190
C           DEFINITE COEFFICIENT MATRIX A OF SYMMETRIC BAND STRUCTURE.  DMCH 200
C                                                                       DMCH 210
C        USAGE                                                          DMCH 220
C           CALL DMCHB (R,A,M,N,MUD,IOP,EPS,IER)                        DMCH 230
C                                                                       DMCH 240
C        DESCRIPTION OF PARAMETERS                                      DMCH 250
C           R      - INPUT IN CASES IOP=-3,-2,-1,1,2,3  DOUBLE PRECISIONDMCH 260
C                    M BY N RIGHT HAND SIDE MATRIX.                     DMCH 270
C                    IN CASE IOP=0   IRRELEVANT.                        DMCH 280
C                 OUTPUT IN CASES IOP=1,-1  INVERSE(A)*R,               DMCH 290
C                        IN CASES IOP=2,-2  INVERSE(TU)*R,              DMCH 300
C                        IN CASES IOP=3,-3  INVERSE(TRANSPOSE(TU))*R,DMCH 310
C                        IN CASE  IOP=0     UNCHANGED.                  DMCH 320
C           A      - INPUT IN CASES IOP=0,1,2,3  DOUBLE PRECISION M BY MDMCH 330
C                    POSITIVE-DEFINITE COEFFICIENT MATRIX OF            DMCH 340
C                    SYMMETRIC BAND STRUCTURE STORED IN                 DMCH 350
C                    COMPRESSED FORM (SEE REMARKS).                     DMCH 360
C                    IN CASES IOP=-1,-2,-3 DOUBLE PRECISION M BY MDMCH 370
C                    BAND MATRIX TU WITH UPPER CODIAGONALS ONLY,        DMCH 380
C                    STORED IN COMPRESSED FORM (SEE REMARKS).           DMCH 390
C                 OUTPUT IN ALL CASES  BAND MATRIX TU WITH UPPER        DMCH 400
C                        CODIAGONALS ONLY, STORED IN COMPRESSED FORM DMCH 410
C                        (THAT MEANS UNCHANGED IF IOP=-1,-2,-3).        DMCH 420
C           M      - INPUT VALUE SPECIFYING THE NUMBER OF ROWS AND     DMCH 430
C                    COLUMNS OF A AND THE NUMBER OF ROWS OF R.          DMCH 440
C           N      - INPUT VALUE SPECIFYING THE NUMBER OF COLUMNS OF R DMCH 450
C                    (IRRELEVANT IF CASE IOP=0).                        DMCH 460
C           MUD    - INPUT VALUE SPECIFYING THE NUMBER OF UPPER        DMCH 470
C                    CODIAGONALS OF A.                                  DMCH 480
C           IOP    - ONE OF THE VALUES -3,-2,-1,0,1,2,3 GIVEN AS INPUT DMCH 490
C                    AND USED AS DECISION PARAMETER.                    DMCH 500
C           EPS    - SINGLE PRECISION INPUT VALUE USED AS RELATIVE     DMCH 510
C                    TOLERANCE FOR TEST ON LOSS OF SIGNIFICANT DIGITS.  DMCH 520
C           IER    - RESULTING ERROR PARAMETER CODED AS FOLLOWS        DMCH 530
C                    IER=0  - NO ERROR.                                 DMCH 540
C                    IER=-1 - NO RESULT BECAUSE OF WRONG INPUT          DMCH 550
C                             PARAMETERS M,MUD,IOP (SEE REMARKS).       DMCH 560
C                             OR BECAUSE OF A NONPOSITIVE RADICAND AT   DMCH 570
C                             SOME FACTORIZATION STEP,                  DMCH 580
C                             OR BECAUSE OF A ZERO DIAGONAL ELEMENT     DMCH 590
C                             AT SOME DIVISION STEP.                    DMCH 600
C                    IER=K  - WARNING DUE TO POSSIBLE LOSS OF SIGNIFI-  DMCH 610
C                             CANCE INDICATED AT FACTORIZATION STEP K+1DMCH 620
C                             WHERE RADICAND WAS NO LONGER GREATER      DMCH 630
C                             THAN EPS*A(K+1,K+1).                      DMCH 640
C                                                                       DMCH 650
C        REMARKS                                                        DMCH 660
C           UPPER PART OF SYMMETRIC BAND MATRIX A CONSISTING OF MAIN    DMCH 670
C           DIAGONAL AND MUD UPPER CODIAGONALS (RESP. BAND MATRIX TU    DMCH 680
C           CONSISTING OF MAIN DIAGONAL AND MUD UPPER CODIAGONALS)      DMCH 690
C           IS ASSUMED TO BE STORED IN COMPRESSED FORM, I.E. ROWWISE    DMCH 700
C           IN TOTALLY NEEDED M*MUD+(2M-MUD-1)/2 SUCCESSIVE STORAGE     DMCH 710
C           LOCATIONS. ON RETURN UPPER BAND FACTOR TU (ON THE LOCATIONS DMCH 720
C           OF A) IS STORED IN THE SAME WAY.                            DMCH 730
C           RIGHT HAND SIDE MATRIX R IS ASSUMED TO BE STORED COLUMNWISE DMCH 740
C           IN NM SUCCESSIVE STORAGE LOCATIONS. ON RETURN RESULT MATRIXDMCH 750
C           INVERSE(A)*R OR INVERSE(TU)*R OR INVERSE(TRANSPOSE(TU))*R.  DMCH 760
C           IS STORED COLUMNWISE TOO ON THE LOCATIONS OF R.             DMCH 770
C           INPUT PARAMETERS M, MUD, IOP SHOULD SATISFY THE FOLLOWING   DMCH 780
C           RESTRICTIONS      MUD NOT LESS THAN ZERO,                   DMCH 790
C                             1+MUD NOT GREATER THAN M,                 DMCH 800
C                             ABS(IOP) NOT GREATER THAN 3.              DMCH 810
C           NO ACTION BESIDES ERROR MESSAGE IER=-1 TAKES PLACE IF THESE DMCH 820
C           RESTRICTIONS ARE NOT SATISFIED.                             DMCH 830
C           THE PROCEDURE GIVES RESULTS IF THE RESTRICTIONS ON INPUT    DMCH 840
C           PARAMETERS ARE SATISFIED. IF RADICANDS AT ALL FACTORIZATION DMCH 850
C           STEPS ARE POSITIVE AND/OR IF ALL DIAGONAL ELEMENTS OF       DMCH 860
C           UPPER BAND FACTOR TU ARE NONZERO.                           DMCH 870
```

```
C                                                                       DMCH 880
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DMCH 890
C           NONE                                                        DMCH 900
C                                                                       DMCH 910
C        METHOD                                                         DMCH 920
C           FACTORIZATION IS DONE USING CHOLESKY-S SQUARE-ROOT METHOD,  DMCH 930
C           WHICH GENERATES THE UPPER BAND MATRIX TU SUCH THAT          DMCH 940
C           TRANSPOSE(TU)*TU=A. TU IS RETURNED AS RESULT ON THE         DMCH 950
C           LOCATIONS OF A. FURTHER, DEPENDENT ON THE ACTUAL VALUE OF   DMCH 960
C           IOP, DIVISION OF R BY TRANSPOSE(TU) AND/OR TU IS PERFORMED  DMCH 970
C           AND THE RESULT IS RETURNED ON THE LOCATIONS OF R.           DMCH 980
C           FOR REFERENCE, SEE H. RUTISHAUSER, ALGORITHMUS 1 - LINEARES DMCH 990
C           GLEICHUNGSSYSTEM MIT SYMMETRISCHER POSITIV-DEFINITER        DMCH1000
C           BANDMATRIX NACH CHOLESKY - , COMPUTING (ARCHIVES FOR        DMCH1010
C           ELECTRONIC COMPUTING), VOL.1, ISS.1 (1966), PP.77-78.       DMCH1020
C                                                                       DMCH1030
C     ******************************************************************DMCH1040
C                                                                       DMCH1050
      SUBROUTINE DMCH3(R,A,M,N,MUD,IOP,EPS,IER)                         DMCH1060
C                                                                       DMCH1070
C                                                                       DMCH1080
      DIMENSION R(1),A(1)                                               DMCH1090
      DOUBLE PRECISION TOL,SUM,PIV,R,A                                  DMCH1100
C                                                                       DMCH1110
C        TEST ON WRONG INPUT PARAMETERS                                 DMCH1120
      IF(IABS(IOP)-3)1,1,43                                             DMCH1130
    1 IF(MUD)43,2,2                                                     DMCH1140
    2 MC=MUD+1                                                          DMCH1150
      IF(M-MC)43,3,3                                                    DMCH1160
    3 MR=M-MUD                                                          DMCH1170
      IER=0                                                             DMCH1180
C                                                                       DMCH1190
C        MC IS THE MAXIMUM NUMBER OF ELEMENTS IN THE ROWS OF ARRAY A    DMCH1200
C        MR IS THE INDEX OF THE LAST ROW IN ARRAY A WITH MC ELEMENTS    DMCH1210
C                                                                       DMCH1220
C     ******************************************************************DMCH1230
C                                                                       DMCH1240
C        START FACTORIZATION OF MATRIX A                                DMCH1250
      IF(IOP)24,4,4                                                     DMCH1260
    4 IEND=0                                                            DMCH1270
      LLDST=MUD                                                         DMCH1280
      DO 23 K=1,M                                                       DMCH1290
      IST=IEND+1                                                        DMCH1300
      IEND=IST+MUD                                                      DMCH1310
      J=K-MR                                                            DMCH1320
      IF(J)6,6,5                                                        DMCH1330
    5 IEND=IEND-J                                                       DMCH1340
    6 IF(J-1)8,8,7                                                      DMCH1350
    7 LLDST=LLDST-1                                                     DMCH1360
    8 LMAX=MUD                                                          DMCH1370
      J=MC-K                                                            DMCH1380
      IF(J)10,10,9                                                      DMCH1390
    9 LMAX=LMAX-J                                                       DMCH1400
   10 ID=0                                                              DMCH1410
      TOL=A(IST)*EPS                                                    DMCH1420
C                                                                       DMCH1430
C        START FACTORIZATION-LOOP OVER K-TH ROW                         DMCH1440
      DO 23 I=IST,IEND                                                  DMCH1450
      SUM=0.D0                                                          DMCH1460
      IF(LMAX)14,14,11                                                  DMCH1470
C                                                                       DMCH1480
C        PREPARE INNER LOOP                                             DMCH1490
   11 LL=IST                                                            DMCH1500
      LLD=LLDST                                                         DMCH1510
C                                                                       DMCH1520
C        START INNER LOOP                                               DMCH1530
      DO 13 L=1,LMAX                                                    DMCH1540
      LL=LL-LLD                                                         DMCH1550
      LLL=LL+ID                                                         DMCH1560
      SUM=SUM+A(LL)*A(LLL)                                              DMCH1570
      IF(LLD-MUD)12,13,13                                               DMCH1580
   12 LLD=LLD+1                                                         DMCH1590
   13 CONTINUE                                                          DMCH1600
C        END OF INNER LOOP                                              DMCH1610
C                                                                       DMCH1620
C        TRANSFORM ELEMENT A(I)                                         DMCH1630
   14 SUM=A(I)-SUM                                                      DMCH1640
      IF(I-IST)15,15,20                                                 DMCH1650
C                                                                       DMCH1660
C        A(I) IS DIAGONAL ELEMENT. ERROR TEST.                          DMCH1670
   15 IF(SUM)43,43,16                                                   DMCH1680
C                                                                       DMCH1690
C        TEST ON LOSS OF SIGNIFICANT DIGITS AND WARNING                 DMCH1700
   16 IF(SUM-TOL)17,17,19                                               DMCH1710
   17 IF(IER)18,18,19                                                   DMCH1720
   18 IER=K-1                                                           DMCH1730
C                                                                       DMCH1740
C        COMPUTATION OF PIVOT ELEMENT                                   DMCH1750
   19 PIV=DSQRT(SUM)                                                    DMCH1760
      A(I)=PIV                                                          DMCH1770
      PIV=1.D0/PIV                                                      DMCH1780
      GO TO 21                                                          DMCH1790
C                                                                       DMCH1800
C        A(I) IS NOT DIAGONAL ELEMENT                                   DMCH1810
   20 A(I)=SUM*PIV                                                      DMCH1820
C                                                                       DMCH1830
C        UPDATE ID AND LMAX                                             DMCH1840
   21 ID=ID+1                                                           DMCH1850
      IF(ID-J)23,23,22                                                  DMCH1860
   22 LMAX=LMAX-1                                                       DMCH1870
   23 CONTINUE                                                          DMCH1880
C                                                                       DMCH1890
C        END OF FACTORIZATION-LOOP OVER K-TH ROW                        DMCH1900
C        END OF FACTORIZATION OF MATRIX A                               DMCH1910
C                                                                       DMCH1920
C     ******************************************************************DMCH1930
C                                                                       DMCH1940
C        PREPARE MATRIX DIVISIONS                                       DMCH1950
      IF(IOP)24,44,24                                                   DMCH1960
   24 ID=N*M                                                            DMCH1970
      IEND=IABS(IOP)-2                                                  DMCH1980
      IF(IEND)25,35,25                                                  DMCH1990
C                                                                       DMCH2000
C     ******************************************************************DMCH2010
C                                                                       DMCH2020
C        START DIVISION BY TRANSPOSE OF MATRIX TU (TU IS STORED IN      DMCH2030
C        LOCATIONS OF A)                                                DMCH2040
   25 IST=1                                                             DMCH2050
      LMAX=0                                                            DMCH2060
      J=-MR                                                             DMCH2070
      LLDST=MUD                                                         DMCH2080
      DO 34 K=1,M                                                       DMCH2090
      PIV=A(IST)                                                        DMCH2100
      IF(PIV)26,43,26                                                   DMCH2110
   26 PIV=1.D0/PIV                                                      DMCH2120
C                                                                       DMCH2130
C        STA-T BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R           DMCH2140
      DO 30 I=K,ID,M                                                    DMCH2150
      SUM=0.D0                                                          DMCH2160
      IF(LMAX)30,30,27                                                  DMCH2170
C                                                                       DMCH2180
C        PREPARE INNER LOOP                                             DMCH2190
   27 LL=IST                                                            DMCH2200
      LLL=I                                                             DMCH2210
      LLD=LLDST                                                         DMCH2220
C                                                                       DMCH2230
C        START INNER LOOP                                               DMCH2240
      DO 29 L=1,LMAX                                                    DMCH2250
      LL=LL-LLD                                                         DMCH2260
      LLL=LLL-1                                                         DMCH2270
      SUM=SUM*A(LL)*R(LLL)                                              DMCH2280
      IF(LLD-MUD)28,29,29                                               DMCH2290
   28 LLD=LLD+1                                                         DMCH2300
   29 CONTINUE                                                          DMCH2310
C        END OF INNER LOOP                                              DMCH2320
C                                                                       DMCH2330
C        TRANSFORM ELEMENT R(I)                                         DMCH2340
   30 R(I)=PIV*(R(I)-SUM)                                               DMCH2350
C        END OF BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R          DMCH2360
C                                                                       DMCH2370
C        UPDATE PARAMETERS LMAX, IST AND LLDST                          DMCH2380
      IF(MC-K)32,32,31                                                  DMCH2390
   31 LMAX=K                                                            DMCH2400
   32 IST=IST+MC                                                        DMCH2410
      J=J+1                                                             DMCH2420
      IF(J)34,34,33                                                     DMCH2430
   33 IST=IST-J                                                         DMCH2440
      LLDST=LLDST-1                                                     DMCH2450
   34 CONTINUE                                                          DMCH2460
C                                                                       DMCH2470
C        END OF DIVISION BY TRANSPOSE OF MATRIX TU                      DMCH2480
C                                                                       DMCH2490
C     ******************************************************************DMCH2500
C                                                                       DMCH2510
C        START DIVISION BY MATRIX TU (TU IS STORED ON LOCATIONS OF A)   DMCH2520
      IF(IEND)35,35,44                                                  DMCH2530
   35 IST=M+(MUD*(M+M-MC))/2+1                                          DMCH2540
      LMAX=0                                                            DMCH2550
      K=M                                                               DMCH2560
   36 IEND=IST-1                                                        DMCH2570
      IST=IEND-LMAX                                                     DMCH2580
      PIV=A(IST)                                                        DMCH2590
      IF(PIV)37,43,37                                                   DMCH2600
   37 PIV=1.D0/PIV                                                      DMCH2610
      L=IST+1                                                           DMCH2620
C                                                                       DMCH2630
C        START BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R           DMCH2640
      DO 40 I=K,ID,M                                                    DMCH2650
      SUM=0.D0                                                          DMCH2660
      IF(LMAX)40,40,38                                                  DMCH2670
   38 LLL=I                                                             DMCH2680
C                                                                       DMCH2690
C        START INNER LOOP                                               DMCH2700
      DO 39 LL=1,IEND                                                   DMCH2710
      LLL=LLL+1                                                         DMCH2720
   39 SUM=SUM+A(LL)*R(LLL)                                              DMCH2730
C        END OF INNER LOOP                                              DMCH2740
C                                                                       DMCH2750
C        TRANSFORM ELEMENT R(I)                                         DMCH2760
   40 R(I)=PIV*(R(I)-SUM)                                               DMCH2770
C        END OF BACKSUBSTITUTION-LOOP FOR K-TH ROW OF MATRIX R          DMCH2780
C                                                                       DMCH2790
C        UPDATE PARAMETERS LMAX AND K                                   DMCH2800
      IF(K-MR)42,42,41                                                  DMCH2810
   41 LMAX=LMAX+1                                                       DMCH2820
   42 K=K-1                                                             DMCH2830
      IF(K)44,44,36                                                     DMCH2840
C                                                                       DMCH2850
C        END OF DIVISION BY MATRIX TU                                   DMCH2860
C                                                                       DMCH2870
C     ******************************************************************DMCH2880
C                                                                       DMCH2890
C        ERROR EXIT IN CASE OF WRONG INPUT PARAMETERS OR PIVOT ELEMENT  DMCH2900
C        LESS THAN OR EQUAL TO ZERO                                     DMCH2910
   43 IER=-1                                                            DMCH2920
   44 RETURN                                                            DMCH2930
      END                                                               DMCH2940
```

Subroutine MFSS and DMFSS

Given a symmetric positive semidefinite matrix A, MFSS will:

(1) Determine the rank and linearly independent rows (and columns)

(2) Compute a triangular factorization of a symmetric submatrix of maximal rank

(3) Express nonbasic rows (and columns) in terms of basic ones and express nonbasic variables in terms of free ones.

1. Mathematical background

The rank is determined using the method of Cholesky (square root method) with pivoting along the main diagonal. Pivoting on diagonal elements means that the same permutation is applied to the rows and columns of A, thereby preserving symmetry. The interchange information is recorded in the auxiliary vector TRAC. The notation A^m is used for the interchanged matrix determined during the mth step. The given matrix A is assumed to be stored column-wise in compressed form, that is, upper triangular part only.

$$A = \begin{bmatrix} a_{11} & a_{12} \cdots & a_{1n} \\ & a_{22} \cdots & a_{2n} \\ & & a_{nn} \end{bmatrix}$$

To allow for easy pivoting, the diagonal terms of A are moved into vector TRAC.

The following two facts will be used repeatedly:

(I) If $S=(s_{ik})$ is symmetric with $s_{11} > 0$, then S admits the unique decomposition:

$$S = \begin{bmatrix} t & 0 \\ V^T & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} t & V \\ 0 & I \end{bmatrix} = \tag{1}$$

$$\begin{bmatrix} s_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix}$$

where $t = \sqrt{s_{11}}$, $V = (1/t)S_{12}$ and $D = S_{22} - V^T V$.

(II) If S is symmetric positive semidefinite:

$$s_{jj} \geq 0 \text{ for all } j \text{ and } |s_{ik}| \leq \max_j(s_{jj}) \text{ for all } i,k \tag{2}$$

a. First step

Let $a_{jj} = \max_i(a_{ii})$. If $a_{jj} = 0$, formula (2) implies $A = 0$, and further calculations are bypassed. Assuming $a_{jj} > 0$, rows 1 and $k_1 = j$ and columns 1 and k_1 of A are interchanged, obtaining A^1. TRAC(k_1) is replaced by TRAC(1), and k_1 is stored in TRAC(1). Decomposition of A^1 gives:

$$A^1 = \begin{bmatrix} A^1_{11} & A^1_{12} \\ A^1_{21} & A^1_{22} \end{bmatrix} = \begin{bmatrix} t^1_{11} & 0 \\ (V^1)^T & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & D^1 \end{bmatrix} \begin{bmatrix} t^1_{11} & V^1 \\ 0 & I \end{bmatrix}$$

where $t^1_{11} = \sqrt{a^1_{11}}$, $V^1 = (1/t^1_{11})A^1_{12}$ and $D^1 = A^1_{22} - (V^1)^T V^1$. D^1 is not calculated explicitly, only its diagonal terms are computed and stored in locations TRAC(2) through TRAC(n).

The first row of A^1 is replaced by t^1_{11}, V^1. The compressed storage scheme now looks like this:

$$\begin{matrix} t^1_{11} & V^1 \\ & A^1_{22} \end{matrix}$$

b. General step

Suppose that after m steps ($1 \leq m < n$) we have determined a transposition vector ($k_1, k_2, \ldots, k_m$), an m by m nonsingular upper triangular matrix:

$$T^m = \begin{bmatrix} t^1_{11} & t^2_{12} \cdots & t^m_{1m} \\ & t^2_{22} \cdots & t^m_{2m} \\ & & t^m_{mm} \end{bmatrix}$$

an m by (n-m) matrix:

$$V^m = \begin{bmatrix} V^m_1 \\ V^m_2 \\ \cdot \\ \cdot \\ \cdot \\ V^m_m \end{bmatrix}$$

and an (n-m) by (n-m) matrix D^m such that:

$$A^m = \begin{bmatrix} A^m_{11} & A^m_{12} \\ A^m_{21} & A^m_{22} \end{bmatrix} = \begin{bmatrix} (T^m)^T & 0 \\ (V^m)^T & I \end{bmatrix} \cdot \begin{bmatrix} I & 0 \\ 0 & D^m \end{bmatrix} \cdot \begin{bmatrix} T^m & V^m \\ 0 & I \end{bmatrix} \tag{3}$$

where $A^m = P^m A (P^m)^T$, and P^m is the permutation matrix associated with the transposition vector $(k_1, \ldots, k_m)$. Again, D^m is not known explicitly, but is computable from:

$$D^m = A^m_{22} - (V^m)^T V^m \tag{4}$$

T^m, V^m and A^m_{22} are assumed to be stored in the following way:

$$\begin{matrix} T^m & V^m \\ & A^m_{22} \end{matrix}$$

If $A^m_{22} = 0$, it follows from (4) and the fact that D^m is positive semidefinite that $D^m = 0$. Then (3) is the desired factorization, and the process terminates. If $A^m_{22} \neq 0$, there is at least one positive diagonal term a^m_{kk}, and we may compute:

$$t = d^m_{jj}/a^m_{jj} = \max_k \left\{ d^m_{kk}/a^m_{kk} \;\middle|\; a^m_{kk} > 0 \right\}$$

From (4) and (2), $\max_{i,k} \left| d^m_{ik} \right| \leq t \max_{i,k} \left| a^m_{ik} \right|$.

If $t \leq \epsilon$, the norm of D^m is small compared to the norm of A^m_{22}, and we interpret this as $D^m = 0$. Assuming $t > \epsilon$, rows m+1 and $k_{m+1} = m+j$ and columns m+1 and k_{m+1} are interchanged in the compressed storage areas, obtaining

$$\begin{matrix} T^m & V'^m \\ & A'^m_{22} \end{matrix}$$

TRAC(k_{m+1}) is replaced by TRAC(m+1), and k_{m+1} is stored in TRAC(m+1). As a result of these interchanges:

$$QA^mQ = \begin{bmatrix} (T^m)^T & 0 \\ (V'^m)^T & I \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & D'^m \end{bmatrix} \begin{bmatrix} T^m & V'^m \\ 0 & I \end{bmatrix} \tag{5}$$

where $D'^m = A'^m_{22} - (V'^m)^T V'^m$, and Q is the permutation matrix associated with the transposition $(m+1, k_{m+1})$. Now decompose D'^m:

$$D'^m = \begin{bmatrix} t^{m+1}_{m+1,m+1} & 0 \\ (V^{m+1}_{m+1})^T & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & D^{m+1} \end{bmatrix}$$

$$\begin{bmatrix} t^{m+1}_{m+1,m+1} & V^{m+1}_{m+1} \\ 0 & I \end{bmatrix}$$

and store $t^{m+1}_{m+1,m+1}$, V^{m+1}_{m+1} in the first row of A'^m_{22}. Let A^{m+1}_{22} denote the remaining rows and columns of A'^m_{22}, and rename the elements of V'^m as follows:

$$V'^m = \begin{bmatrix} t^{m+1}_{1,m+1} & V^{m+1}_1 \\ t^{m+1}_{m,m+1} & V^{m+1}_m \end{bmatrix}$$

The storage area now looks like this:

$$\begin{matrix} T^m & t^{m+1}_{1,m+1} & V^{m+1}_1 \\ & t^{m+1}_{m+1,m+1} & V^{m+1}_{m+1} \\ & & A^{m+1}_{22} \end{matrix}$$

Let T^{m+1} be the enlarged triangular t-array indicated above, (that is, add the column of t's to T^m), and let V^{m+1} be the V-array. An easy calculation shows that we are back in the situation (3), (4) with m replaced by m+1. This completes step m+1.

c. Final result of elimination

The above process is continued until for some m=r, say, $D^r = 0$ (in the sense mentioned above) or until m=n whichever comes first. In the case r < n, the result is the factorization:

$$A^r = P^r A (P^r)^T = \begin{bmatrix} (T^r)^T & 0 \\ (V^r)^T & I \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T^r & V^r \\ 0 & I \end{bmatrix}$$

where P^r is the permutation matrix associated with the permutation:

$$P^r = (r,k_r)(r-1,k_{r-1})\ldots(1,k_1)$$

T^r is nonsingular, and we may compute $U^r = (T^r)^{-1}V^r$ in the same storage locations occupied by V^r. An easy calculation shows:

$$A^r = \begin{bmatrix} I & 0 \\ (U^r)^T & I \end{bmatrix} \begin{bmatrix} A_s & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & U^r \\ 0 & I \end{bmatrix}$$

where $A_s = (T^r)^T T^r$. We also compute $I + (U^r)^T U^r$ and store it in the last n-r rows and columns of A, producing the storage scheme:

$$\begin{matrix} T^r & U^r \\ & I + (U^r)^T U^r \end{matrix}$$

We now operate on $I + (U^r)^T U^r$ in the same way that we operated on A. Whenever it is necessary to interchange two columns of this matrix, we also interchange the same columns of U^r in the storage area. Interchange information is recorded in TRAC (r+1) up to TRAC(n). The foregoing proof establishes a permutation matrix Q and an (n-r) by (n-r) nonsingular upper triangular T_u such that:

$$(T_u)^T T_u = Q(I + (U^r)^T U^r)Q^T = I + (U^r Q^T)^T (U^r Q^T)$$

Let $T = T^r$, $U = U^r Q^T$ and $p^{n-r} = \begin{bmatrix} I & 0 \\ 0 & Q \end{bmatrix}$. An easy calculation then gives the final results:

$$A^n = PAP^T = \begin{bmatrix} I & 0 \\ U^T & I \end{bmatrix} \begin{bmatrix} A_s & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & U \\ 0 & I \end{bmatrix} \tag{6}$$

$$(T_u)^T T_u = I + U^T U \tag{7}$$

where $P = P^{n-r}P^r$. The final storage scheme is

$$\begin{matrix} T & U \\ & T_u \end{matrix}$$

TRAC contains the permutation P^T in the form of a transposition vector $(k_1, k_2, \ldots, k_n)$.

d. Dependencies among rows and columns

The basic rows of A^n are $A_b = A_s\,[I, U]$. The nonbasic rows are then $U^T A_b$. Of course basic columns are $(A_b)^T$, and nonbasic columns are $(A_b)^T U$.

e. Systems of equations

Subroutines MFSS and DMFSS may be used to solve systems of equations

$$AX = R$$

with positive semidefinite coefficient matrix A. Using the permutation P, this system is transformed into the system

$$A^n X^n = R^n \tag{8}$$

where $X^n = PX$ and $R^n = PR$. If A has rank r=n, equation (8) is

$$A_s X^n = R^n$$

which is easily solved since $A_s^{-1} = T^{-1}(T^{-1})^T$.

Assuming $r < n$, the vectors X^n, R^n are partitioned as follows:

$$X^n = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \text{ and } R^n = \begin{bmatrix} R_1 \\ R_2 \end{bmatrix},$$

where X_1 consists of the first r elements of X^n, etc. Using the decomposition (6), it is easy to see that (8) is equivalent to:

$$\begin{aligned} A_s(X_1 + UX_2) &= R_1 \\ 0X_2 &= -U^T R_1 + R_2 \end{aligned} \tag{9}$$

(By equivalent systems we mean they have the same solutions.)

If $R_2 = U^T R_1$ (a compatible system), equation (9) has the n-r parameter family of solutions:

$$\begin{aligned} X_1 &= A_s^{-1} R_1 - UX_2 \\ X_2 &= \text{arbitrary} \end{aligned} \tag{10}$$

If $R_2 \neq U^T R_1$ (an incompatible system), it is natural to ask for a least-squares solution of (8), that is a vector X^n such that $\|A^n X^n - R^n\|^2$ = minimum. This problem leads to the system of equations:

$$(A^n)^2 X^n = A^n R^n \tag{11}$$

Given the factorization (6) of A^n, it is easy to show that $B^n = (A^n)^2$ admits the factorization:

$$B^n = \begin{bmatrix} I & 0 \\ U^T & I \end{bmatrix} \begin{bmatrix} B_s & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & U \\ 0 & I \end{bmatrix}$$

where $B_s = A_s(I+UU^T)A_s$. Using this fact, (11) is equivalent to:

$$\begin{aligned} B_s(X_1 + UX_2) &= A_s(R_1 + UR_2) \\ 0 \cdot X_2 &= 0 \cdot R_2 \end{aligned} \tag{12}$$

which is clearly a compatible system. Therefore (12) has the solutions:

$$\begin{aligned} X_1 &= B_s^{-1} A_s (R_1 + UR_2) - UX_2 \\ X_2 &= \text{arbitrary} \end{aligned} \tag{13}$$

Now the identity $(I + UU^T)^{-1} = I - U(I + U^TU)^{-1}U^T$ shows that it suffices to know $(I + U^TU)^{-1}$ in order to compute B_s^{-1}. But by (7):

$$(I + U^TU)^{-1} = T_u^{-1} (T_u^{-1})^T$$

and T_u^{-1} is easy to compute, since T_u is upper triangular.

f. Solutions of minimal length

The solutions (10) or (13) contain n-r free parameters X_2, and it is possible to choose them so that X^n has minimal Euclidean length, that is so that $\|X^n\| = \min$. Some easy calculations show that in the compatible case (10), X^n is given by:

$$X_1 = (I + UU^T)^{-1} A_s^{-1} R_1$$

$$X_2 = U^T X_1$$

and in the incompatible case (13), X^n is:

$$X_1 = (I + UU^T)^{-1} A_s^{-1} (I + UU^T)^{-1} (R_1 + UR_2)$$

$$X_2 = U^T X_1$$

g. Accuracy

Determining the rank of a matrix on a computer is a difficult matter, and it is not claimed that MFSS will compute the correct rank in all cases. This is an intrinsic difficulty due to the fact that we work with finite arithmetic.

h. Error returns

Normally IRANK contains the computed rank r. A nonpositive value returned in IRANK means:

IRANK = 0 A has no positive diagonal element, or $|\epsilon| \geq 1$.

IRANK = -1 A nonpositive dimension of A was specified.

IRANK = -2 $I + U^TU$ is ill-conditioned and/or the relative tolerance ϵ was specified incorrectly.

```
C                                                                       MFSS  10
C  ......................................................................MFSS  20
C                                                                       MFSS  30
C     SUBROUTINE MFSS                                                   MFSS  40
C                                                                       MFSS  50
C     PURPOSE                                                           MFSS  60
C        GIVEN A SYMMETRIC POSITIVE SEMI DEFINITE MATRIX , MFSS WILL    MFSS  70
C        (1) DETERMINE THE RANK AND LINEARLY INDEPENDENT ROWS AND       MFSS  80
C            COLUMNS                                                    MFSS  90
C        (2) FACTOR A SYMMETRIC SUBMATRIX OF MAXIMAL RANK               MFSS 100
C        (3) EXPRESS NONBASIC ROWS IN TERMS OF BASIC ONES.              MFSS 110
C            EXPRESS NONBASIC COLUMNS IN TERMS OF BASIC ONES            MFSS 120
C            EXPRESS BASIC VARIABLES IN TERMS OF FREE ONES              MFSS 130
C        SUBROUTINE MFSS MAY BE USED AS A PREPARATORY STEP FOR THE      MFSS 140
C        CALCULATION OF THE LEAST SQUARES SOLUTION OF MINIMAL           MFSS 150
C        LENGTH OF A SYSTEM OF LINEAR EQUATIONS WITH SYMMETRIC          MFSS 160
C        POSITIVE SEMI-DEFINITE COEFFICIENT MATRIX                      MFSS 170
C                                                                       MFSS 180
C     USAGE                                                             MFSS 190
C        CALL MFSS(A,N,EPS,IRANK,TRAC)                                  MFSS 200
C                                                                       MFSS 210
C     DESCRIPTION OF PARAMETERS                                         MFSS 220
C        A      - UPPER TRIANGULAR PART OF GIVEN SYMMETRIC SEMI-        MFSS 230
C                 DEFINITE MATRIX STORED COLUMNWISE IN COMPRESSED FORMMFSS 240
C                 ON RETURN A CONTAINS THE MATRIX T AND, IF IRANK IS    MFSS 250
C                 LESS THAN N, THE MATRICES U AND TU                    MFSS 260
C        N      - DIMENSION OF GIVEN MATRIX A                           MFSS 270
C        EPS    - TESTVALUE FOR ZERO AFFECTED BY ROUND-OFF NOISE        MFSS 280
C        IRANK  - RESULTANT VARIABLE, CONTAINING THE RANK OF GIVEN      MFSS 290
C                 MATRIX A IF A IS SEMI-DEFINITE                        MFSS 300
C                 IRANK = 0 MEANS A HAS NO POSITIVE DIAGONAL ELEMENT    MFSS 310
C                           AND/OR EPS IS NOT ABSOLUTELY LESS THAN ONEMFSS 320
C                 IRANK =-1 MEANS DIMENSION N IS NOT POSITIVE           MFSS 330
C                 IRANK =-2 MEANS COMPLETE FAILURE, POSSIBLY DUE TO     MFSS 340
C                           INADEQUATE RELATIVE TOLERANCE EPS           MFSS 350
C        TRAC   - VECTOR OF DIMENSION N CONTAINING THE                  MFSS 360
C                 SOURCE INDEX OF THE I-TH PIVOT ROW IN ITS I-TH        MFSS 370
C                 LOCATION, THIS MEANS THAT TRAC CONTAINS THE           MFSS 380
C                 PRODUCT REPRESENTATION OF THE PERMUTATION WHICH       MFSS 390
C                 IS APPLIED TO ROWS AND COLUMNS OF A IN TERMS OF       MFSS 400
C                 TRANSPOSITIONS                                        MFSS 410
C                                                                       MFSS 420
C     REMARKS                                                           MFSS 430
C        EPS MUST BE ABSOLUTELY LESS THAN ONE. A SENSIBLE VALUE IS      MFSS 440
C        SOMEWHERE IN BETWEEN 10**(-4) AND 10**(-6)                     MFSS 450
C        THE ABSOLUTE VALUE OF INPUT PARAMETER EPS IS USED AS           MFSS 460
C        RELATIVE TOLERANCE.                                            MFSS 470
C        IN ORDER TO PRESERVE SYMMETRY ONLY PIVOTING ALONG THE          MFSS 480
C        DIAGONAL IS BUILT IN.                                          MFSS 490
C        ALL PIVOTELEMENTS MUST BE GREATER THAN THE ABSOLUTE VALUE      MFSS 500
C        OF EPS TIMES ORIGINAL DIAGONAL ELEMENT                         MFSS 510
C        OTHERWISE THEY ARE TREATED AS IF THEY WERE ZERO                MFSS 520
C        MATRIX A REMAINS UNCHANGED IF THE RESULTANT VALUE IRANK        MFSS 530
C        EQUALS ZERO                                                    MFSS 540
C                                                                       MFSS 550
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     MFSS 560
C        NONE                                                           MFSS 570
C                                                                       MFSS 580
C     METHOD                                                            MFSS 590
C        THE SQUARE ROOT METHOD WITH DIAGONAL PIVOTING IS USED FOR      MFSS 600
C        CALCULATION OF THE RIGHT HAND TRIANGULAR FACTOR.               MFSS 610
C        IN CASE OF AN ONLY SEMI-DEFINITE MATRIX THE SUBROUTINE         MFSS 620
C        RETURNS THE IRANK X IRANK UPPER TRIANGULAR FACTOR T OF A       MFSS 630
C        SUBMATRIX OF MAXIMAL RANK, THE IRANK X (N-IRANK) MATRIX U      MFSS 640
C        AND THE (N-IRANK) X (N-IRANK) UPPER TRIANGULAR TU SUCH         MFSS 650
C        THAT TRANSPOSE(TU)*TU=I+TRANSPOSE(U)*U                         MFSS 660
C                                                                       MFSS 670
C  ......................................................................MFSS 680
C                                                                       MFSS 690
      SUBROUTINE MFSS(A,N,EPS,IRANK,TRAC)                               MFSS 700
C                                                                       MFSS 710
C                                                                       MFSS 720
C        DIMENSIONED DUMMY VARIABLES                                    MFSS 730
      DIMENSION A(1),TRAC(1)                                            MFSS 740
      DOUBLE PRECISION SUM                                              MFSS 750
C                                                                       MFSS 760
C        TEST OF SPECIFIED DIMENSION                                    MFSS 770
      IF(N)36,36,1                                                      MFSS 780
```

```
      INITIALIZE TRIANGULAR FACTORIZATION
    1 IRANK=0
      ISUB=0
      KPIV=0
      J=0
      PIV=0.

      SEARCH FIRST PIVOT ELEMENT
      DO 3 K=1,N
      J=J+K
      TRAC(K)=A(J)
      IF(A(J)-PIV)3,3,2
    2 PIV=A(J)
      KSUB=J
      KPIV=K
    3 CONTINUE

      START LOOP OVER ALL ROWS OF A
      DO 32 I=1,N
      ISUB=ISUB+I
      IMI=I-1
    4 KMI=KPIV-I
      IF(KMI)35,9,5

      PERFORM PARTIAL COLUMN INTERCHANGE
    5 JI=KSUB-KMI
      IDC=JI-ISUB
      JJ=ISUB-IMI
      DO 6 K=JJ,ISUB
      KK=K+IDC
      HOLD=A(K)
      A(K)=A(KK)
    6 A(KK)=HOLD

      PERFORM PARTIAL ROW INTERCHANGE
      KK=KSUB
      DO 7 K=KPIV,N
      II=KK-KMI
      HOLD=A(KK)
      A(KK)=A(II)
      A(II)=HOLD
    7 KK=KK+K

      PERFORM REMAINING INTERCHANGE
      JJ=KPIV-1
      II=ISUB
      DO 8 K=I,JJ
      HOLD=A(II)
      A(II)=A(JI)
      A(JI)=HOLD
      II=II+K
    8 JI=JI+1
    9 IF(IRANK)22,10,10

      RECORD INTERCHANGE IN TRANSPOSITION VECTOR
   10 TRAC(KPIV)=TRAC(I)
      TRAC(I)=KPIV

      MODIFY CURRENT PIVOT ROW
      KK=IMI-IRANK
      KMI=ISUB-KK
      PIV=0.
      IDC=IRANK+1
      JI=ISUB-1
      JK=KMI
      JJ=ISUB-I
      DO 15 K=I,N
      SUM=0.D0

      BUILD UP SCALAR PRODUCT IF NECESSARY
      IF(KK)11,13,11
   11 DO 12 J=KMI,JI
      SUM=SUM-A(J)*A(JK)
   12 JK=JK+1
   13 JJ=JJ+K
      IF(K-I)14,14,16
   14 SUM=A(ISUB)+SUM

      TEST RADICAND FOR LOSS OF SIGNIFICANCE
      IF(SUM-ABS(A(ISUB)*EPS))20,20,15
   15 A(ISUB)=DSQRT(SUM)
      KPIV=I+1
      GOTO 19
   16 SUM=(A(JK)+SUM)/A(ISUB)
      A(JK)=SUM

      SEARCH FOR NEXT PIVOT ROW
      IF(A(JJ))19,19,17
   17 TRAC(K)=TRAC(K)-SUM*SUM
      HOLD=TRAC(K)/A(JJ)
      IF(PIV-HOLD)18,19,19
   18 PIV=HOLD
      KPIV=K
      KSUB=JJ
   19 JK=JJ+IDC
      GOTO 32

      CALCULATE MATRIX OF DEPENDENCIES U
   20 IF(IRANK)21,21,37
   21 IRANK=-1
      GOTO 4
   22 IRANK=IMI
      II=ISUB-IRANK
      JI=II
      DO 26 K=1,IRANK
      JI=JI-1
      JK=ISUB-1
      JJ=K-1
      DO 26 J=I,N
      IDC=IRANK
      SUM=0.D0
      KMI=JI
      KK=JK
      IF(JJ)25,25,23
   23 DO 24 L=1,JJ
      IDC=IDC-1
      SUM=SUM-A(KMI)*A(KK)
      KMI=KMI-IDC
   24 KK=KK-1
   25 A(KK)=(SUM+A(KK))/A(KMI)
   26 JK=JK+J

      CALCULATE I+TRANSPOSE(U)*U
      JJ=ISUB-I
      PIV=0.
      KK=ISUB-1
      DO 31 K=I,N
      JJ=JJ+K
      IDC=0
      DO 28 J=K,N
      SUM=0.D0
      KMI=JJ+IDC
      DO 27 L=II,KK
      JK=L+IDC
   27 SUM=SUM+A(L)*A(JK)
      A(KMI)=SUM
   28 IDC=IDC+J
      A(JJ)=A(JJ)+1.D0
      TRAC(K)=A(JJ)
C
C        SEARCH NEXT DIAGONAL ELEMENT
      IF(PIV-A(JJ))29,30,30
   29 KPIV=K
      KSUB=JJ
      PIV=A(JJ)
   30 II=II+K
      KK=KK+K
   31 CONTINUE
      GOTO 4
   32 CONTINUE
   33 IF(IRANK)35,34,35
   34 IRANK=N
   35 RETURN
C
C        ERROR RETURNS
C
C        RETURN IN CASE OF ILLEGAL DIMENSION
   36 IRANK=-1
      RETURN
C
C        INSTABLE FACTORIZATION OF I+TRANSPOSE(U)*U
   37 IRANK=-2
      RETURN
      END
```

```
C                                                                       DMSS  10
C     ..................................................................DMSS  20
C                                                                       DMSS  30
C        SUBROUTINE DMFSS                                               DMSS  40
C                                                                       DMSS  50
C        PURPOSE                                                        DMSS  60
C           GIVEN A SYMMETRIC POSITIVE SEMI DEFINITE MATRIX ,DMFSS WILL DMSS  70
C           (1) DETERMINE THE RANK AND LINEARLY INDEPENDENT ROWS AND    DMSS  80
C               COLUMNS                                                 DMSS  90
C           (2) FACTOR A SYMMETRIC SUBMATRIX OF MAXIMAL RANK            DMSS 100
C           (3) EXPRESS NONBASIC ROWS IN TERMS OF BASIC ONES,           DMSS 110
C               EXPRESS NONBASIC COLUMNS IN TERMS OF BASIC ONES         DMSS 120
C               EXPRESS BASIC VARIABLES IN TERMS OF FREE ONES           DMSS 130
C        SUBROUTINE DMFSS MAY BE USED AS A PREPARATORY STEP FOR THE     DMSS 140
C        CALCULATION OF THE LEAST SQUARES SOLUTION OF MINIMAL           DMSS 150
C        LENGTH OF A SYSTEM OF LINEAR EQUATIONS WITH SYMMETRIC          DMSS 160
C        POSITIVE SEMI-DEFINITE COEFFICIENT MATRIX                      DMSS 170
C                                                                       DMSS 180
C        USAGE                                                          DMSS 190
C           CALL DMFSS(A,N,EPS,IRANK,TRAC)                              DMSS 200
C                                                                       DMSS 210
C        DESCRIPTION OF PARAMETERS                                      DMSS 220
C           A     - UPPER TRIANGULAR PART OF GIVEN SYMMETRIC SEMI-      DMSS 230
C                   DEFINITE MATRIX STORED COLUMNWISE IN COMPRESSED FORMDMSS 240
C                   ON RETURN A CONTAINS THE MATRIX T AND, IF IRANK IS  DMSS 250
C                   LESS THAN N, THE MATRICES U AND TU                  DMSS 260
C                   A MUST BE OF DOUBLE PRECISION                       DMSS 270
C           N     - DIMENSION OF GIVEN MATRIX A                         DMSS 280
C           EPS   - TESTVALUE FOR ZERO AFFECTED BY ROUND-OFF NOISE      DMSS 290
C           IRANK - RESULTANT VARIABLE, CONTAINING THE RANK OF GIVEN    DMSS 300
C                   MATRIX A IF A IS SEMI-DEFINITE                      DMSS 310
C                   IRANK = 0 MEANS A HAS NO POSITIVE DIAGONAL ELEMENT  DMSS 320
C                             AND/OR EPS IS NOT ABSOLUTELY LESS THAN ONEDMSS 330
C                   IRANK =-1 MEANS DIMENSION N IS NOT POSITIVE         DMSS 340
C                   IRANK =-2 MEANS COMPLETE FAILURE, POSSIBLY DUE TO   DMSS 350
C                             INADEQUATE RELATIVE TOLERANCE EPS         DMSS 360
C           TRAC  - VECTOR OF DIMENSION N CONTAINING THE                DMSS 370
C                   SOURCE INDEX OF THE I-TH PIVOT ROW IN ITS I-TH      DMSS 380
C                   LOCATION. THIS MEANS THAT TRAC CONTAINS THE         DMSS 390
C                   PRODUCT REPRESENTATION OF THE PERMUTATION WHICH     DMSS 400
C                   IS APPLIED TO ROWS AND COLUMNS OF A IN TERMS OF     DMSS 410
C                   TRANSPOSITIONS                                      DMSS 420
C                   TRAC MUST BE OF DOUBLE PRECISION                    DMSS 430
C                                                                       DMSS 440
C        REMARKS                                                        DMSS 450
C           EPS MUST BE ABSOLUTELY LESS THAN ONE. A SENSIBLE VALUE IS   DMSS 460
C           SOMEWHERE IN BETWEEN 10**(-4) AND 10**(-6).                 DMSS 470
C           THE ABSOLUTE VALUE OF INPUT PARAMETER EPS IS USED AS        DMSS 480
C           RELATIVE TOLERANCE.                                         DMSS 490
C           IN ORDER TO PRESERVE SYMMETRY ONLY PIVOTING ALONG THE       DMSS 500
C           DIAGONAL IS BUILT IN.                                       DMSS 510
C           ALL PIVOTELEMENTS MUST BE GREATER THAN THE ABSOLUTE VALUE   DMSS 520
C           OF EPS TIMES ORIGINAL DIAGONAL ELEMENT                      DMSS 530
C           OTHERWISE THEY ARE TREATED AS IF THEY WERE ZERO             DMSS 540
C           MATRIX A REMAINS UNCHANGED IF THE RESULTANT VALUE IRANK     DMSS 550
C           EQUALS ZERO                                                 DMSS 560
C                                                                       DMSS 570
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DMSS 580
C           NONE                                                        DMSS 590
C                                                                       DMSS 600
C        METHOD                                                         DMSS 610
C           THE SQUARE ROOT METHOD WITH DIAGONAL PIVOTING IS USED FOR   DMSS 620
C           CALCULATION OF THE RIGHT HAND TRIANGULAR FACTOR.            DMSS 630
C           IN CASE OF AN ONLY SEMI-DEFINITE MATRIX THE SUBROUTINE      DMSS 640
C           RETURNS THE IRANK X IRANK UPPER TRIANGULAR FACTOR T OF A    DMSS 650
C           SUBMATRIX OF MAXIMAL RANK, THE IRANK X (N-IRANK) MATRIX U   DMSS 660
C           AND THE (N-IRANK) X (N-IRANK) UPPER TRIANGULAR TU SUCH      DMSS 670
C           THAT TRANSPOSE(TU)*TU=I+TRANSPOSE(U)*U                      DMSS 680
C                                                                       DMSS 690
C     ..................................................................DMSS 700
C                                                                       DMSS 710
      SUBROUTINE DMFSS(A,N,EPS,IRANK,TRAC)                              DMSS 720
C                                                                       DMSS 730
C                                                                       DMSS 740
C        DIMENSIONED DUMMY VARIABLES                                    DMSS 750
      DIMENSION A(1),TRAC(1)                                            DMSS 760
      DOUBLE PRECISION SUM,A,TRAC,PIV,HOLD                              DMSS 770
C                                                                       DMSS 780
C        TEST OF SPECIFIED DIMENSION                                    DMSS 790
      IF(N)36,36,1                                                      DMSS 800
C                                                                       DMSS 810
C        INITIALIZE TRIANGULAR FACTORIZATION                            DMSS 820
    1 IRANK=0                                                           DMSS 830
      ISUB=0                                                            DMSS 840
```

```
      KPIV=0
      J=0
      PIV=0.D0
C
C        SEARCH FIRST PIVOT ELEMENT
      DO 3 K=I,N
      J=J+K
      TRAC(K)=A(J)
      IF(A(J)-PIV)3,3,2
    2 PIV=A(J)
      KSUB=J
      KPIV=K
    3 CONTINUE
C
C        START LOOP OVER ALL ROWS OF A
      DO 32 I=1,N
      ISUB=ISUB+I
      IM1=I-1
    4 KMI=KPIV-I
      IF(KMI)35,6,5
C
C        PERFORM PARTIAL COLUMN INTERCHANGE
    5 JI=KSUB-KMI
      IDC=JI-ISUB
      JJ=ISUB-IM1
      DO 6 K=JJ,ISUB
      KK=K+IDC
      HOLD=A(K)
      A(K)=A(KK)
    6 A(KK)=HOLD
C
C        PERFORM PARTIAL ROW INTERCHANGE
      KK=KSUB
      DO 7 K=KPIV,N
      II=KK-KMI
      HOLD=A(KK)
      A(KK)=A(II)
      A(II)=HOLD
    7 KK=KK+K
C
C        PERFORM REMAINING INTERCHANGE
      JJ=KPIV-1
      II=ISUB
      DO 8 K=I,JJ
      HOLD=A(II)
      A(II)=A(JI)
      A(JI)=HOLD
      II=II+K
    8 JI=JI+1
    9 IF(IRANK)22,10,10
C
C        RECORD INTERCHANGE IN TRANSPOSITION VECTOR
   10 TRAC(KPIV)=TRAC(I)
      TRAC(I)=KPIV
C
C        MODIFY CURRENT PIVOT ROW
      KK=IM1-IRANK
      KMI=ISUB-KK
      PIV=0.D0
      IDC=IRANK+1
      JI=ISUB-1
      JK=KMI
      JJ=ISUB-I
      DO 19 K=I,N
      SUM=0.D0
C
C        BUILD UP SCALAR PRODUCT IF NECESSARY
      IF(KK)13,13,11
   11 DO 12 J=KMI,JI
      SUM=SUM-A(J)*A(JK)
   12 JK=JK+1
   13 JJ=JJ+K
      IF(K-I)14,14,15
   14 SUM=A(ISUB)+SUM
C
C        TEST RADICAND FOR LOSS OF SIGNIFICANCE
      IF(SUM-DABS(A(ISUB)*DBLE(EPS)))20,20,15
   15 A(ISUB)=DSQRT(SUM)
      KPIV=I+1
      GOTO 19
   16 SUM=(A(JK)+SUM)/A(ISUB)
      A(JK)=SUM
C
C        SEARCH FOR NEXT PIVOT ROW
      IF(A(JJ))19,19,17
   17 TRAC(K)=TRAC(K)-SUM*SUM
      HOLD=TRAC(K)/A(JJ)
      IF(PIV-HOLD)18,19,19
   18 PIV=HOLD
      KPIV=K
      KSUB=JJ
   19 JK=JJ+IDC
      GOTO 32
C
C        CALCULATE MATRIX OF DEPENDENCIES U
   20 IF(IRANK)21,21,37
   21 IRANK=-1
      GOTO 4
   22 IRANK=IM1
      II=ISUB-IRANK
      JI=II
      DO 26 K=1,IRANK
      JI=JI-1
      JK=ISUB-1
      JJ=K-1
      DO 26 J=I,N
      IDC=IRANK
      SUM=0.D0
      KMI=JI
      KK=JK
      IF(JJ)25,25,23
   23 DO 24 L=1,JJ
      IDC=IDC-1
      SUM=SUM-A(KMI)*A(KK)
      KMI=KMI-IDC
   24 KK=KK-1
   25 A(KK)=(SUM+A(KK))/A(KMI)
   26 JK=JK+J
C
C        CALCULATE I+TRANSPOSE(U)*U
      JJ=ISUB-I
      PIV=0.D0
      KK=ISUB-1
      DO 31 K=I,N
      JJ=JJ+K
      IDC=0
      DO 28 J=K,N
      SUM=0.D0
      KMI=JJ+IDC
      DO 27 L=II,KK
      JK=L+IDC
   27 SUM=SUM+A(L)*A(JK)
      A(KMI)=SUM
   28 IDC=IDC+J
      A(JJ)=A(JJ)+1.D0
      TRAC(K)=A(JJ)
C
C        SEARCH NEXT DIAGONAL ELEMENT
      IF(PIV-A(JJ))29,30,30
   29 KPIV=K
      KSUB=JJ
      PIV=A(JJ)
   30 II=II+K
      KK=KK+K
   31 CONTINUE
      GOTO 4
   32 CONTINUE
   33 IF(IRANK)35,34,35
   34 IRANK=N
   35 RETURN
C
C        ERROR RETURNS
C
C        RETURN IN CASE OF ILLEGAL DIMENSION
   36 IRANK=-1
      RETURN
C
C        INSTABLE FACTORIZATION OF I+TRANSPOSE(U)*U
   37 IRANK=-2
      RETURN
      END
```

Subroutines MFSD and DMFSD

These subroutines will compute a triangular factorization of a symmetric positive definite matrix using the square root method of Cholesky.

1. Mathematical background

Given an n by n symmetric positive definite matrix A, we compute an upper triangular matrix R such that

$$A = R^T R$$

The elements r_{ik} of R are computed using the following recursive formulas:

$$r_{1k} = a_{1k}/r_{11} \qquad k=1, 2, 3, \ldots, n$$

$$r_{jk} = (1/r_{jj})(a_{jk} - \sum_{i=1}^{j-1} r_{ij} r_{ik}) \qquad j=2, 3, \ldots, n$$

$$k=j, j+1, \ldots, n$$

Note: The determinant of A is $\det(A) = \left(\prod_{i=1}^{n} r_{ii}\right)^2$.

2. Programming considerations

The given matrix A is assumed stored columnwise in compressed form, that is upper triangular part only. MFSD stores the solution R in the same locations as A.

If any calculated radicand r_{kk}^2 $(k=1, 2, \ldots, n)$ is not positive, further calculation is bypassed, and the error parameter IER is set to -1. This means that A is not positive definite, possibly due to roundoff errors. IER is also set to -1 if the input parameter n is less than 1.

Let all radicands be positive and let r_{kk}^2 be the first radicand which is no longer greater than the internal tolerance $TOL = |EPS\ a_{kk}|$. The subroutine then gives the warning IER = k-1; however, calculation is continued. The warning indicates that there may be loss of significance at factorization step k due to loss of significant digits in the calculation of r_{kk}^2.

```
C
C     ..................................................................
C
C        SUBROUTINE MFSD
C
C        PURPOSE
C           FACTOR A GIVEN SYMMETRIC POSITIVE DEFINITE MATRIX
C
C        USAGE
C           CALL MFSD(A,N,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           A       - UPPER TRIANGULAR PART OF THE GIVEN SYMMETRIC
C                     POSITIVE DEFINITE N BY N COEFFICIENT MATRIX.
C                     ON RETURN A CONTAINS THE RESULTANT UPPER
C                     TRIANGULAR MATRIX.
C           N       - THE NUMBER OF ROWS (COLUMNS) IN GIVEN MATRIX.
C           EPS     - AN INPUT CONSTANT WHICH IS USED AS RELATIVE
C                     TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.
C           IER     - RESULTING ERROR PARAMETER CODED AS FOLLOWS
C                     IER=0  - NO ERROR
C                     IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARAME-
C                              TER N OR BECAUSE SOME RADICAND IS NON-
C                              POSITIVE (MATRIX A IS NOT POSITIVE
C                              DEFINITE, POSSIBLY DUE TO LOSS OF SIGNI-
C                              FICANCE)
C                     IER=K  - WARNING WHICH INDICATES LOSS OF SIGNIFI-
C                              CANCE. THE RADICAND FORMED AT FACTORIZA-
C                              TION STEP K+1 WAS STILL POSITIVE BUT NO
C                              LONGER GREATER THAN ABS(EPS*A(K+1,K+1)).
C
C        REMARKS
C           THE UPPER TRIANGULAR PART OF GIVEN MATRIX IS ASSUMED TO BE
C           STORED COLUMNWISE IN N*(N+1)/2 SUCCESSIVE STORAGE LOCATIONS.
C           IN THE SAME STORAGE LOCATIONS THE RESULTING UPPER TRIANGU-
C           LAR MATRIX IS STORED COLUMNWISE TOO.
C           THE PROCEDURE GIVES RESULTS IF N IS GREATER THAN 0 AND ALL
C           CALCULATED RADICANDS ARE POSITIVE.
C           THE PRODUCT OF RETURNED DIAGONAL TERMS IS EQUAL TO THE
C           SQUARE-ROOT OF THE DETERMINANT OF THE GIVEN MATRIX.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SOLUTION IS DONE USING THE SQUARE-ROOT METHOD OF CHOLESKY.
C           THE GIVEN MATRIX IS REPRESENTED AS PRODUCT OF TWO TRIANGULAR
C           MATRICES, WHERE THE LEFT HAND FACTOR IS THE TRANSPOSE OF
C           THE RETURNED RIGHT HAND FACTOR.
C
C     ..................................................................
C
      SUBROUTINE MFSD(A,N,EPS,IER)
C
C
      DIMENSION A(1)
      DOUBLE PRECISION DPIV,DSUM
C
C        TEST ON WRONG INPUT PARAMETER N
      IF(N-1) 12,1,1
    1 IER=0
C
C        INITIALIZE DIAGONAL-LOOP
      KPIV=0
      DO 11 K=1,N
      KPIV=KPIV+K
      IND=KPIV
      LEND=K-1
C
C        CALCULATE TOLERANCE
      TOL=ABS(EPS*A(KPIV))
C
C        START FACTORIZATION-LOOP OVER K-TH ROW
      DO 11 I=K,N
      DSUM=0.D0
      IF(LEND) 2,4,2
C
C        START INNER LOOP
    2 DO 3 L=1,LEND
      LANF=KPIV-L
      LIND=IND-L
    3 DSUM=DSUM+DBLE(A(LANF)*A(LIND))
C
C        TRANSFORM ELEMENT A(IND)
    4 DSUM=DBLE(A(IND))-DSUM
      IF(I-K) 10,5,10
C
C        TEST FOR NEGATIVE PIVOT ELEMENT AND FOR LOSS OF SIGNIFICANCE
    5 IF(SNGL(DSUM)-TOL) 6,6,9
    6 IF(DSUM) 12,12,7
    7 IF(IER) 8,8,9
    8 IER=K-1
C
C        COMPUTE PIVOT ELEMENT
    9 DPIV=DSQRT(DSUM)
      A(KPIV)=DPIV
      DPIV=1.D0/DPIV
      GO TO 11
C
C        CALCULATE TERMS IN ROW
   10 A(IND)=DSUM*DPIV
   11 IND=IND+I
C
C        END OF DIAGONAL-LOOP
      RETURN
   12 IER=-1
      RETURN
      END
```

```
C                                                                       DMSD  10
C     ...................................................................DMSD  20
C                                                                       DMSD  30
C        SUBROUTINE DMFSD                                               DMSD  40
C                                                                       DMSD  50
C        PURPOSE                                                        DMSD  60
C           FACTOR A GIVEN SYMMETRIC POSITIVE DEFINITE MATRIX           DMSD  70
C                                                                       DMSD  80
C        USAGE                                                          DMSD  90
C           CALL DMFSD(A,N,EPS,IER)                                     DMSD 100
C                                                                       DMSD 110
C        DESCRIPTION OF PARAMETERS                                      DMSD 120
C           A       - DOUBLE PRECISION UPPER TRIANGULAR PART OF GIVEN   DMSD 130
C                     SYMMETRIC POSITIVE DEFINITE N BY N COEFFICIENT    DMSD 140
C                     MATRIX.                                           DMSD 150
C                     ON RETURN A CONTAINS THE RESULTANT UPPER          DMSD 160
C                     TRIANGULAR MATRIX IN DOUBLE PRECISION.            DMSD 170
C           N       - THE NUMBER OF ROWS (COLUMNS) IN GIVEN MATRIX.     DMSD 180
C           EPS     - SINGLE PRECISION INPUT CONSTANT WHICH IS USED     DMSD 190
C                     AS RELATIVE TOLERANCE FOR TEST ON LOSS OF         DMSD 200
C                     SIGNIFICANCE.                                     DMSD 210
C           IER     - RESULTING ERROR PARAMETER CODED AS FOLLOWS        DMSD 220
C                     IER=0  - NO ERROR                                 D4SD 230
C                     IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARAME- DMSD 240
C                              TER N OR BECAUSE SOME RADICAND IS NON-   DMSD 250
C                              POSITIVE (MATRIX A IS NOT POSITIVE       DMSD 260
C                              DEFINITE, POSSIBLY DUE TO LOSS OF SIGNI- DMSD 270
C                              FICANCE)                                 DMSD 280
C                     IER=K  - WARNING WHICH INDICATES LOSS OF SIGNIFI- DMSD 290
C                              CANCE. THE RADICAND FORMED AT FACTORIZA- DMSD 300
C                              TION STEP K+1 WAS STILL POSITIVE BUT NO  DMSD 310
C                              LONGER GREATER THAN ABS(EPS*A(K+1,K+1)). DMSD 320
C                                                                       DMSD 330
C                                                                       DMSD 340
C        REMARKS                                                        DMSD 350
C           THE UPPER TRIANGULAR PART OF GIVEN MATRIX IS ASSUMED TO BE  DMSD 360
C           STORED COLUMNWISE IN N*(N+1)/2 SUCCESSIVE STORAGE LOCATIONS.DMSD 360
C           IN THE SAME STORAGE LOCATIONS THE RESULTING UPPER TRIANGU-  DMSD 370
C           LAR MATRIX IS STORED COLUMNWISE TOO.                        DMSD 380
C           THE PROCEDURE GIVES RESULTS IF N IS GREATER THAN 0 AND ALL  DMSD 390
C           CALCULATED RADICANDS ARE POSITIVE.                          DMSD 400
C           THE PRODUCT OF RETURNED DIAGONAL TERMS IS EQUAL TO THE      DMSD 410
C           SQUARE-ROOT OF THE DETERMINANT OF THE GIVEN MATRIX.         DMSD 420
C                                                                       DMSD 430
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DMSD 440
C           NONE                                                        DMSD 450
C                                                                       DMSD 460
C        METHOD                                                         DMSD 470
C           SOLUTION IS DONE USING THE SQUARE-ROOT METHOD OF CHOLESKY.  DMSD 480
C           THE GIVEN MATRIX IS REPRESENTED AS PRODUCT OF TWO TRIANGULARDMSD 490
C           MATRICES, WHERE THE LEFT HAND FACTOR IS THE TRANSPOSE OF    DMSD 500
C           THE RETURNED RIGHT HAND FACTOR.                             DMSD 510
C                                                                       DMSD 520
C     ...................................................................DMSD 530
C                                                                       DMSD 540
      SUBROUTINE DMFSD(A,N,EPS,IER)                                     DMSD 550
C                                                                       DMSD 560
C                                                                       DMSD 5
      DIMENSION A(1)                                                    DMSD 5
      DOUBLE PRECISION DPIV,DSUM,A                                      DMSD 590
C                                                                       DMSD 600
C        TEST ON WRONG INPUT PARAMETER N                                DMSD 610
      IF(N-1) 12,1,1                                                    DMSD 620
    1 IER=0                                                             DMSD 630
C                                                                       DMSD 640
C        INITIALIZE DIAGONAL-LOOP                                       DMSD 650
      KPIV=0                                                            DMSD 660
      DO 11 K=1,N                                                       DMSD 670
      KPIV=KPIV+K                                                       DMSD 680
      IND=KPIV                                                          DMSD 690
      LEND=K-1                                                          DMSD 700
C                                                                       DMSD 710
C        CALCULATE TOLERANCE                                            DMSD 720
      TOL=ABS(EPS*SNGL(A(KPIV)))                                        DMSD 730
C                                                                       DMSD 740
C        START FACTORIZATION-LOOP OVER K-TH ROW                         DMSD 750
      DO 11 I=K,N                                                       DMSD 760
      DSUM=0.D0                                                         DMSD 770
      IF(LEND) 2,4,2                                                    DMSD 780
C                                                                       DMSD 790
C        START INNER LOOP                                               DMSD 800
    2 DO 3 L=1,LEND                                                     DMSD 810
      LANF=KPIV-L                                                       DMSD 820
      LIND=IND-L                                                        DMSD 830
    3 DSUM=DSUM+A(LANF)*A(LIND)                                         DMSD 840
C        END OF INNER LOOP                                              DMSD 850
C                                                                       DMSD 860
C        TRANSFORM ELEMENT A(IND)                                       DMSD 870
    4 DSUM=A(IND)-DSUM                                                  DMSD 880
      IF(I-K) 10,5,10                                                   DMSD 890
C                                                                       DMSD 900
C        TEST FOR NEGATIVE PIVOT ELEMENT AND FOR LOSS OF SIGNIFICANCE   DMSD 910
    5 IF(SNGL(DSUM)-TOL) 6,6,9                                          DMSD 920
    6 IF(DSUM) 12,12,7                                                  DMSD 930
    7 IF(IER) 8,8,9                                                     DMSD 940
    8 IER=K-1                                                           DMSD 950
C                                                                       DMSD 960
C        COMPUTE PIVOT ELEMENT                                          DMSD 970
    9 DPIV=DSQRT(DSUM)                                                  DMSD 980
      A(KPIV)=DPIV                                                      DMSD 990
      DPIV=1.D0/DPIV                                                    DMSD1000
      GO TO 11                                                          DMSD1010
C                                                                       DMSD1020
C        CALCULATE TERMS IN ROW                                         DMSD1030
   10 A(IND)=DSUM*DPIV                                                  DMSD1040
   11 IND=IND+I                                                         DMSD1050
C        END OF DIAGONAL-LOOP                                           DMSD1060
C                                                                       DMSD1070
      RETURN                                                            DMSD1080
   12 IER=-1                                                            DMSD1090
      RETURN                                                            DMSD1100
      END                                                               DMSD1110
```

Subroutines LLSQ and DLLSQ

These subroutines obtain the solution of linear least-squares problems. Consider a real m by n matrix A of rank n ($m \geq n$) with elements a_{ik} ($i = 1,2,\ldots,m$; $k = 1,2,\ldots,n$) and l right-hand side column vectors B_j ($j = 1,2,\ldots,l$) of dimension m, written as an m by l right-hand side matrix B with elements b_{ik} ($i = 1,2,\ldots,m$; $k = 1,2,\ldots,l$). The problem is to determine l column vectors X_j ($j = 1,2,\ldots,l$) of dimension n -- written as an n by l matrix X with elements x_{ik} ($i = 1,2,\ldots,n$; $k = 1,2,\ldots,l$)-- such that:

$$\|B_j - A * X_j\| = \min_j \quad (j = 1,2,\ldots,l) \qquad (1)$$

where $\|R\|$ indicates the Euclidean norm of any column vector R with elements r_i ($i = 1,2,\ldots,m$); that is:

$$\|R\| = \sqrt{\sum_{i=1}^{m} r_i^2} \qquad (2)$$

In the special case m = n, solution of the linear least-squares problem means solution of the system of simultaneous linear equations:

$$A * X = B \qquad (3)$$

Solution of the given problem is based on the well known principle that transformation of any column vector R by an orthogonal matrix Q does not change its norm $\|R\|$; that is:

$$\|Q * R\| = \|R\| \qquad (4)$$

Writing the given problem in the following form:

$$\begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1l} \\ b_{21} & b_{22} & \cdots & b_{2l} \\ b_{31} & b_{32} & \cdots & b_{3l} \\ \vdots & \vdots & & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{ml} \end{pmatrix} - \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ a_{31} & a_{32} & \cdots & a_{3n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \cdot$$

$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1l} \\ x_{21} & x_{22} & \cdots & x_{2l} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nl} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1l} \\ r_{21} & r_{22} & \cdots & r_{2l} \\ r_{31} & r_{32} & \cdots & r_{3l} \\ \vdots & \vdots & & \vdots \\ r_{m1} & r_{m2} & \cdots & r_{ml} \end{pmatrix} \qquad (5)$$

the orthogonal transformation matrix Q is determined so that matrix A is transformed to upper triangular form; that is, all elements a_{ik} with $i > k$ are transformed to zero. An effective way to realize this decomposition is via Householder transformations. The algorithm is a recursive n-step procedure defined by the following recursion formulas:

$$A^{(1)} = A \qquad (6)$$

$$A^{(k+1)} = P^{(k)} * A^{(k)} \quad (k = 1,2,\ldots,n) \qquad (7)$$

In order to get an upper triangular matrix $A^{(n+1)}$, every matrix $P^{(k)}$ ($k = 1,2,\ldots,n$) should be defined so that it is symmetric and orthogonal, and so that it transforms all elements of the kth column of $A^{(k)}$ below the main diagonal to zero. All these restrictions to $P^{(k)}$ are satisfied, setting:

$$p^{(k)} = I - \beta_k * U^{(k)} * U^{(k)^T} \qquad (8)$$

with

$$\beta_k = \frac{1}{\sigma_k \, (\sigma_k + a_{kk}^{(k)})} \qquad (9)$$

and

$$\sigma_k = \pm \sqrt{\sum_{i=k}^{m} (a_{ik}^{(k)})^2} \quad \text{with} \begin{cases} + \text{ for } a_{kk}^{(k)} \geq o \\ - \text{ for } a_{kk}^{(k)} < o \end{cases} \qquad (10)$$

where I is the identity matrix.

$U^{(k)T}$ denotes the transpose of column vector $U^{(k)}$, the m components of which are defined as follows:

$$u_i^{(k)} = 0 \quad \text{for } i < k \tag{11}$$

$$u_k^{(k)} = \sigma_k + a_{kk}^{(k)} \tag{12}$$

$$u_i^{(k)} = a_{ik}^{(k)} \quad \text{for } i > k \tag{13}$$

Neither matrices $P^{(k)}$ $(k = 1, 2, \ldots, n)$ nor matrix $Q = P^{(n)} * P^{(n-1)} * \ldots * P^{(1)}$ are computed explicitly, since from (7) and (8):

$$A^{(k+1)} = A^{(k)} - U^{(k)} * Y^{(k)T} \tag{14}$$

with

$$Y^{(k)T} = \beta_k * U^{(k)T} * A^{(k)} \tag{15}$$

Writing the components of row vector $Y^{(k)T}$ explicitly:

$$y_j^{(k)} = 0 \quad \text{for } j < k \tag{16}$$

$$y_k^{(k)} = 1 \tag{17}$$

$$y_j^{(k)} = \beta_k \sum_{i=k}^{m} u_i^{(k)} \cdot a_{ij}^{(k)} \quad \text{for } j > k \tag{18}$$

Using (14), (16), (17), and (18), the explicit transformation formulas for matrix $A^{(k)}$ appear as follows:

$$a_{kk}^{(k+1)} = -\sigma_k \tag{19}$$

$$a_{ik}^{(k+1)} = 0 \quad (i = k+1, k+2, \ldots, m) \tag{20}$$

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - u_i^{(k)} \cdot y_j^{(k)}$$
$$(j = k+1, k+2, \ldots, n \text{ and for fixed } j, i = k, k+1, \ldots, m) \tag{21}$$

All other elements of matrix $A^{(k)}$ do not change. Naturally, the same transformation is performed on matrix B.

Using (6) and (7):

$$B^{(1)} = B \tag{22}$$

$$B^{(k+1)} = P^{(k)} * B^{(k)} \quad (k = 1, 2, \ldots, n) \tag{23}$$

Using (8) and (23):

$$B^{(k+1)} = B^{(k)} - U^{(k)} * Z^{(k)T} \tag{24}$$

with

$$Z^{(k)T} = \beta_k * U^{(k)T} * B^{(k)} \tag{25}$$

or explicitly:

$$b_{ij}^{(k+1)} = b_{ij}^{(k)} - u_i^{(k)} \cdot z_j^{(k)} \quad (j = 1, 2, \ldots, l;\ i = 1, 2, \ldots, m) \tag{26}$$

with

$$z_j^{(k)} = \beta_k \sum_{i=k}^{m} u_i^{(k)} \cdot b_{ij}^{(k)} \quad (j = 1, 2, \ldots, l) \tag{27}$$

In order to keep roundoff errors as small as possible, column interchange is performed in such a way that, at the kth stage, the column of $A^{(k)}$ is chosen to be reduced next, which will maximize $|a_{kk}^{(k+1)}|$ Using (19) and (10), the index of this column is determined by giving the maximum overall j of:

$$s_j^{(k)} = \sum_{i=k}^{m} (a_{ij}^{(k)})^2 \quad (j = k, k+1, \ldots, n) \tag{28}$$

After $A^{(k+1)}$ has been computed, it is possible to compute $s_j^{(k+1)}$ as follows:

$$s_j^{(k+1)} = s_j^{(k)} - (a_{kj}^{(k+1)})^2 \quad (j = k+1, k+2, \ldots, n) \tag{29}$$

since the orthogonal transformations leave the column lengths invariant.

After having computed matrices $A^{(n+1)}$ (upper triangular matrix) and $B^{(n+1)}$, the computation of the n by l solution matrix X is performed by back substitution according to the following formulas:

$$x_{ni} = \frac{1}{a_{nn}^{(n+1)}} \cdot b_{ni}^{(n+1)} \qquad (i = 1, 2, \ldots, l) \tag{30}$$

$$x_{ki} = \frac{1}{a_{kk}^{(n+1)}} \left(b_{ki}^{(n+1)} - a_{k,k+1}^{(n+1)} \cdot x_{k+1,i} - a_{k,k+2}^{(n+1)} \cdot x_{k+2,i} - \ldots - a_{k,n-1}^{(n+1)} \cdot x_{n-1,i} - a_{kn}^{(n+1)} \cdot x_{ni}\right)$$

$$(k = n-j+1;\ j = 2, 3, \ldots, n \text{ and } i = 1, 2, \ldots, l \text{ for fixed } j) \tag{31}$$

After any row of matrix X has been computed, back interchange of rows is performed according to column interchanges in matrices $A^{(k)}$ in order to get the correct sequence of components in the solution vectors X_j.

Finally, the vector of linear least squares with components

$$lsq_j = \sum_{i=n+1}^{m} (b_{ij}^{(n+1)})^2 \qquad (j = 1, 2, \ldots, l) \tag{32}$$

is computed, which gives the squares of the minimized Euclidean norms $\|B_j - A * X_j\|$ for all right-hand side column vectors B_j $(j = 1, 2, \ldots, l)$.

The only case in which the whole procedure can fail occurs when, at any stage k, no column with nonzero parameter σ_k can be found; that is, no nonzero main diagonal element can be generated. In this case, the rank of matrix A is less than n. With respect to roundoff errors, the test is made in the following way.

At first, all elements $s_j^{(1)}$ are computed according to (28) and the maximum overall j (that is, σ_1^2) is determined. With the relative tolerance ε given by input, the following absolute tolerance is generated:

$$tol = \sigma_1 \cdot \varepsilon \tag{33}$$

If at any stage k (= 1, 2, . . . , n+1) the square root of the maximum overall j = k+1, . . . , n of $s_j^{(k+1)}$, say piv, is not greater than tol, the rank of matrix A is declared to be k < n, and the procedure returns an error message to the calling program, thus indicating that no solution could be found. In this case, the last n - k elements of vector IPIV denote the useless columns of matrix A; the remaining useful columns form a base of matrix A.

For reference see G. Golub, "Numerical Methods for Solving Linear Least Squares Problems", Numerische Mathematik, vol. 7, no. 3 (1965), pp. 206 - 216.

```
C                                                                       LLSQ  10
C     ...................................................................LLSQ  20
C                                                                       LLSQ  30
C        SUBROUTINE LLSQ                                                LLSQ  40
C                                                                       LLSQ  50
C        PURPOSE                                                        LLSQ  60
C           TO SOLVE LINEAR LEAST SQUARES PROBLEMS, I.E. TO MINIMIZE    LLSQ  70
C           THE EUCLIDEAN NORM OF B-A*X, WHERE A IS A M BY N MATRIX     LLSQ  80
C           WITH M NOT LESS THAN N. IN THE SPECIAL CASE M=N SYSTEMS OF  LLSQ  90
C           LINEAR EQUATIONS MAY BE SOLVED.                             LLSQ 100
C                                                                       LLSQ 110
C        USAGE                                                          LLSQ 120
C           CALL LLSQ (A,B,M,N,L,X,IPIV,EPS,IER,AUX)                    LLSQ 130
C                                                                       LLSQ 140
C        DESCRIPTION OF PARAMETERS                                      LLSQ 150
C           A      - M BY N COEFFICIENT MATRIX (DESTROYED).             LLSQ 160
C           B      - M BY L RIGHT HAND SIDE MATRIX (DESTROYED).         LLSQ 170
C           M      - ROW NUMBER OF MATRICES A AND B.                    LLSQ 180
C           N      - COLUMN NUMBER OF MATRIX A. ROW NUMBER OF MATRIX X. LLSQ 190
C           L      - COLUMN NUMBER OF MATRICES B AND X.                 LLSQ 200
C           X      - N BY L SOLUTION MATRIX.                            LLSQ 210
C           IPIV   - INTEGER OUTPUT VECTOR OF DIMENSION N WHICH         LLSQ 220
C                    CONTAINS INFORMATIONS ON COLUMN INTERCHANGES       LLSQ 230
C                    IN MATRIX A. (SEE REMARK NO.3).                    LLSQ 240
C           EPS    - INPUT PARAMETER WHICH SPECIFIES A RELATIVE         LLSQ 250
C                    TOLERANCE FOR DETERMINATION OF RANK OF MATRIX A.   LLSQ 260
C           IER    - A RESULTING ERROR PARAMETER.                       LLSQ 270
C           AUX    - AUXILIARY STORAGE ARRAY OF DIMENSION MAX(2*N,L).   LLSQ 280
C                    ON RETURN FIRST L LOCATIONS OF AUX CONTAIN THE     LLSQ 290
C                    RESULTING LEAST SQUARES.                           LLSQ 300
C                                                                       LLSQ 310
C        REMARKS                                                        LLSQ 320
C           (1) NO ACTION BESIDES ERROR MESSAGE IER=-2 IN CASE          LLSQ 330
C           M LESS THAN N.                                              LLSQ 340
C           (2) NO ACTION BESIDES ERROR MESSAGE IER=-1 IN CASE          LLSQ 350
C           OF A ZERO-MATRIX A.                                         LLSQ 360
C           (3) IF RANK K OF MATRIX A IS FOUND TO BE LESS THAN N BUT    LLSQ 370
C              GREATER THAN 0, THE PROCEDURE RETURNS WITH ERROR CODE    LLSQ 380
C              IER=K INTO CALLING PROGRAM. THE LAST N-K ELEMENTS OF     LLSQ 390
C              VECTOR IPIV DENOTE THE USELESS COLUMNS IN MATRIX A.      LLSQ 400
C              THE REMAINING USEFUL COLUMNS FORM A BASE OF MATRIX A.    LLSQ 410
C           (4) IF THE PROCEDURE WAS SUCCESSFUL, ERROR PARAMETER IER    LLSQ 420
C              IS SET TO 0.                                             LLSQ 430
C                                                                       LLSQ 440
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  LLSQ 450
C           NONE                                                        LLSQ 460
C                                                                       LLSQ 470
C        METHOD                                                         LLSQ 480
C           HOUSEHOLDER TRANSFORMATIONS ARE USED TO TRANSFORM MATRIX A  LLSQ 490
C           TO UPPER TRIANGULAR FORM. AFTER HAVING APPLIED THE SAME     LLSQ 500
C           TRANSFORMATION TO THE RIGHT HAND SIDE MATRIX B, AN          LLSQ 510
C           APPROXIMATE SOLUTION OF THE PROBLEM IS COMPUTED BY          LLSQ 520
C           BACK SUBSTITUTION. FOR REFERENCE, SEE                       LLSQ 530
C           G. GOLUB, NUMERICAL METHODS FOR SOLVING LINEAR LEAST        LLSQ 540
C           SQUARES PROBLEMS, NUMERISCHE MATHEMATIK, VOL.7,             LLSQ 550
C           ISS.3 (1965), PP.206-216.                                   LLSQ 560
C                                                                       LLSQ 570
C     ...................................................................LLSQ 580
C                                                                       LLSQ 590
      SUBROUTINE LLSQ(A,B,M,N,L,X,IPIV,EPS,IER,AUX)                     LLSQ 600
C                                                                       LLSQ 610
      DIMENSION A(1),B(1),X(1),IPIV(1),AUX(1)                           LLSQ 620
C                                                                       LLSQ 630
C     ERROR TEST                                                        LLSQ 640
      IF(M-N)30,1,1                                                     LLSQ 650
C                                                                       LLSQ 660
C     GENERATION OF INITIAL VECTOR S(K) (K=1,2,...,N) IN STORAGE        LLSQ 670
C     LOCATIONS AUX(K) (K=1,2,...,N)                                    LLSQ 680
    1 PIV=0.                                                            LLSQ 690
      IEND=0                                                            LLSQ 700
      DO 4 K=1,N                                                        LLSQ 710
      IPIV(K)=K                                                         LLSQ 720
      H=0.                                                              LLSQ 730
      IST=IEND+1                                                        LLSQ 740
      IEND=IEND+M                                                       LLSQ 750
      DO 2 I=IST,IEND                                                   LLSQ 760
    2 H=H+A(I)*A(I)                                                     LLSQ 770
      AUX(K)=H                                                          LLSQ 780
      IF(H-PIV)4,4,3                                                    LLSQ 790
    3 PIV=H                                                             LLSQ 800
      KPIV=K                                                            LLSQ 810
    4 CONTINUE                                                          LLSQ 820
C                                                                       LLSQ 830
C     ERROR TEST                                                        LLSQ 840
      IF(PIV)31,31,5                                                    LLSQ 850
C                                                                       LLSQ 860
C     DEFINE TOLERANCE FOR CHECKING RANK OF A                           LLSQ 870
    5 SIG=SQRT(PIV)                                                     LLSQ 880
      TOL=SIG*ABS(EPS)                                                  LLSQ 890
C                                                                       LLSQ 900
C                                                                       LLSQ 910
C     DECOMPOSITION LOOP                                                LLSQ 920
      LM=L*M                                                            LLSQ 930
      IST=-M                                                            LLSQ 940
      DO 21 K=1,N                                                       LLSQ 950
      IST=IST+M+1                                                       LLSQ 960
```

```
      IEND=IST+M-K
      I=KPIV-K
      IF(I)8,8,6
C
C     INTERCHANGE K-TH COLUMN OF A WITH KPIV-TH IN CASE KPIV.GT.K
    6 H=AUX(K)
      AUX(K)=AUX(KPIV)
      AUX(KPIV)=H
      ID=I*M
      DO 7 I=IST,IEND
      J=I+ID
      H=A(I)
      A(I)=A(J)
    7 A(J)=H
C
C     COMPUTATION OF PARAMETER SIG
    8 IF(K-1)11,11,9
    9 SIG=0.
      DO 10 I=IST,IEND
   10 SIG=SIG+A(I)*A(I)
      SIG=SQRT(SIG)
C
C     TEST ON SINGULARITY
      IF(SIG-TOL)32,32,11
C
C     GENERATE CORRECT SIGN OF PARAMETER SIG
   11 H=A(IST)
      IF(H)12,13,13
   12 SIG=-SIG
C
C     SAVE INTERCHANGE INFORMATION
   13 IPIV(KPIV)=IPIV(K)
      IPIV(K)=KPIV
C
C     GENERATION OF VECTOR UK IN K-TH COLUMN OF MATRIX A AND OF
C     PARAMETER BETA
      BETA=H*SIG
      A(IST)=BETA
      BETA=1./(SIG*BETA)
      J=N*K
      AUX(J)=-SIG
      IF(K-N)14,19,19
C
C     TRANSFORMATION OF MATRIX A
   14 PIV=0.
      ID=0
      JST=K+1
      KPIV=JST
      DO 18 J=JST,N
      ID=ID+M
      H=0.
      DO 15 I=IST,IEND
      II=I+ID
   15 H=H+A(I)*A(II)
      H=BETA*H
      DO 16 I=IST,IEND
      II=I+ID
   16 A(II)=A(II)-A(I)*H
C
C     UPDATING OF ELEMENT S(J) STORED IN LOCATION AUX(J)
      II=IST+ID
      H=AUX(J)-A(II)*A(II)
      AUX(J)=H
      IF(H-PIV)18,18,17
   17 PIV=H
      KPIV=J
   18 CONTINUE
C
C     TRANSFORMATION OF RIGHT HAND SIDE MATRIX B
   19 DO 21 J=K,LM,M
      H=0.
      IEND=J+M-K
      II=IST
      DO 20 I=J,IEND
      H=H+A(II)*B(I)
   20 II=II+1
      H=BETA*H
      II=IST
      DO 21 I=J,IEND
      B(I)=B(I)-A(II)*H
   21 II=II+1
C     END OF DECOMPOSITION LOOP
C
C
C     BACK SUBSTITUTION AND BACK INTERCHANGE
      IER=0
      I=N
      LN=L*N
      PIV=1./AUX(2*N)
      DO 22 K=N,LN,N
      X(K)=PIV*B(I)
   22 I=I+M
      IF(N-1)25,26,23
   23 JST=(N-1)*M+N
      DO 25 J=2,N
      JST=JST-M-1
      K=N+N+1-J
      PIV=1./AUX(K)
      KST=K-M
      ID=IPIV(KST)-KST
      IST=2-J
      DO 25 K=1,L
      H=B(KST)
      IST=IST+N
      IEND=IST+J-2
      II=JST
      DO 24 I=IST,IEND
      II=II+M
   24 H=H-A(II)*X(I)
      I=IST-1
      II=I+ID
      X(I)=X(II)
      X(II)=PIV*H
   25 KST=KST+M
C
C
C     COMPUTATION OF LEAST SQUARES
   26 IST=N+1
      IEND=0
      DO 29 J=1,L
      IEND=IEND+M
      H=0.
      IF(M-N)29,29,27
   27 DO 28 I=IST,IEND
   28 H=H+B(I)*B(I)
      IST=IST+M
   29 AUX(J)=H
      RETURN
C
C     ERROR RETURN IN CASE M LESS THAN N
   30 IER=-2
      RETURN
C
C     ERROR RETURN IN CASE OF ZERO-MATRIX A
   31 IER=-1
      RETURN
C
C     ERROR RETURN IN CASE OF RANK OF MATRIX A LESS THAN N
   32 IER=K-1
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DLLSQ
C
C        PURPOSE
C           TO SOLVE LINEAR LEAST SQUARES PROBLEMS, I.E. TO MINIMIZE
C           THE EUCLIDEAN NORM OF B-A*X, WHERE A IS A M BY N MATRIX
C           WITH M NOT LESS THAN N. IN THE SPECIAL CASE M=N SYSTEMS OF
C           LINEAR EQUATIONS MAY BE SOLVED.
C
C        USAGE
C           CALL DLLSQ (A,B,M,N,L,X,IPIV,EPS,IER,AUX)
C
C        DESCRIPTION OF PARAMETERS
C           A       - DOUBLE PRECISION M BY N COEFFICIENT MATRIX
C                     (DESTROYED).
C           B       - DOUBLE PRECISION M BY L RIGHT HAND SIDE MATRIX
C                     (DESTROYED).
C           M       - ROW NUMBER OF MATRICES A AND B.
C           N       - COLUMN NUMBER OF MATRIX A, ROW NUMBER OF MATRIX X.
C           L       - COLUMN NUMBER OF MATRICES B AND X.
C           X       - DOUBLE PRECISION N BY L SOLUTION MATRIX.
C           IPIV    - INTEGER OUTPUT VECTOR OF DIMENSION N WHICH
C                     CONTAINS INFORMATIONS ON COLUMN INTERCHANGES
C                     IN MATRIX A. (SEE REMARK NO.3).
C           EPS     - SINGLE PRECISION INPUT PARAMETER WHICH SPECIFIES
C                     A RELATIVE TOLERANCE FOR DETERMINATION OF RANK OF
C                     MATRIX A.
C           IER     - A RESULTING ERROR PARAMETER.
C           AUX     - A DOUBLE PRECISION AUXILIARY STORAGE ARRAY OF
C                     DIMENSION MAX(2*N,L). ON RETURN FIRST L LOCATIONS
C                     OF AUX CONTAIN THE RESULTING LEAST SQUARES.
C
C        REMARKS
C           (1) NO ACTION BESIDES ERROR MESSAGE IER=-2 IN CASE
C               M LESS THAN N.
C           (2) NO ACTION BESIDES ERROR MESSAGE IER=-1 IN CASE
C               OF A ZERO-MATRIX A.
C           (3) IF RANK K OF MATRIX A IS FOUND TO BE LESS THAN N BUT
C               GREATER THAN 0, THE PROCEDURE RETURNS WITH ERROR CODE
C               IER=K INTO CALLING PROGRAM. THE LAST N-K ELEMENTS OF
C               VECTOR IPIV DENOTE THE USELESS COLUMNS IN MATRIX A.
C               THE REMAINING USEFUL COLUMNS FORM A BASE OF MATRIX A.
C           (4) IF THE PROCEDURE WAS SUCCESSFUL, ERROR PARAMETER IER
C               IS SET TO 0.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           HOUSEHOLDER TRANSFORMATIONS ARE USED TO TRANSFORM MATRIX A
C           TO UPPER TRIANGULAR FORM. AFTER HAVING APPLIED THE SAME
C           TRANSFORMATION TO THE RIGHT HAND SIDE MATRIX B, AN
C           APPROXIMATE SOLUTION OF THE PROBLEM IS COMPUTED BY
C           BACK SUBSTITUTION. FOR REFERENCE, SEE
C           J. GOLUB, NUMERICAL METHODS FOR SOLVING LINEAR LEAST
C           SQUARES PROBLEMS, NUMERISCHE MATHEMATIK, VOL.7,
C           ISS.3 (1965), PP.206-216.
C
C     ..................................................................
C
      SUBROUTINE DLLSQ(A,B,M,N,L,X,IPIV,EPS,IER,AUX)
C
      DIMENSION A(1),B(1),X(1),IPIV(1),AUX(1)
      DOUBLE PRECISION A,B,X,AUX,PIV,H,SIG,BETA,TOL
C
C     ERROR TEST
      IF(M-N)30,1,1
C
C     GENERATION OF INITIAL VECTOR S(K) (K=1,2,...,N) IN STORAGE
C     LOCATIONS AUX(K) (K=1,2,...,N)
    1 PIV=0.D0
      IEND=0
      DO 4 K=1,N
      IPIV(K)=K
      H=0.D0
      IST=IEND+1
      IEND=IEND+M
      DO 2 I=IST,IEND
    2 H=H+A(I)*A(I)
      AUX(K)=H
      IF(H-PIV)4,4,3
    3 PIV=H
      KPIV=K
    4 CONTINUE
C
C     ERROR TEST
      IF(PIV)31,31,5
C
C     DEFINE TOLERANCE FOR CHECKING RANK OF A
    5 SIG=DSQRT(PIV)
      TOL=SIG*ABS(EPS)
C
C
C     DECOMPOSITION LOOP
      LM=L*M
      IST=-M
      DO 21 K=1,N
      IST=IST+M+1
      IEND=IST+M-K
      I=KPIV-K
      IF(I)8,8,6
C
C     INTERCHANGE K-TH COLUMN OF A WITH KPIV-TH IN CASE KPIV.GT.K
    6 H=AUX(K)
      AUX(K)=AUX(KPIV)
      AUX(KPIV)=H
      ID=I*M
      DO 7 I=IST,IEND
```

```
      J=I+ID                                                            DLLS1120
      H=A(I)                                                            DLLS1130
      A(I)=A(J)                                                         DLLS1140
    7 A(J)=H                                                            DLLS1150
C                                                                       DLLS1160
C     COMPUTATION OF PARAMETER SIG                                      DLLS1170
    8 IF(K-1)11,11,9                                                    DLLS1180
    9 SIG=0.D0                                                          DLLS1190
      DO 10 I=IST,IEND                                                  DLLS1200
   10 SIG=SIG+A(I)*A(I)                                                 DLLS1210
      SIG=DSQRT(SIG)                                                    DLLS1220
C                                                                       DLLS1230
C     TEST ON SINGULARITY                                               DLLS1240
      IF(SIG-TOL)32,32,11                                               DLLS1250
C                                                                       DLLS1260
C     GENERATE CORRECT SIGN OF PARAMETER SIG                            DLLS1270
   11 H=A(IST)                                                          DLLS1280
      IF(H)12,13,13                                                     DLLS1290
   12 SIG=-SIG                                                          DLLS1300
C                                                                       DLLS1310
C     SAVE INTERCHANGE INFORMATION                                      DLLS1320
   13 IPIV(KPIV)=IPIV(K)                                                DLLS1330
      IPIV(K)=KPIV                                                      DLLS1340
C                                                                       DLLS1350
C     GENERATION OF VECTOR UK IN K-TH COLUMN OF MATRIX A AND OF         DLLS1360
C     PARAMETER BETA                                                    DLLS1370
      BETA=H+SIG                                                        DLLS1380
      A(IST)=BETA                                                       DLLS1390
      BETA=1.D0/(SIG*BETA)                                              DLLS1400
      J=N+K                                                             DLLS1410
      AUX(J)=-SIG                                                       DLLS1420
      IF(K-N)14,19,19                                                   DLLS1430
C                                                                       DLLS1440
C     TRANSFORMATION OF MATRIX A                                        DLLS1450
   14 PIV=0.D0                                                          DLLS1460
      ID=0                                                              DLLS1470
      JST=K+1                                                           DLLS1480
      KPIV=JST                                                          DLLS1490
      DO 18 J=JST,N                                                     DLLS1500
      ID=ID+M                                                           DLLS1510
      H=0.D0                                                            DLLS1520
      DO 15 I=IST,IEND                                                  DLLS1530
      II=I+ID                                                           DLLS1540
   15 H=H+A(I)*A(II)                                                    DLLS1550
      H=BETA*H                                                          DLLS1560
      DO 16 I=IST,IEND                                                  DLLS1570
      II=I+ID                                                           DLLS1580
   16 A(II)=A(II)-A(I)*H                                                DLLS1590
C                                                                       DLLS1600
C     UPDATING OF ELEMENT S(J) STORED IN LOCATION AUX(J)                DLLS1610
      II=IST+ID                                                         DLLS1620
      H=AUX(J)-A(II)*A(II)                                              DLLS1630
      AUX(J)=H                                                          DLLS1640
      IF(H-PIV)18,18,17                                                 DLLS1650
   17 PIV=H                                                             DLLS1660
      KPIV=J                                                            DLLS1670
   18 CONTINUE                                                          DLLS1680
C                                                                       DLLS1690
C     TRANSFORMATION OF RIGHT HAND SIDE MATRIX B                        DLLS1700
   19 DO 21 J=K,LM,M                                                    DLLS1710
      H=0.D0                                                            DLLS1720
      IEND=J+M-K                                                        DLLS1730
      II=IST                                                            DLLS1740
      DO 20 I=J,IEND                                                    DLLS1750
      H=H+A(II)*B(I)                                                    DLLS1760
   20 II=II+1                                                           DLLS1770
      H=BETA*H                                                          DLLS1780
      II=IST                                                            DLLS1790
      DO 21 I=J,IEND                                                    DLLS1800
      B(I)=B(I)-A(II)*H                                                 DLLS1810
   21 II=II+1                                                           DLLS1820
C     END OF DECOMPOSITION LOOP                                         DLLS1830
C                                                                       DLLS1840
C                                                                       DLLS1850
C     BACK SUBSTITUTION AND BACK INTERCHANGE                            DLLS1860
      IER=0                                                             DLLS1870
      I=N                                                               DLLS1880
      LN=L*N                                                            DLLS1890
      PIV=1.D0/AUX(2*N)                                                 DLLS1900
      DO 22 K=N,LN,N                                                    DLLS1910
      X(K)=PIV*B(I)                                                     DLLS1920
   22 I=I+M                                                             DLLS1930
      IF(N-1)26,26,23                                                   DLLS1940
   23 JST=(N-1)*M+N                                                     DLLS1950
      DO 25 J=2,N                                                       DLLS1960
      JST=JST-M-1                                                       DLLS1970
      K=N+N+1-J                                                         DLLS1980
      PIV=1.D0/AUX(K)                                                   DLLS1990
      KST=K+N                                                           DLLS2000
      ID=IPIV(KST)-KST                                                  DLLS2010
      IST=2-J                                                           DLLS2020
      DO 25 K=1,L                                                       DLLS2030
      H=B(KST)                                                          DLLS2040
      IST=IST+N                                                         DLLS2050
      IEND=IST+J-2                                                      DLLS2060
      II=JST                                                            DLLS2070
      DO 24 I=IST,IEND                                                  DLLS2080
      II=II+M                                                           DLLS2090
   24 H=H-A(II)*X(I)                                                    DLLS2100
      I=IST-1                                                           DLLS2110
      II=I+ID                                                           DLLS2120
      X(I)=X(II)                                                        DLLS2130
      X(II)=PIV*H                                                       DLLS2140
   25 KST=KST+M                                                         DLLS2150
C                                                                       DLLS2160
C                                                                       DLLS2170
C     COMPUTATION OF LEAST SQUARES                                      DLLS2180
   26 IST=N+1                                                           DLLS2190
      IEND=0                                                            DLLS2200
      DO 29 J=1,L                                                       DLLS2210
      IEND=IEND+M                                                       DLLS2220
      H=0.D0                                                            DLLS2230
      IF(M-N)29,29,27                                                   DLLS2240
   27 DO 28 I=IST,IEND                                                  DLLS2250
   28 H=H+B(I)*B(I)                                                     DLLS2260
      IST=IST+M                                                         DLLS2270
   29 AUX(J)=H                                                          DLLS2280
      RETURN                                                            DLLS2290
C                                                                       DLLS2300
C     ERROR RETURN IN CASE M LESS THAN N                                DLLS2310
   30 IER=-2                                                            DLLS2320
      RETURN                                                            DLLS2330
C                                                                       DLLS2340
C     ERROR RETURN IN CASE OF ZERO-MATRIX A                             DLLS2350
   31 IER=-1                                                            DLLS2360
      RETURN                                                            DLLS2370
C                                                                       DLLS2380
C     ERROR RETURN IN CASE OF RANK OF MATRIX A LESS THAN N              DLLS2390
   32 IER=K-1                                                           DLLS2400
      RETURN                                                            DLLS2410
      END
```

Matrices: Eigenanalysis and Related Topics

Subroutine EIGEN

This subroutine computes the eigenvalues and eigenvectors of a real symmetric matrix.

Given a symmetric matrix A of order N, eigenvalues are to be developed in the diagonal elements of the matrix. A matrix of eigenvectors R is also to be generated.

An identity matrix is used as a first approximation of R.

The initial off-diagonal norm is computed:

$$\nu_I = \left\{ \sum_{i<k} 2A_{ik}^2 \right\}^{1/2} \tag{1}$$

ν_I = initial norm

A = input matrix (symmetric)

This norm is divided by N at each stage to produce the threshold.

The final norm is computed:

$$\nu_F = \frac{\nu_I \times 10^{-6}}{N} \tag{2}$$

This final norm is set sufficiently small that the requirement that any off-diagonal element A_{lm} shall be smaller than ν_F in absolute magnitude defines the convergence of the process.

An indicator is initialized. This indicator is later used to determine whether any off-diagonal elements have been found that are greater than the present threshold.

Each off-diagonal element is selected in turn and a transformation is performed to annihilate the off-diagonal (pivotal) element, as shown by the following equations:

$$\lambda = -A_{lm} \tag{3}$$

$$\mu = 1/2\,(A_{11} - A_{mm}) \tag{4}$$

$$\omega = \text{sign}\,(\mu)\,\frac{\lambda}{\sqrt{\lambda^2 + \mu^2}} \tag{5}$$

$$\sin\theta = \frac{\omega}{\sqrt{2\,(1 + \sqrt{1 - \omega^2})}} \tag{6}$$

$$\cos\theta = \sqrt{1 - \sin^2\theta} \tag{7}$$

$$B = A_{il} \cos\theta - A_{im} \sin\theta \tag{8}$$

$$A_{im} = A_{il} \sin\theta + A_{im} \cos\theta \tag{9}$$

$$A_{il} = B \tag{10}$$

$$B = R_{il} \cos\theta - R_{im} \sin\theta \tag{11}$$

$$R_{im} = R_{il} \sin\theta + R_{im} \cos\theta \tag{12}$$

$$R_{il} = B \tag{13}$$

$$A_{ll} = A_{11} \cos^2\theta + A_{mm} \sin^2\theta - 2A_{lm} \sin\theta \cos\theta \tag{14}$$

$$A_{mm} = A_{ll} \sin^2\theta + A_{mm} \cos^2\theta + 2A_{lm} \sin\theta \cos\theta \tag{15}$$

$$A_{lm} = (A_{ll} - A_{mm}) \sin\theta \cos\theta + A_{lm} (\cos^2\theta - \sin^2\theta) \tag{16}$$

The above calculations are repeated until all of the pivotal elements are less than the threshold.

```
C                                                                       EIGE  10
C     ..................................................................EIGE  20
C                                                                       EIGE  30
C        SUBROUTINE EIGEN                                               EIGE  40
C                                                                       EIGE  50
C        PURPOSE                                                        EIGE  60
C           COMPUTE EIGENVALUES AND EIGENVECTORS OF A REAL SYMMETRIC    EIGE  70
C           MATRIX                                                      EIGE  80
C                                                                       EIGE  90
C        USAGE                                                          EIGE 100
C           CALL EIGEN(A,R,N,MV)                                        EIGE 110
C                                                                       EIGE 120
C        DESCRIPTION OF PARAMETERS                                      EIGE 130
C           A - ORIGINAL MATRIX (SYMMETRIC), DESTROYED IN COMPUTATION.  EIGE 140
C               RESULTANT EIGENVALUES ARE DEVELOPED IN DIAGONAL OF      EIGE 150
C               MATRIX A IN DESCENDING ORDER.                           EIGE 160
C           R - RESULTANT MATRIX OF EIGENVECTORS (STORED COLUMNWISE,    EIGE 170
C               IN SAME SEQUENCE AS EIGENVALUES)                        EIGE 180
C           N - ORDER OF MATRICES A AND R                               EIGE 190
C           MV- INPUT CODE                                              EIGE 200
C                 0  COMPUTE EIGENVALUES AND EIGENVECTORS               EIGE 210
C                 1  COMPUTE EIGENVALUES ONLY (R NEED NOT BE            EIGE 220
C                    DIMENSIONED BUT MUST STILL APPEAR IN CALLING       EIGE 230
C                    SEQUENCE)                                          EIGE 240
C                                                                       EIGE 250
C        REMARKS                                                        EIGE 260
C           ORIGINAL MATRIX A MUST BE REAL SYMMETRIC (STORAGE MODE=1)   EIGE 270
C           MATRIX A CANNOT BE IN THE SAME LOCATION AS MATRIX R         EIGE 280
C                                                                       EIGE 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  EIGE 300
C           NONE                                                        EIGE 310
C                                                                       EIGE 320
C        METHOD                                                         EIGE 330
C           DIAGONALIZATION METHOD ORIGINATED BY JACOBI AND ADAPTED     EIGE 340
C           BY VON NEUMANN FOR LARGE COMPUTERS AS FOUND IN *MATHEMATICAL EIGE 350
C           METHODS FOR DIGITAL COMPUTERS*, EDITED BY A. RALSTON AND    EIGE 360
C           H.S. WILF, JOHN WILEY AND SONS, NEW YORK, 1962, CHAPTER 7   EIGE 370
C                                                                       EIGE 380
C     ..................................................................EIGE 390
C                                                                       EIGE 400
      SUBROUTINE EIGEN(A,R,N,MV)                                        EIGE 410
      DIMENSION A(1),R(1)                                               EIGE 420
C                                                                       EIGE 430
C     ..................................................................EIGE 440
C                                                                       EIGE 450
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  EIGE 460
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      EIGE 470
C        STATEMENT WHICH FOLLOWS.                                       EIGE 480
C                                                                       EIGE 490
C     DOUBLE PRECISION A,R,ANORM,ANRMX,THR,X,Y,SINX,SINX2,COSX,         EIGE 500
C    1                 COSX2,SINCS,RANGE                                EIGE 510
C                                                                       EIGE 520
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    EIGE 530
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      EIGE 540
C        ROUTINE.                                                       EIGE 550
C                                                                       EIGE 560
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      EIGE 570
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENTS EIGE 580
C        40, 68, 75, AND 78 MUST BE CHANGED TO DSQRT.  ABS IN STATEMENT EIGE 590
C        62 MUST BE CHANGED TO DABS. THE CONSTANT IN STATEMENT 5 SHOULD EIGE 600
C        BE CHANGED TO 1.0D-12.                                         EIGE 610
C                                                                       EIGE 620
C     ..................................................................EIGE 630
C                                                                       EIGE 640
C        GENERATE IDENTITY MATRIX                                       EIGE 650
C                                                                       EIGE 660
    5 RANGE=1.0E-6                                                      EIGE 670
      IF(MV-1) 10,25,10                                                 EIGE 680
   10 IQ=-N                                                             EIGE 690
      DO 20 J=1,N                                                       EIGE 700
      IQ=IQ+N                                                           EIGE 710
      DO 20 I=1,N                                                       EIGE 720
      IJ=IQ+I                                                           EIGE 730
      R(IJ)=0.0                                                         EIGE 740
      IF(I-J) 20,15,20                                                  EIGE 750
   15 R(IJ)=1.0                                                         EIGE 760
   20 CONTINUE                                                          EIGE 770
C                                                                       EIGE 780
C        COMPUTE INITIAL AND FINAL NORMS (ANORM AND ANORMX)             EIGE 790
C                                                                       EIGE 800
   25 ANORM=0.0                                                         EIGE 810
      DO 35 I=1,N                                                       EIGE 820
      DO 35 J=I,N                                                       EIGE 830
      IF(I-J) 30,35,30                                                  EIGE 840
   30 IA=I+(J*J-J)/2                                                    EIGE 850
      ANORM=ANORM+A(IA)*A(IA)                                           EIGE 860
   35 CONTINUE                                                          EIGE 870
      IF(ANORM) 165,165,40                                              EIGE 880
   40 ANORM=1.414*SQRT(ANORM)                                           EIGE 890
      ANRMX=ANORM*RANGE/FLOAT(N)                                        EIGE 900
C                                                                       EIGE 910
C        INITIALIZE INDICATORS AND COMPUTE THRESHOLD, THR               EIGE 920
C                                                                       EIGE 930
      IND=0                                                             EIGE 940
      THR=ANORM                                                         EIGE 950
   45 THR=THR/FLOAT(N)                                                  EIGE 960
   50 L=1                                                               EIGE 970
   55 M=L+1                                                             EIGE 980
C                                                                       EIGE 990
C        COMPUTE SIN AND COS                                            EIGE1000
C                                                                       EIGE1010
   60 MQ=(M*M-M)/2                                                      EIGE1020
      LQ=(L*L-L)/2                                                      EIGE1030
      LM=L+MQ                                                           EIGE1040
   62 IF( ABS(A(LM))-THR) 130,65,65                                     EIGE1050
   65 IND=1                                                             EIGE1060
      LL=L+LQ                                                           EIGE1070
      MM=M+MQ                                                           EIGE1080
      X=0.5*(A(LL)-A(MM))                                               EIGE1090
   68 Y=-A(LM)/ SQRT(A(LM)*A(LM)+X*X)                                   EIGE1100
      IF(X) 70,75,75                                                    EIGE1110
   70 Y=-Y                                                              EIGE1120
   75 SINX=Y/ SQRT(2.0*(1.0+( SQRT(1.0-Y*Y))))                          EIGE1130
      SINX2=SINX*SINX                                                   EIGE1140
   78 COSX= SQRT(1.0-SINX2)                                             EIGE1150
      COSX2=COSX*COSX                                                   EIGE1160
      SINCS =SINX*COSX                                                  EIGE1170
C                                                                       EIGE1180
C        ROTATE L AND M COLUMNS                                         EIGE1190
C                                                                       EIGE1200
      ILQ=N*(L-1)                                                       EIGE1210
      IMQ=N*(M-1)                                                       EIGE1220
      DO 125 I=1,N                                                      EIGE1230
      IQ=(I*I-I)/2                                                      EIGE1240
      IF(I-L) 80,115,80                                                 EIGE1250
   80 IF(I-M) 85,115,90                                                 EIGE1260
   85 IM=I+MQ                                                           EIGE1270
      GO TO 95                                                          EIGE1280
   90 IM=M+IQ                                                           EIGE1290
   95 IF(I-L) 100,105,105                                               EIGE1300
  100 IL=I+LQ                                                           EIGE1310
      GO TO 110                                                         EIGE1320
  105 IL=L+IQ                                                           EIGE1330
  110 X=A(IL)*COSX-A(IM)*SINX                                           EIGE1340
      A(IM)=A(IL)*SINX+A(IM)*COSX                                       EIGE1350
      A(IL)=X                                                           EIGE1360
  115 IF(MV-1) 120,125,120                                              EIGE1370
  120 ILR=ILQ+I                                                         EIGE1380
      IMR=IMQ+I                                                         EIGE1390
      X=R(ILR)*COSX-R(IMR)*SINX                                         EIGE1400
      R(IMR)=R(ILR)*SINX+R(IMR)*COSX                                    EIGE1410
      R(ILR)=X                                                          EIGE1420
  125 CONTINUE                                                          EIGE1430
      X=2.0*A(LM)*SINCS                                                 EIGE1440
      Y=A(LL)*COSX2+A(MM)*SINX2-X                                       EIGE1450
      X=A(LL)*SINX2+A(MM)*COSX2+X                                       EIGE1460
      A(LM)=(A(LL)-A(MM))*SINCS+A(LM)*(COSX2-SINX2)                     EIGE1470
      A(LL)=Y                                                           EIGE1480
      A(MM)=X                                                           EIGE1490
C                                                                       EIGE1500
C        TESTS FOR COMPLETION                                           EIGE1510
C                                                                       EIGE1520
C        TEST FOR M = LAST COLUMN                                       EIGE1530
C                                                                       EIGE1540
  130 IF(M-N) 135,140,135                                               EIGE1550
  135 M=M+1                                                             EIGE1560
      GO TO 60                                                          EIGE1570
C                                                                       EIGE1580
C        TEST FOR L = SECOND FROM LAST COLUMN                           EIGE1590
C                                                                       EIGE1600
  140 IF(L-(N-1)) 145,150,145                                           EIGE1610
  145 L=L+1                                                             EIGE1620
      GO TO 55                                                          EIGE1630
  150 IF(IND-1) 160,155,160                                             EIGE1640
  155 IND=0                                                             EIGE1650
      GO TO 50                                                          EIGE1660
C                                                                       EIGE1670
C        COMPARE THRESHOLD WITH FINAL NORM                              EIGE1680
C                                                                       EIGE1690
  160 IF(THR-ANRMX) 165,165,45                                          EIGE1700
C                                                                       EIGE1710
C        SORT EIGENVALUES AND EIGENVECTORS                              EIGE1720
C                                                                       EIGE1730
  165 IQ=-N                                                             EIGE1740
      DO 185 I=1,N                                                      EIGE1750
      IQ=IQ+N                                                           EIGE1760
      LL=I+(I*I-I)/2                                                    EIGE1770
      JQ=N*(I-2)                                                        EIGE1780
      DO 185 J=I,N                                                      EIGE1790
      JQ=JQ+N                                                           EIGE1800
      MM=J+(J*J-J)/2                                                    EIGE1810
      IF(A(LL)-A(MM)) 170,185,185                                       EIGE1820
  170 X=A(LL)                                                           EIGE1830
      A(LL)=A(MM)                                                       EIGE1840
      A(MM)=X                                                           EIGE1850
      IF(MV-1) 175,185,175                                              EIGE1860
  175 DO 180 K=1,N                                                      EIGE1870
      ILR=IQ+K                                                          EIGE1880
      IMR=JQ+K                                                          EIGE1890
      X=R(ILR)                                                          EIGE1900
      R(ILR)=R(IMR)                                                     EIGE1910
  180 R(IMR)=X                                                          EIGE1920
  185 CONTINUE                                                          EIGE1930
      RETURN                                                            EIGE1940
      END                                                               EIGE1950
```

25 IF (N.EQ.1) RETURN

1.E-9

$A_{ij} = A(i + (j*j-j)/2)$

Storage mode 1

1	2	4	7
	3	5	8
		6	9
			10

Subroutine NROOT

This subroutine calculates the eigenvalues, λ_i, and the matrix of eigenvectors, V, of a real square nonsymmetric matrix of the special form $B^{-1}A$, where both B and A are real symmetric matrices and B is positive-definite. This subroutine is normally called by the subroutine CANOR in performing a canonical correlation analysis. The computational steps are as follows.

A symmetric matrix (storage mode 1) is formed by using the upper triangle elements of the square matrix B. Then the eigenvalues, h_i, and the matrix of eigenvectors, H, of the symmetric matrix are calculated by the subroutine EIGEN.

The reciprocal of square root of each eigenvalue is formed as follows:

$$\mu_i = \frac{1}{\sqrt{h_i}} \tag{1}$$

where $i = 1, 2, \ldots, m$

m = order of matrix B

The matrix $B^{-1/2}$ is formed by multiplying the ith column vector of H by μ_j, where $j = 1, 2, \ldots, m$.

The symmetric matrix $S = (B^{-1/2})' AB^{-1/2}$ is formed in the following two matrix multiplications:

$$Q = (B^{-1/2})' A \tag{2}$$

$$S = QB^{-1/2} \tag{3}$$

and eigenvalues, λ_i, and the matrix of eigenvectors, H, of S are calculated by the subroutine EIGEN.

The matrix $W = B^{-1/2}M$ is formed, and the vectors in W are normalized to form the matrix of eigenvectors, V, by the following equation:

$$V_{ij} = \frac{W_{ij}}{\sqrt{SUMV_j}} \tag{4}$$

where $i = 1, 2, \ldots, m$

$j = 1, 2, \ldots, m$

$$SUMV_j = \sum_{i=1}^{m} W_{ij}^2 \tag{5}$$

```
.......................................................................
      SUBROUTINE NROOT
      PURPOSE
         COMPUTE EIGENVALUES AND EIGENVECTORS OF A REAL NONSYMMETRIC
         MATRIX OF THE FORM B-INVERSE TIMES A.  THIS SUBROUTINE IS
         NORMALLY CALLED BY SUBROUTINE CANOR IN PERFORMING A
         CANONICAL CORRELATION ANALYSIS.
      USAGE
         CALL NROOT (M,A,B,XL,X)
C        DESCRIPTION OF PARAMETERS
C           M  - ORDER OF SQUARE MATRICES A, B, AND X.
C           A  - INPUT MATRIX (M X M).
C           B  - INPUT MATRIX (M X M).
C           XL - OUTPUT VECTOR OF LENGTH M CONTAINING EIGENVALUES OF
C                B-INVERSE TIMES A.
C           X  - OUTPUT MATRIX (M X M) CONTAINING EIGENVECTORS COLUMN-
C                WISE.
C        REMARKS
C           NONE
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           EIGEN
C        METHOD
C           REFER TO W. W. COOLEY AND P. R. LOHNES, 'MULTIVARIATE PRO-
C           CEDURES FOR THE BEHAVIORAL SCIENCES', JOHN WILEY AND SONS,
C           1962, CHAPTER 3.
C     .................................................................
      SUBROUTINE NROOT (M,A,B,XL,X)
      DIMENSION A(1),B(1),XL(1),X(1)
C     .................................................................
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C     DOUBLE PRECISION A,B,XL,X,SUMV
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  SQRT IN STATEMENTS
C        110 AND 175 MUST BE CHANGED TO DSQRT.  ABS IN STATEMENT 110
C        MUST BE CHANGED TO DABS.
C     .................................................................
C     COMPUTE EIGENVALUES AND EIGENVECTORS OF B
      K=1
      DO 100 J=2,M
      L=M*(J-1)
      DO 100 I=1,J
      L=L+1
      K=K+1
  100 B(K)=B(L)
C        THE MATRIX B IS A REAL SYMMETRIC MATRIX.
      MV=0
      CALL EIGEN (B,X,M,MV)
C     FORM RECIPROCALS OF SQUARE ROOT OF EIGENVALUES.  THE RESULTS
C     ARE PREMULTIPLIED BY THE ASSOCIATED EIGENVECTORS.
      L=0
      DO 110 J=1,M
      L=L+J
  110 XL(J)=1.0/ SQRT( ABS(B(L)))
      K=0
      DO 115 J=1,M
      DO 115 I=1,M
      K=K+1
  115 B(K)=X(K)*XL(J)
C     FORM (B**(-1/2))PRIME * A * (B**(-1/2))
      DO 120 I=1,M
      N2=0
      DO 120 J=1,M
      N1=M*(I-1)
      L=M*(J-1)+I
      X(L)=0.0
      DO 120 K=1,M
      N1=N1+1
      N2=N2+1
  120 X(L)=X(L)+B(N1)*A(N2)
      L=0
      DO 130 J=1,M
      DO 130 I=1,J
      N1=I-M
      N2=M*(J-1)
      L=L+1
      A(L)=0.0
      DO 130 K=1,M
      N1=N1+M
      N2=N2+1
  130 A(L)=A(L)+X(N1)*B(N2)
C     COMPUTE EIGENVALUES AND EIGENVECTORS OF A
      CALL EIGEN (A,X,M,MV)
      L=0
      DO 140 I=1,M
      L=L+I
  140 XL(I)=A(L)
C     COMPUTE THE NORMALIZED EIGENVECTORS
      DO 150 I=1,M
      N2=0
      DO 150 J=1,M
      N1=I-M
      L=M*(J-1)+I
      A(L)=0.0
      DO 150 K=1,M
      N1=N1+M
      N2=N2+I
  150 A(L)=A(L)+B(N1)*X(N2)
      L=0
      K=0
      DO 180 J=1,M
      SUMV=0.0
      DO 170 I=1,M
      L=L+1
  170 SUMV=SUMV+A(L)*A(L)
  175 SUMV= SQRT(SUMV)
      DO 180 I=1,M
      K=K+1
  180 X(K)=A(K)/SUMV
      RETURN
      END
```

Subroutine ATEIG

This subroutine computes the eigenvalues of a real upper almost-triangular matrix (Hessenberg form -- see subroutine HSBG) using the double QR iteration of J. G. F. Francis.

1. Mathematical background

a. Definition of the QR iteration

Let A be a real or complex nonsingular matrix of order n. Then a decomposition of A exists of the form

$$A = Q\,R$$

where Q is unitary and R is upper triangular. If the diagonal elements of R are real and positive, Q is unique. Consider now the sequence of matrices $A^{(p)}$ defined recursively by

$$A^{(o)} = A,\ A^{(p)} = Q^{(p)}\,R^{(p)},\ A^{(p+1)} = R^{(p)}\,Q^{(p)}\ p \geq 0.$$

Note that $A^{(p+1)} = Q^{(p)*}\,A^{(p)}\,Q^{(p)}$ for $p \geq 0$; hence it follows that $A^{(p)}$ is similar to A for all p.

Furthermore, if A satisfies certain conditions, it can be proved that $A^{(p)}$ tends to an upper triangular matrix as $p \to \infty$; thus the eigenvalues of A are the diagonal elements of this limit matrix.

b. Convergence

If the moduli of the eigenvalues are distinct, the elements $a^{(p)}_{ij}$ below the main diagonal of $A^{(p)}$ tend to zero, as $d^{o\,ij}|\lambda_i|^p/|\lambda_j|^p$, the eigenvalues being subscripted so that $|\lambda_i| > |\lambda_{i+1}|$

Thus, in general, the eigenvalues appear on the main diagonal, starting from the last position, in increasing order of moduli.

So, when the smallest eigenvalue λ_n has been found, we can reduce the order of the matrix by neglecting the last row and column and find λ_{n-1} by the same process, without any special deflation.

Note that the speed of convergence is considerably improved when the origin of the eigenvalues is shifted close to λ_n.

Such a shift, say $s^{(p)}$, can be introduced before an iteration and the opposite one afterwards. Then the iteration can be written as:

$$A^{(p)} - s^{(p)}\,I = Q^{(p)}\,R^{(p)}$$

$$A^{(p+1)} = R^{(p)}\,Q^{(p)} + s^{(p)}\,I$$

In general, $A^{(p)}_{n,n}$, for p large enough, can provide an efficient value for $s^{(p)}$.

c. Use of the Hessenberg form

The Hessenberg form is preserved under the QR iteration. Thus, a reduction of the initial matrix to the Hessenberg form can give a significant saving of computation in each iteration for the QR decomposition, the lower part of the matrix consisting only of the codiagonal terms.

Before each iteration, the codiagonal terms will be inspected. If some of these are zero, the matrix will be split according to this occurrence, and the iteration will be applied to the lower main submatrix only.

d. The double QR iteration

Let A be a diagonalizable real upper Hessenberg matrix. Such a matrix must be expected to have complex conjugate pairs of eigenvalues. If these pairs are the only eigenvalues of equal modulus, it can be shown that they will appear as the latent roots of main submatrices of order 2. In this case, if a shift is close to one of these roots, it will be complex, and we will have to deal with complex matrices, although the initial one is real. The use of the double QR iteration avoids this inconvenience.

Taking $s^{(p+1)} = \bar{s}^{(p)}$, consider the transformation giving $A^{(p+2)}$ from $A^{(p)}$:

$$A^{(p+2)} = Q^{(p+1)*}\,Q^{(p)*}\,A^{(p)}\,Q^{(p)}\,Q^{(p+1)}$$

It can be proved that the product $Q^{(p)}Q^{(p+1)}$ derives from the QR decomposition of the matrix $M = (A^{(p)} - s^{(p)}\,I)\,(A^{(p)} - s^{(p+1)}I)$, which is real.

In fact, Francis (1961, 1962) showed that only the first column m_1 of M is necessary for determining the transformation which gives $A^{(p+2)}$ from $A^{(p)}$, if they both have the Hessenberg form.

Practically, the first part of the double iteration consists of the application of an initial transformation $N_1^*\,A^{(p)}\,N_1$ where N_1 is unitary and such that $N_1^*\,m_1 = \pm\|m_1\|\,e_1$. This leads to a matrix which no longer has the Hessenberg form.

Thus, the remaining part of the iteration will involve the application of (n-1) successive transformations, which have the same form as the initial one whose matrices N_i are such that the resulting matrix $A^{(p+2)}$ has the Hessenberg form.

This process can fail when a subdiagonal term of the given matrix is zero. In this case, the matrix can be split, and the iteration is performed on the lower main submatrix only.

In the subroutine, N_i are Householder's matrices.

2. Programming considerations

At each iteration, the latent roots x_1 and x_2 of the lower main submatrix of order 2 are computed. Then, the following situations can occur:

a. The term $a_{n-1,n-2}$ can be taken as zero. Then, x_1 and x_2 are eigenvalues of the original matrix, and the order of the matrix is reduced by 2. IANA(N) and IANA(N-1) are set to 0 and 2 respectively.

b. The term $a_{n,n-1}$ can be taken as zero. In this case, $a_{n,n}$ is an eigenvalue of the original matrix, and the order of the matrix is reduced by 1. IANA(N) is set to 1.

c. One of the last two subdiagonal terms is stable through one iteration. Then the smaller one is considered as zero. The corresponding components of IANA are set to 0, 1, or 2, according to a. or b.

d. The maximum number of iterations is reached. In this case, the smaller of the last two subdiagonal elements is taken as zero. The corresponding components of IANA are set to 0, 1, or 2, according to a. or b.

The user can check the results by inspecting the subdiagonal terms of the matrix on return from the subroutine, according to the vector IANA, in the following way:

If for each IANA(I) containing 1 or 2, $2 \le I \le M$,

$$\left| A(I,I-1) \right| \le 10^{-7} \left(\left| RR(I) \right| + \left| RI(I) \right| \right),$$

then RR(I) and RI(I) were computed with a satisfactory accuracy.

For reference see:

1) J. G. F. Francis, Computer Journal. October, 1961 4-3, January, 1962 4-4.
2) J. H. Wilkinson, The Algebraic Eigenvalue Problem. Clarendon Press, Oxford, 1965.

```
C     ...................................................................ATEI 430
C                                                                        ATEI 440
      SUBROUTINE ATEIG(M,A,RR,RI,IANA,IA)                                ATEI 450
      DIMENSION A(1),RR(1),RI(1),PRR(2),PRI(2),IANA(1)                   ATEI 460
      INTEGER P,P1,C                                                     ATEI 470
C                                                                        ATEI 480
      E7=1.0E-8                                                          ATEI 490
      E6=1.0E-6                                                          ATEI 500
      E10=1.0E-10                                                        ATEI 510
      DELTA=0.5                                                          ATEI 520
      MAXIT=30                                                           ATEI 530
C                                                                        ATEI 540
C        INITIALIZATION                                                  ATEI 550
C                                                                        ATEI 560
      N=M                                                                ATEI 570
   20 N1=N-1                                                             ATEI 580
      IN=N1*IA                                                           ATEI 590
      NN=IN+N                                                            ATEI 600
      IF(N1) 30,130,30                                                   ATEI 610
   30 NP=N+1                                                             ATEI 620
C                                                                        ATEI 630
C        ITERATION COUNTER                                               ATEI 640
C                                                                        ATEI 650
      IT=0                                                               ATEI 660
C                                                                        ATEI 670
C        ROOTS OF THE 2ND ORDER MAIN SUBMATRIX AT THE PREVIOUS           ATEI 680
C        ITERATION                                                       ATEI 690
C                                                                        ATEI 700
      DO 40 I=1,2                                                        ATEI 710
      PRR(I)=0.0                                                         ATEI 720
   40 PRI(I)=0.0                                                         ATEI 730
C                                                                        ATEI 740
C        LAST TWO SUBDIAGONAL ELEMENTS AT THE PREVIOUS ITERATION         ATEI 750
C                                                                        ATEI 760
      PAN=0.0                                                            ATEI 770
      PAN1=0.0                                                           ATEI 780
C                                                                        ATEI 790
C        ORIGIN SHIFT                                                    ATEI 800
C                                                                        ATEI 810
      R=0.0                                                              ATEI 820
      S=0.0                                                              ATEI 830
C                                                                        ATEI 840
C        ROOTS OF THE LOWER MAIN 2 BY 2 SUBMATRIX                        ATEI 850
C                                                                        ATEI 860
      N2=N1-1                                                            ATEI 870
      IN1=IN-IA                                                          ATEI 880
      NN1=IN1+N                                                          ATEI 890
      N1N=IN+N1                                                          ATEI 900
      N1N1=IN1+N1                                                        ATEI 910
   60 T=A(N1N1)-A(NN)                                                    ATEI 920
      L=T*T                                                              ATEI 930
      V=4.0*A(N1N)*A(NN1)                                                ATEI 940
      IF(ABS(V)-U*E7) 100,100,65                                         ATEI 950
   65 T=U+V                                                              ATEI 960
      IF(ABS(T)-AMAX1(L,ABS(V))*E6) 67,67,68                             ATEI 970
   67 T=0.0                                                              ATEI 980
   68 L=(A(N1N1)+A(NN))/2.0                                              ATEI 990
      V=SQRT(ABS(T))/2.0                                                 ATEI1000
      IF(T)140,70,70                                                     ATEI1010
   70 IF(L) 80,75,75                                                     ATEI1020
   75 RR(N1)=L+V                                                         ATEI1030
      RR(N)=L-V                                                          ATEI1040
      GO TO 130                                                          ATEI1050
   80 RR(N1)=L-V                                                         ATEI1060
      RR(N)=L+V                                                          ATEI1070
      GO TO 130                                                          ATEI1080
  100 IF(T)120,110,110                                                   ATEI1090
  110 RR(N1)=A(N1N1)                                                     ATEI1100
      RR(N)=A(NN)                                                        ATEI1110
      GO TO 130                                                          ATEI1120
  120 RR(N1)=A(NN)                                                       ATEI1130
      RR(N)=A(N1N1)                                                      ATEI1140
  130 RI(N)=0.0                                                          ATEI1150
      RI(N1)=0.0                                                         ATEI1160
      GO TO 160                                                          ATEI1170
  140 RR(N1)=L                                                           ATEI1180
      RR(N)=L                                                            ATEI1190
      RI(N1)=V                                                           ATEI1200
      RI(N)=-V                                                           ATEI1210
  160 IF(N2)120,120,180                                                  ATEI1220
C                                                                        ATEI1230
C        TESTS OF CONVERGENCE                                            ATEI1240
C                                                                        ATEI1250
  180 N1N2=N1N1-IA                                                       ATEI1260
      RMOD=RR(N1)*RR(N1)+RI(N1)*RI(N1)                                   ATEI1270
      EPS=E10*SQRT(RMOD)                                                 ATEI1280
      IF(ABS(A(N1N2))-EPS)120,120,240                                    ATEI1290
  240 IF(ABS(A(NN1))-E10*ABS(A(NN))) 130,130,250                         ATEI1300
  250 IF(ABS(PAN1-A(N1N2))-ABS(A(N1N2))*E6) 140,140,260                  ATEI1310
  260 IF(ABS(PAN-A(NN1))-ABS(A(NN1))*E6)140,140,300                      ATEI1320
  300 IF(IT-MAXIT) 320,140,140                                           ATEI1330
C                                                                        ATEI1340
C        COMPUTE THE SHIFT                                               ATEI1350
C                                                                        ATEI1360
  320 J=1                                                                ATEI1370
      DO 360 I=1,2                                                       ATEI1380
      K=NP-I                                                             ATEI1390
      IF(ABS(RR(K)-PRR(I))+ABS(RI(K)-PRI(I))-DELTA*(ABS(RR(K))           ATEI1400
     1    +ABS(RI(K)))) 340,360,360                                      ATEI1410
  340 J=J+1                                                              ATEI1420
  360 CONTINUE                                                           ATEI1430
      GO TO (440,460,460,460),J                                          ATEI1440
  440 R=0.0                                                              ATEI1450
      S=0.0                                                              ATEI1460
      GO TO 500                                                          ATEI1470
  460 J=N+2-J                                                            ATEI1480
      R=RR(J)+RR(J)                                                      ATEI1490
      S=RR(J)*RR(J)                                                      ATEI1500
      GO TO 500                                                          ATEI1510
  480 R=RR(N1)+RR(N1)-RI(N1)*RI(N1)                                      ATEI1520
      S=RR(N1)*RR(N1)                                                    ATEI1530
C                                                                        ATEI1540
C        SAVE THE LAST TWO SUBDIAGONAL TERMS AND THE ROOTS OF THE        ATEI1550
C        SUBMATRIX BEFORE ITERATION                                      ATEI1560
C                                                                        ATEI1570
  500 PAN=A(NN1)                                                         ATEI1580
      PAN1=A(N1N2)                                                       ATEI1590
      DO 520 I=1,2                                                       ATEI1600
      K=NP-I                                                             ATEI1610
      PRR(I)=RR(K)                                                       ATEI1620
  520 PRI(I)=RI(K)                                                       ATEI1630
C                                                                        ATEI1640
C        SEARCH FOR A PARTITION OF THE MATRIX, DEFINED BY P AND Q        ATEI1650
C                                                                        ATEI1660
      P=N2                                                               ATEI1670
      IF (N-3)600,600,525                                                ATEI1675
  525 IPI=N1N2                                                           ATEI1680
      DO 580 J=2,N2                                                      ATEI1690
      IPI=IPI-IA-1                                                       ATEI1700
      IF(ABS(A(IPI))-EPS) 600,600,530                                    ATEI1710
```

```
 53C IPIF=IFI+IA                                                        ATEI1720
     IPIF2=IPIF+IA                                                      ATEI1730
     C=A(IPIF)*(A(IPIP1-S)+A(IPIP2)*A(IPIP+1)+R                         ATEI1740
     IF(C)54C,56C,54C                                                   ATEI1750
 54C IF(ABS(A(IPI)*A(IPIP+1))*(ABS(A(IPIP)+A(IPIP2+1)-S)+ABS(A(IPIP2+2)ATEI1760
    1 )) -ABS(C)*EPS) 62C,62C,56C                                       ATEI1770
 56C F=N1-J                                                             ATEI1780
 58C CCNTINLE                                                           ATEI1790
 6CC C=P                                                                ATEI1800
     GC TC 6EC                                                          ATEI1810
 62C F1=F-1                                                             ATEI1820
     C=P1                                                               ATEI1830
     IF (P1-1) 68C,6EC,65C                                              ATEI1835
 65C CC 6EC I=2, P1                                                     ATEI1840
     IPI=IPI-IA-1                                                       ATEI1850
     IF(ABS(A(IPI))-EPS)6EC,68C,66C                                     ATEI1860
 66C C=C-1                                                              ATEI1870
C                                                                       ATEI1880
C        CR ECLBLE ITERATICN                                            ATEI1890
C                                                                       ATEI1900
 68C II=(P-1)*IA+P                                                      ATEI1910
     CC 122C I=P,N1                                                     ATEI1920
     II1=II-IA                                                          ATEI1930
     IIP=II+IA                                                          ATEI1940
     IF(I-P)72C,7CC,72C                                                 ATEI1950
 7CC IPI=II+1                                                           ATEI1960
     IPIP=IIP+1                                                         ATEI1970
C                                                                       ATEI1980
C        INITIALIZATICN CF THE TRANSFCRMATICN                           ATEI1990
C                                                                       ATEI2000
     G1=A(II)*(A(II)-S)+A(IIP)*A(IPI)+R                                 ATEI2010
     G2=A(IPI)*(A(IPIP)+A(II)-S)                                        ATEI2020
     G3=A(IPI)*A(IPIP+1)                                                ATEI2030
     A(IPI+1)=C.C                                                       ATEI2040
     GC TC 7EC                                                          ATEI2050
 72C G1=A(III)                                                          ATEI2060
     G2=A(III+1)                                                        ATEI2070
     IF(I-N2)74C,74C,76C                                                ATEI2080
 74C G3=A(III+2)                                                        ATEI2090
     GC TC 7EC                                                          ATEI2100
 76C G3=C.C                                                             ATEI2110
 7EC CAF=SCRT(G1*G1+G2*G2+G3*G3)                                        ATEI2120
     IF(CAF)ECC,E6C,8CC                                                 ATEI2130
 8CC IF(G1)E2C,E4C,E4C                                                  ATEI2140
 82C CAF=-CAF                                                           ATEI2150
 E4C T=G1+CAF                                                           ATEI2160
     FSI1=G2/T                                                          ATEI2170
     FSI2=G3/T                                                          ATEI2180
     ALPHA=2.C/(1.C+PSI1*PSI1+PSI2*PSI2)                                ATEI2190
     GC TC 88C                                                          ATEI2200
 86C ALPHA=2.C                                                          ATEI2210
     FSI1=C.C                                                           ATEI2220
     FSI2=C.C                                                           ATEI2230
 88C IF(I-C)SCC,S6C,9CC                                                 ATEI2240
 SCC IF(I-P)S2C,S4C,92C                                                 ATEI2250
 92C A(III)=-CAP                                                        ATEI2260
     GC TC S6C                                                          ATEI2270
 S4C A(III)=-A(III)                                                     ATEI2280
C                                                                       ATEI2290
C        RCW CPERATICN                                                  ATEI2300
C                                                                       ATEI2310
 S6C IJ=II                                                              ATEI2320
     CC 1C4C J=I,N                                                      ATEI2330
     T=FSI1*A(IJ+1)                                                     ATEI2340
     IF(I-N1)9EC,1CCC,1CCC                                              ATEI2350
 9EC IP2J=IJ+2                                                          ATEI2360
     T=T+FSI2*A(IP2J)                                                   ATEI2370
 1CCC ETA=ALPHA*(T+A(IJ))                                               ATEI2380
     A(IJ)=A(IJ)-ETA                                                    ATEI2390
     A(IJ+1)=A(IJ+1)-PSI1*ETA                                           ATEI2400
     IF(I-N)1C2C,1C4C,1C4C                                              ATEI2410
 1C2C A(IP2J)=A(IP2J)-FSI2*ETA                                          ATEI2420
 1C4C IJ=IJ+IA                                                          ATEI2430
C                                                                       ATEI2440
C        CCLUMN CPERATICN                                               ATEI2450
C                                                                       ATEI2460
     IF(I-N)11C8C,1C6C,1C6C                                             ATEI2470
 1C6C K=N                                                               ATEI2480
     GC TC 11CC                                                         ATEI2490
 1C8C K=I+2                                                             ATEI2500
 11CC IF=1IF-1                                                          ATEI2510
     CC 11EC J=G,K                                                      ATEI2520
     JIF=IP+J                                                           ATEI2530
     JI=JIF-IA                                                          ATEI2540
     T=FSI1*A(JIP)                                                      ATEI2550
     IF(I-N)112C,114C,114C                                              ATEI2560
 112C JIF2=JIF+IA                                                       ATEI2570
     T=T+PSI2*A(JIP2)                                                   ATEI2580
 114C ETA=ALPHA*(T+A(JI))                                               ATEI2590
     A(JI)=A(JI)-ETA                                                    ATEI2600
     A(JIF)=A(JIF)-ETA*PSI1                                             ATEI2610
     IF(I-N)116C,118C,118C                                              ATEI2620
 116C A(JIF2)=A(JIP2)-ETA*PSI2                                          ATEI2630
 118C CCNTINLE                                                          ATEI2640
     IF(I-N2)12CC,122C,122C                                             ATEI2650
 12CC JI=II+3                                                           ATEI2660
     JIF=JI+IA                                                          ATEI2670
     JIF2=JIF+IA                                                        ATEI2680
     ETA=ALPHA*PSI2*A(JIP2)                                             ATEI2690
     A(JI)=-ETA                                                         ATEI2700
     A(JIF)=-ETA*PSI1                                                   ATEI2710
     A(JIP2)=A(JIP2)-ETA*PSI2                                           ATEI2720
 122C II=IIF+1                                                          ATEI2730
     IT=IT+1                                                            ATEI2740
     GC TC 6C                                                           ATEI2750
C                                                                       ATEI2760
C        ENC CF ITERATICN                                               ATEI2770
C                                                                       ATEI2780
 124C IF(ABS(A(NN1))-ABS(A(N1N2))) 13CC,128C,128C                       ATEI2790
C                                                                       ATEI2800
C        THE EIGENVALUES HAVE BEEN FCUND                                ATEI2810
C                                                                       ATEI2820
 128C IANA(N)=C                                                         ATEI2830
     IANA(N1)=2                                                         ATEI2840
     N=N2                                                               ATEI2850
     IF(N2)14CC,14CC,2C                                                 ATEI2860
C                                                                       ATEI2870
C        CNE EIGENVALLE HAS BEEN FCUND                                  ATEI2880
C                                                                       ATEI2890
 13CC RR(N)=A(NN)                                                       ATEI2900
     RI(N)=C.C                                                          ATEI2910
     IANA(N)=1                                                          ATEI2920
     IF(N)14CC,14CC,132C                                                ATEI2930
 132C N=N1                                                              ATEI2940
     GC TC 2C                                                           ATEI2950
 14CC RETURN                                                            ATEI2960
     END                                                                ATEI2970
```

Subroutine HSBG

This subroutine reduces an n by n real matrix A by a similarity transformation to upper almost-triangular (Hessenberg) form. Each row is reduced in turn, starting from the last one, by applying a suitable right elimination matrix, and similarity is achieved by also applying the left inverse transformation. Thus the eigenvalues of A are preserved.

1. Mathematical background

Let $A^{(p)}$ denote the matrix obtained from $A^{(o)} = A$ after reducing rows $n, n-1, \ldots, n-p+1$. The similarity which transforms $A^{(p)}$ to $A^{(p+1)}$ is as follows:

a. First we determine a pivot element $a^{(p)}_{n-p,k}$ whose column subscript k is such that

$$\left| \left| a^{(p)}_{n-p,k} \right| = \max_i \left| a^{(p)}_{n-p,i} \right| \right. , i = 1, 2, \ldots, n-p-1$$

b. If the pivot element $a^{(p)}_{n-p,k} = 0$, no transformations are necessary; that is, the (p+1)th similarity is the identity transformation. Otherwise, if it is necessary ($k \neq n-p-1$), we interchange the kth and (n-p-1)th columns so that the pivot element is in the subdiagonal position. The same interchange is applied to the rows of $A^{(p)}$. The resulting matrix is similar to $A^{(p)}$, and, to ease the notation, we denote it by $A^{(p)}$.

c. Define multipliers

$$b^{(p)}_j = a^{(p)}_{n-p,j} / a^{(p)}_{n-p,n-p-1} \qquad j=1,2,\ldots,n-p-2$$

Then the (p+1)th similarity is given by the following.

Right elimination:

$$\bar{a}^{(p)}_{ij} = a^{(p)}_{ij} - b^{(p)}_j \; a^{(p)}_{i,n-p-1}$$

$i=1,2,\ldots,n-p$
$j=1,2,\ldots,n-p-2$

$$\bar{a}^{(p)}_{ij} = a^{(p)}_{ij} \qquad \text{other indices}$$

Left inverse transformation:

$$a^{(p+1)}_{n-p-1,j} = \bar{a}^{(p)}_{n-p-1,j} + \sum_{i=1}^{n-p-2} b^{(p)}_i \, \bar{a}^{(p)}_{ij}$$

$$a_{ij}^{(p+1)} = \bar{a}_{ij}^{(p)} \qquad i \neq n-p-1$$

Finally, $A^{(n-2)}$ will have the upper almost-triangular form.

For reference see J. H. Wilkinson, The Algebraic Eigenvalue Problem. Clarendon Press, Oxford, 1965.

```
      ..................................................................
        SUBROUTINE HSBG
        PURPOSE
           TO REDUCE A REAL MATRIX INTO UPPER ALMOST TRIANGULAR FORM
        USAGE
           CALL HSBG(N,A,IA)
        DESCRIPTION OF THE PARAMETERS
           N       ORDER OF THE MATRIX
           A       THE INPUT MATRIX, N BY N
           IA      SIZE OF THE FIRST DIMENSION ASSIGNED TO THE ARRAY
                   A IN THE CALLING PROGRAM WHEN THE MATRIX IS IN
                   DOUBLE SUBSCRIPTED DATA STORAGE MODE.  IA=N WHEN
                   THE MATRIX IS IN SSP VECTOR STORAGE MODE.
        REMARKS
           THE HESSENBERG FORM REPLACES THE ORIGINAL MATRIX IN THE
           ARRAY A.
        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
           NONE
        METHOD
           SIMILARITY TRANSFORMATIONS USING ELEMENTARY ELIMINATION
           MATRICES, WITH PARTIAL PIVOTING.
        REFERENCES
           J.H. WILKINSON - THE ALGEBRAIC EIGENVALUE PROBLEM -
           CLARENDON PRESS, OXFORD, 1965.
      ..................................................................
      SUBROUTINE HSBG(N,A,IA)
      DIMENSION A(1)
      DOUBLE PRECISION S
      L=N
      NIA=L*IA
      LIA=NIA-IA
         L IS THE ROW INDEX OF THE ELIMINATION
   20 IF(L-3) 360,40,40
   40 LIA=LIA-IA
      L1=L-1
      L2=L1-1
         SEARCH FOR THE PIVOTAL ELEMENT IN THE LTH ROW
      ISUB=LIA+L
      IPIV=ISUB-IA
      PIV=ABS(A(IPIV))
      IF(L-3) 90,90,50
   50 M=IPIV-IA
      DO 80 I=L,M,IA
      T=ABS(A(I))
      IF(T-PIV) 80,80,60
   60 IPIV=I
      PIV=T
   80 CONTINUE
   90 IF(PIV) 100,320,100
  100 IF(PIV-ABS(A(ISUB))) 180,180,120
         INTERCHANGE THE COLUMNS
  120 M=IPIV-L
      DO 140 I=1,L
      J=M+I
      T=A(J)
      K=LIA+I
      A(J)=A(K)
  140 A(K)=T
         INTERCHANGE THE ROWS
      M=L2-M/IA
      DO 160 I=L1,NIA,IA
      T=A(I)
      J=I-M
      A(I)=A(J)
  160 A(J)=T
         TERMS OF THE ELEMENTARY TRANSFORMATION
  180 DO 200 I=L,LIA,IA
  200 A(I)=A(I)/A(ISUB)
         RIGHT TRANSFORMATION
      J=-IA
      DO 240 I=1,L2
      J=J+IA
      LJ=L+J
      DO 220 K=1,L1
      KJ=K+J
      KL=K+LIA
  220 A(KJ)=A(KJ)-A(LJ)*A(KL)
  240 CONTINUE
         LEFT TRANSFORMATION
      K=-IA
      DO 300 I=1,N
      K=K+IA
      LK=K+L1
      S=A(LK)
      LJ=L-IA
      DO 280 J=1,L2
      JK=K+J
      LJ=LJ+IA
  280 S=S+A(LJ)*A(JK)*1.0D0
  300 A(LK)=S
C
C        SET THE LOWER PART OF THE MATRIX TO ZERO
C
      DO 310 I=L,LIA,IA
  310 A(I)=0.0
  320 L=L1
      GO TO 20
  360 RETURN
      END
```

Subroutine PADD

```
C                                                                       PADD  10
C     ..................................................................PADD  20
C                                                                       PADD  30
C        SUBROUTINE PADD                                                PADD  40
C                                                                       PADD  50
C        PURPOSE                                                        PADD  60
C           ADD TWO POLYNOMIALS                                         PADD  70
C                                                                       PADD  80
C        USAGE                                                          PADD  90
C           CALL PADD(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          PADD 100
C                                                                       PADD 110
C        DESCRIPTION OF PARAMETERS                                      PADD 120
C           Z     - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM      PADD 130
C                   SMALLEST TO LARGEST POWER                           PADD 140
C           IDIMZ - DIMENSION OF Z (CALCULATED)                         PADD 150
C           X     - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL, ORDEREDPADD 160
C                   FROM SMALLEST TO LARGEST POWER                      PADD 170
C           IDIMX - DIMENSION OF X (DEGREE IS IDIMX-1)                  PADD 180
C           Y     - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,       PADD 190
C                   ORDERED FROM SMALLEST TO LARGEST POWER              PADD 200
C           IDIMY - DIMENSION OF Y (DEGREE IS IDIMY-1)                  PADD 210
C                                                                       PADD 220
C        REMARKS                                                        PADD 230
C           VECTOR Z MAY BE IN SAME LOCATION AS EITHER VECTOR X OR      PADD 240
C           VECTOR Y ONLY IF THE DIMENSION OF THAT VECTOR IS NOT LESS   PADD 250
C           THAN THE OTHER INPUT VECTOR                                 PADD 260
C           THE RESULTANT POLYNOMIAL MAY HAVE TRAILING ZERO COEFFICIENTSPADD 270
C                                                                       PADD 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PADD 290
C           NONE                                                        PADD 300
C                                                                       PADD 310
C        METHOD                                                         PADD 320
C           DIMENSION OF RESULTANT VECTOR IDIMZ IS CALCULATED AS THE    PADD 330
C           LARGER OF THE TWO INPUT VECTOR DIMENSIONS. CORRESPONDING    PADD 340
C           COEFFICIENTS ARE THEN ADDED TO FORM Z.                      PADD 350
C                                                                       PADD 360
C     ..................................................................PADD 370
C                                                                       PADD 380
      SUBROUTINE PADD(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          PADD 390
      DIMENSION Z(1),X(1),Y(1)                                          PADD 400
C                                                                       PADD 410
C     TEST DIMENSIONS OF SUMMANDS                                       PADD 420
C                                                                       PADD 430
      NDIM=IDIMX                                                        PADD 440
      IF (IDIMX-IDIMY) 10,20,20                                         PADD 450
   10 NDIM=IDIMY                                                        PADD 460
   20 IF(NDIM) 90,90,30                                                 PADD 470
   30 DO 80 I=1,NDIM                                                    PADD 480
      IF(I-IDIMX) 40,40,60                                              PADD 490
   40 IF(I-IDIMY) 50,50,70                                              PADD 500
   50 Z(I)=X(I)+Y(I)                                                    PADD 510
      GO TO 80                                                          PADD 520
   60 Z(I)=Y(I)                                                         PADD 530
      GO TO 80                                                          PADD 540
   70 Z(I)=X(I)                                                         PADD 550
   80 CONTINUE                                                          PADD 560
   90 IDIMZ=NDIM                                                        PADD 570
      RETURN                                                            PADD 580
      END                                                               PADD 590
```

Subroutine PSUB

```
C                                                                       PSUB  10
C     ..................................................................PSUB  20
C                                                                       PSUB  30
C        SUBROUTINE PSUB                                                PSUB  40
C                                                                       PSUB  50
C        PURPOSE                                                        PSUB  60
C           SUBTRACT ONE POLYNOMIAL FROM ANOTHER                        PSUB  70
C                                                                       PSUB  80
C        USAGE                                                          PSUB  90
C           CALL PSUB(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          PSUB 100
C                                                                       PSUB 110
C        DESCRIPTION OF PARAMETERS                                      PSUB 120
C           Z     - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM      PSUB 130
C                   SMALLEST TO LARGEST POWER                           PSUB 140
C           IDIMZ - DIMENSION OF Z (CALCULATED)                         PSUB 150
C           X     - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL, ORDEREDPSUB 160
C                   FROM SMALLEST TO LARGEST POWER                      PSUB 170
C           IDIMX - DIMENSION OF X (DEGREE IS IDIMX-1)                  PSUB 180
C           Y     - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,       PSUB 190
C                   ORDERED FROM SMALLEST TO LARGEST POWER              PSUB 200
C           IDIMY - DIMENSION OF Y (DEGREE IS IDIMY-1)                  PSUB 210
C                                                                       PSUB 220
C        REMARKS                                                        PSUB 230
C           VECTOR Z MAY BE IN SAME LOCATION AS EITHER VECTOR X OR      PSUB 240
C           VECTOR Y ONLY IF THE DIMENSION OF THAT VECTOR IS NOT LESS   PSUB 250
C           THAN THE OTHER INPUT VECTOR                                 PSUB 260
C           THE RESULTANT POLYNOMIAL MAY HAVE TRAILING ZERO COEFFICIENTSPSUB 270
C                                                                       PSUB 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PSUB 290
C           NONE                                                        PSUB 300
C                                                                       PSUB 310
C        METHOD                                                         PSUB 320
C           DIMENSION OF RESULTANT VECTOR IDIMZ IS CALCULATED AS THE    PSUB 330
C           LARGER OF THE TWO INPUT VECTOR DIMENSIONS. COEFFICIENTS IN  PSUB 340
C           VECTOR Y ARE THEN SUBTRACTED FROM CORRESPONDING COEFFICIENTSPSUB 350
C           IN VECTOR X.                                                PSUB 360
C                                                                       PSUB 370
C     ..................................................................PSUB 380
C                                                                       PSUB 390
      SUBROUTINE PSUB(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          PSUB 400
      DIMENSION Z(1),X(1),Y(1)                                          PSUB 410
C                                                                       PSUB 420
C     TEST DIMENSIONS OF SUMMANDS                                       PSUB 430
C                                                                       PSUB 440
      NDIM=IDIMX                                                        PSUB 450
      IF (IDIMX-IDIMY) 10,20,20                                         PSUB 460
   10 NDIM=IDIMY                                                        PSUB 470
   20 IF (NDIM) 90,90,30                                                PSUB 480
   30 DO 80 I=1,NDIM                                                    PSUB 490
      IF (I-IDIMX) 40,40,60                                             PSUB 500
   40 IF (I-IDIMY) 50,50,70                                             PSUB 510
   50 Z(I)=X(I)-Y(I)                                                    PSUB 520
      GO TO 80                                                          PSUB 530
   60 Z(I)=-Y(I)                                                        PSUB 540
      GO TO 80                                                          PSUB 550
   70 Z(I)=X(I)                                                         PSUB 560
   80 CONTINUE                                                          PSUB 570
   90 IDIMZ=NDIM                                                        PSUB 580
      RETURN                                                            PSUB 590
      END                                                               PSUB 600
```

Subroutine PMPY

```
C                                                                       PMPY  10
C     ..................................................................PMPY  20
C                                                                       PMPY  30
C        SUBROUTINE PMPY                                                PMPY  40
C                                                                       PMPY  50
C        PURPOSE                                                        PMPY  60
C           MULTIPLY TWO POLYNOMIALS                                    PMPY  70
C                                                                       PMPY  80
C        USAGE                                                          PMPY  90
C           CALL PMPY(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          PMPY 100
C                                                                       PMPY 110
C        DESCRIPTION OF PARAMETERS                                      PMPY 120
C           Z     - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM      PMPY 130
C                   SMALLEST TO LARGEST POWER                           PMPY 140
C           IDIMZ - DIMENSION OF Z (CALCULATED)                         PMPY 150
C           X     - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL, ORDERED PMPY 160
C                   FROM SMALLEST TO LARGEST POWER                      PMPY 170
C           IDIMX - DIMENSION OF X (DEGREE IS IDIMX-1)                  PMPY 180
C           Y     - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,       PMPY 190
C                   ORDERED FROM SMALLEST TO LARGEST POWER              PMPY 200
C           IDIMY - DIMENSION OF Y (DEGREE IS IDIMY-1)                  PMPY 210
C                                                                       PMPY 220
C        REMARKS                                                        PMPY 230
C           Z CANNOT BE IN THE SAME LOCATION AS X                       PMPY 240
C           Z CANNOT BE IN THE SAME LOCATION AS Y                       PMPY 250
C                                                                       PMPY 260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PMPY 270
C           NONE                                                        PMPY 280
C                                                                       PMPY 290
C        METHOD                                                         PMPY 300
C           DIMENSION OF Z IS CALCULATED AS IDIMX+IDIMY-1               PMPY 310
C           THE COEFFICIENTS OF Z ARE CALCULATED AS SUM OF PRODUCTS     PMPY 320
C           OF COEFFICIENTS OF X AND Y , WHOSE EXPONENTS ADD UP TO THE  PMPY 330
C           CORRESPONDING EXPONENT OF Z.                                PMPY 340
C                                                                       PMPY 350
C     ..................................................................PMPY 360
      SUBROUTINE PMPY(Z,IDIMZ,X,IDIMX,Y,IDIMY)                          PMPY 370
      DIMENSION Z(1),X(1),Y(1)                                          PMPY 380
C                                                                       PMPY 390
      IF(IDIMX*IDIMY)10,10,20                                           PMPY 400
   10 IDIMZ=0                                                           PMPY 410
      GO TO 50                                                          PMPY 420
   20 IDIMZ=IDIMX+IDIMY-1                                               PMPY 430
      DO 30 I=1,IDIMZ                                                   PMPY 440
   30 Z(I)=0.                                                           PMPY 450
      DO 40 I=1,IDIMX                                                   PMPY 460
      DO 40 J=1,IDIMY                                                   PMPY 470
      K=I+J-1                                                           PMPY 480
   40 Z(K)=X(I)*Y(J)+Z(K)                                               PMPY 490
   50 RETURN                                                            PMPY 500
      END                                                               PMPY 510
                                                                        PMPY 520
```

Subroutine PDIV

```
C                                                                       PDIV  10
C     ..................................................................PDIV  20
C                                                                       PDIV  30
C        SUBROUTINE PDIV                                                PDIV  40
C                                                                       PDIV  50
C        PURPOSE                                                        PDIV  60
C           DIVIDE ONE POLYNOMIAL BY ANOTHER                            PDIV  70
C                                                                       PDIV  80
C        USAGE                                                          PDIV  90
C           CALL PDIV(P,IDIMP,X,IDIMX,Y,IDIMY,TOL,IER)                  PDIV 100
C                                                                       PDIV 110
C        DESCRIPTION OF PARAMETERS                                      PDIV 120
C           P     - RESULTANT VECTOR OF INTEGRAL PART                   PDIV 130
C           IDIMP - DIMENSION OF P                                      PDIV 140
C           X     - VECTOR OF COEFFICIENTS FOR DIVIDEND POLYNOMIAL,     PDIV 150
C                   ORDERED FROM SMALLEST TO LARGEST POWER. IT IS       PDIV 160
C                   REPLACED BY REMAINDER AFTER DIVISION.               PDIV 170
C           IDIMX - DIMENSION OF X                                      PDIV 180
C           Y     - VECTOR OF COEFFICIENTS FOR DIVISOR POLYNOMIAL,      PDIV 190
C                   ORDERED FROM SMALLEST TO LARGEST POWER              PDIV 200
C           IDIMY - DIMENSION OF Y                                      PDIV 210
C           TOL   - TOLERANCE VALUE BELOW WHICH COEFFICIENTS ARE        PDIV 220
C                   ELIMINATED DURING NORMALIZATION                     PDIV 230
C           IER   - ERROR CODE. 0 IS NORMAL, 1 IS FOR ZERO DIVISOR      PDIV 240
C                                                                       PDIV 250
C        REMARKS                                                        PDIV 260
C           THE REMAINDER R REPLACES X.                                 PDIV 270
C           THE DIVISOR Y REMAINS UNCHANGED.                            PDIV 280
C           IF DIMENSION OF Y EXCEEDS DIMENSION OF X, IDIMP IS SET TO   PDIV 290
C           ZERO AND CALCULATION IS BYPASSED                            PDIV 300
C                                                                       PDIV 310
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PDIV 320
C           PNORM                                                       PDIV 330
C                                                                       PDIV 340
C        METHOD                                                         PDIV 350
C           POLYNOMIAL X IS DIVIDED BY POLYNOMIAL Y GIVING INTEGER PART PDIV 360
C           P AND REMAINDER R SUCH THAT X = P*Y + R.                    PDIV 370
C           DIVISOR Y AND REMAINDER  VECTOR GET NORMALIZED.             PDIV 380
C                                                                       PDIV 390
C     ..................................................................PDIV 400
C                                                                       PDIV 410
      SUBROUTINE PDIV(P,IDIMP,X,IDIMX,Y,IDIMY,TOL,IER)                  PDIV 420
      DIMENSION P(1),X(1),Y(1)                                          PDIV 430
C                                                                       PDIV 440
      CALL PNORM (Y,IDIMY,TOL)                                          PDIV 450
      IF(IDIMY) 50,50,10                                                PDIV 460
   10 IDIMP=IDIMX-IDIMY+1                                               PDIV 470
      IF(IDIMP) 20,30,60                                                PDIV 480
C                                                                       PDIV 490
C     DEGREE OF DIVISOR WAS GREATER THAN DEGREE OF DIVIDEND             PDIV 500
C                                                                       PDIV 510
   20 IDIMP=0                                                           PDIV 520
   30 IER=0                                                             PDIV 530
   40 RETURN                                                            PDIV 540
C                                                                       PDIV 550
C     Y IS ZERO POLYNOMIAL                                              PDIV 560
C                                                                       PDIV 570
   50 IER=1                                                             PDIV 580
      GO TO 40                                                          PDIV 590
C                                                                       PDIV 600
C     START REDUCTION                                                   PDIV 610
C                                                                       PDIV 620
   60 IDIMX=IDIMY-1                                                     PDIV 630
      I=IDIMP                                                           PDIV 640
   70 II=I+IDIMX                                                        PDIV 650
      P(I)=X(II)/Y(IDIMY)                                               PDIV 660
C                                                                       PDIV 670
C     SUBTRACT MULTIPLE OF DIVISOR                                      PDIV 680
C                                                                       PDIV 690
      DO 80 K=1,IDIMX                                                   PDIV 700
      J=K-1+I                                                           PDIV 710
      X(J)=X(J)-P(I)*Y(K)                                               PDIV 720
   80 CONTINUE                                                          PDIV 730
      I=I-1                                                             PDIV 740
      IF(I) 90,90,70                                                    PDIV 750
C                                                                       PDIV 760
C     NORMALIZE REMAINDER POLYNOMIAL                                    PDIV 770
C                                                                       PDIV 780
   90 CALL PNORM(X,IDIMX,TOL)                                           PDIV 790
      GO TO 30                                                          PDIV 800
      END                                                               PDIV 810
```

Subroutine PCLA

```
C                                                                       PCLA  10
C     ...................................................................PCLA  20
C                                                                       PCLA  30
C        SUBROUTINE PCLA                                                PCLA  40
C                                                                       PCLA  50
C        PURPOSE                                                        PCLA  60
C           MOVE POLYNOMIAL X TO Y                                      PCLA  70
C                                                                       PCLA  80
C        USAGE                                                          PCLA  90
C           CALL PCLA(Y,IDIMY,X,IDIMX)                                  PCLA 100
C                                                                       PCLA 110
C        DESCRIPTION OF PARAMETERS                                      PCLA 120
C           Y     - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM      PCLA 130
C                   SMALLEST TO LARGEST POWER                           PCLA 140
C           IDIMY - DIMENSION OF Y                                      PCLA 150
C           X     - VECTOR OF COEFFICIENTS FOR POLYNOMIAL, ORDERED      PCLA 160
C                   FROM SMALLEST TO LARGEST POWER                      PCLA 170
C           IDIMX - DIMENSION OF X                                      PCLA 180
C                                                                       PCLA 190
C        REMARKS                                                        PCLA 200
C           NONE                                                        PCLA 210
C                                                                       PCLA 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PCLA 230
C           NONE                                                        PCLA 240
C                                                                       PCLA 250
C        METHOD                                                         PCLA 260
C           IDIMY IS REPLACED BY IDIMX AND VECTOR X IS MOVED TO Y       PCLA 270
C                                                                       PCLA 280
C     ...................................................................PCLA 290
C                                                                       PCLA 300
      SUBROUTINE PCLA (Y,IDIMY,X,IDIMX)                                 PCLA 310
      DIMENSION X(1),Y(1)                                               PCLA 320
C                                                                       PCLA 330
      IDIMY=IDIMX                                                       PCLA 340
      IF(IDIMX) 30,30,10                                                PCLA 350
   10 DO 20 I=1,IDIMX                                                   PCLA 360
   20 Y(I)=X(I)                                                         PCLA 370
   30 RETURN                                                            PCLA 380
      END                                                               PCLA 390
```

Subroutine PADDM

```
C                                                                       PDDM  10
C     ...................................................................PDDM  20
C                                                                       PDDM  30
C        SUBROUTINE PADDM                                               PDDM  40
C                                                                       PDDM  50
C        PURPOSE                                                        PDDM  60
C           ADD COEFFICIENTS OF ONE POLYNOMIAL TO THE PRODUCT OF A      PDDM  70
C           FACTOR BY COEFFICIENTS OF ANOTHER POLYNOMIAL                PDDM  80
C                                                                       PDDM  90
C        USAGE                                                          PDDM 100
C           CALL PADDM(Z,IDIMZ,X,IDIMX,FACT,Y,IDIMY)                    PDDM 110
C                                                                       PDDM 120
C        DESCRIPTION OF PARAMETERS                                      PDDM 130
C           Z     - VECTOR OF RESULTANT COEFFICIENTS, ORDERED FROM      PDDM 140
C                   SMALLEST TO LARGEST POWER                           PDDM 150
C           IDIMZ - DIMENSION OF Z (CALCULATED)                         PDDM 160
C           X     - VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL, ORDERED PDDM 170
C                   FROM SMALLEST TO LARGEST POWER                      PDDM 180
C           IDIMX - DIMENSION OF X (DEGREE IS IDIMX-1)                  PDDM 190
C           FACT  - FACTOR TO BE MULTIPLIED BY VECTOR Y                 PDDM 200
C           Y     - VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,       PDDM 210
C                   ORDERED FROM SMALLEST TO LARGEST POWER              PDDM 220
C           IDIMY - DIMENSION OF Y (DEGREE IS IDIMY-1)                  PDDM 230
C                                                                       PDDM 240
C        REMARKS                                                        PDDM 250
C           VECTOR Z MAY BE IN SAME LOCATION AS EITHER VECTOR X OR      PDDM 260
C           VECTOR Y ONLY IF THE DIMENSION OF THAT VECTOR IS NOT LESS   PDDM 270
C           THAN THE OTHER INPUT VECTOR                                 PDDM 280
C           THE RESULTANT POLYNOMIAL MAY HAVE TRAILING ZERO COEFFICIENTSPDDM 290
C                                                                       PDDM 300
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PDDM 310
C           NONE                                                        PDDM 320
C                                                                       PDDM 330
C        METHOD                                                         PDDM 340
C           DIMENSION OF RESULTANT VECTOR IDIMZ IS CALCULATED AS THE    PDDM 350
C           LARGER OF THE TWO INPUT VECTOR DIMENSIONS. COEFFICIENT IN   PDDM 360
C           VECTOR X IS THEN ADDED TO COEFFICIENT IN VECTOR Y MULTIPLIEDPDDM 370
C           BY FACTOR TO FORM Z.                                        PDDM 380
C                                                                       PDDM 390
C     ...................................................................PDDM 400
C                                                                       PDDM 410
      SUBROUTINE PADDM(Z,IDIMZ,X,IDIMX,FACT,Y,IDIMY)                    PDDM 420
      DIMENSION Z(1),X(1),Y(1)                                          PDDM 430
C                                                                       PDDM 440
C     TEST DIMENSIONS OF SUMMANDS                                       PDDM 450
C                                                                       PDDM 460
      NDIM=IDIMX                                                        PDDM 470
      IF(IDIMX-IDIMY) 10,20,20                                          PDDM 480
   10 NDIM=IDIMY                                                        PDDM 490
   20 IF(NDIM) 90,90,30                                                 PDDM 500
   30 DO 80 I=1,NDIM                                                    PDDM 510
      IF(I-IDIMX) 40,40,60                                              PDDM 520
   40 IF(I-IDIMY) 50,50,70                                              PDDM 530
   50 Z(I)=FACT*Y(I)+X(I)                                               PDDM 540
      GO TO 80                                                          PDDM 550
   60 Z(I)=FACT*Y(I)                                                    PDDM 560
      GO TO 80                                                          PDDM 570
   70 Z(I)=X(I)                                                         PDDM 580
   80 CONTINUE                                                          PDDM 590
   90 IDIMZ=NDIM                                                        PDDM 600
      RETURN                                                            PDDM 610
      END                                                               PDDM 620
```

Subroutine PVAL

```
*********************************************************************
      SUBROUTINE PVAL
      PURPOSE
         EVALUATE A POLYNOMIAL FOR A GIVEN VALUE OF THE VARIABLE
      USAGE
         CALL PVAL(RES,ARG,X,IDIMX)
      DESCRIPTION OF PARAMETERS
         RES    - RESULTANT VALUE OF POLYNOMIAL
         ARG    - GIVEN VALUE OF THE VARIABLE
         X      - VECTOR OF COEFFICIENTS, ORDERED FROM SMALLEST TO
                  LARGEST POWER
         IDIMX  - DIMENSION OF X
      REMARKS
         NONE
      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
         NONE
      METHOD
         EVALUATION IS DONE BY MEANS OF NESTED MULTIPLICATION
*********************************************************************
      SUBROUTINE PVAL(RES,ARG,X,IDIMX)
      DIMENSION X(1)
      RES=0.
      J=IDIMX
    1 IF(J)3,3,2
    2 RES=RES*ARG+X(J)
      J=J-1
      GO TO 1
    3 RETURN
      END
```

Subroutine PVSUB

```
C
C     ******************************************************************
C
C        SUBROUTINE PVSUB
C
C        PURPOSE
C           SUBSTITUTE VARIABLE OF A POLYNOMIAL BY ANOTHER POLYNOMIAL
C
C        USAGE
C           CALL PVSUB(Z,ICIMZ,X,ICIMX,Y,ICIMY,WORK1,WORK2)
C
C        DESCRIPTION OF PARAMETERS
C           Z      - VECTOR OF COEFFICIENTS FOR RESULTANT POLYNOMIAL,
C                    ORDERED FROM SMALLEST TO LARGEST POWER
C           ICIMZ  - DIMENSION OF Z
C           X      - VECTOR OF COEFFICIENTS FOR ORIGINAL POLYNOMIAL,
C                    ORDERED FROM SMALLEST TO LARGEST POWER
C           ICIMX  - DIMENSION OF X
C           Y      - VECTOR OF COEFFICIENTS FOR POLYNOMIAL WHICH IS
C                    SUBSTITUTED FOR VARIABLE, ORDERED FROM SMALLEST TO
C                    LARGEST POWER
C           ICIMY  - DIMENSION OF Y
C           WORK1  - WORKING STORAGE ARRAY (SAME DIMENSION AS Z)
C           WORK2  - WORKING STORAGE ARRAY (SAME DIMENSION AS Z)
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           FMPY
C           FADDM
C           PCLA
C
C        METHOD
C           VARIABLE OF POLYNOMIAL X IS SUBSTITUTED BY POLYNOMIAL Y
C           TO FORM POLYNOMIAL Z. DIMENSION OF NEW POLYNOMIAL IS
C           (ICIMX-1)*(ICIMY-1)+1. SUBROUTINE REQUIRES THE WORK AREAS
C
C     ******************************************************************
C
      SUBROUTINE PVSUB(Z,ICIMZ,X,ICIMX,Y,ICIMY,WORK1,WORK2)
      DIMENSION Z(1),X(1),Y(1),WORK1(1),WORK2(1)
C
C     TEST OF DIMENSIONS
C
      IF (ICIMX-1) 1,3,3
    1 ICIMZ=0
    2 RETURN
C
    3 ICIMZ=1
      Z(1)=X(1)
      IF (ICIMY*ICIMX-ICIMY) 2,2,4
    4 IK1=1
      WORK1(1)=1.
C
      DO 5 I=2,ICIMX
      CALL FMPY(WORK2,IK2,Y,ICIMY,WORK1,IK1)
      CALL PCLA(WORK1,IK1,WORK2,IK2)
      FACT=X(I)
      CALL FADDM(Z,ICIMR,Z,ICIMZ,FACT,WORK1,IK1)
      ICIMZ=ICIMR
    5 CONTINUE
      GO TO 2
      END
```

Subroutine PILD

```
C                                                                       PILD  10
C     ...................................................................PILD  20
C
C        SUBROUTINE PILD                                                PILD  30
C                                                                       PILD  40
C        PURPOSE                                                        PILD  50
C           EVALUATE POLYNOMIAL AND ITS FIRST DERIVATIVE FOR A GIVEN    PILD  60
C           ARGUMENT                                                    PILD  70
C                                                                       PILD  80
C        USAGE                                                          PILD  90
C           CALL PILD(POLY,DVAL,ARGUM,X,IDIMX)                          PILD 100
C                                                                       PILD 110
C        DESCRIPTION OF PARAMETERS                                      PILD 120
C           POLY  - VALUE OF POLYNOMIAL                                 PILD 130
C           DVAL  - DERIVATIVE                                          PILD 140
C           ARGUM - ARGUMENT                                            PILD 150
C           X     - VECTOR OF COEFFICIENTS FOR POLYNOMIAL, ORDERED      PILD 160
C                   FROM SMALLEST TO LARGEST POWER                      PILD 170
C           IDIMX - DIMENSION OF X                                      PILD 180
C                                                                       PILD 190
C        REMARKS                                                        PILD 200
C           NONE                                                        PILD 210
C                                                                       PILD 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PILD 230
C           PQSD                                                        PILD 240
C                                                                       PILD 250
C        METHOD                                                         PILD 260
C           EVALUATION IS DONE BY MEANS OF SUBROUTINE PQSD (QUADRATIC   PILD 270
C           SYNTHETIC DIVISION)                                         PILD 280
C                                                                       PILD 290
C                                                                       PILD 300
C     ...................................................................PILD 310
C                                                                       PILD 320
      SUBROUTINE PILD (POLY,DVAL,ARGUM,X,IDIMX)                         PILD 330
      DIMENSION X(1)                                                    PILD 340
C                                                                       PILD 350
      P=ARGUM+ARGUM                                                     PILD 360
      Q=-ARGUM*ARGUM                                                    PILD 370
C                                                                       PILD 380
      CALL PQSD (DVAL,POLY,P,Q,X,IDIMX)                                 PILD 390
C                                                                       PILD 400
      POLY=ARGUM*DVAL+POLY                                              PILD 410
C                                                                       PILD 420
      RETURN                                                            PILD 430
      END                                                               PILD 440
```

Subroutine PDER

```
C                                                                       PDER  10
C     ...................................................................PDER  20
C                                                                       PDER  30
C        SUBROUTINE PDER                                                PDER  40
C                                                                       PDER  50
C        PURPOSE                                                        PDER  60
C           FIND DERIVATIVE OF A POLYNOMIAL                             PDER  70
C                                                                       PDER  80
C        USAGE                                                          PDER  90
C           CALL PDER(Y,IDIMY,X,IDIMX)                                  PDER 100
C                                                                       PDER 110
C        DESCRIPTION OF PARAMETERS                                      PDER 120
C           Y     - VECTOR OF COEFFICIENTS FOR DERIVATIVE, ORDERED FROM PDER 130
C                   SMALLEST TO LARGEST POWER                           PDER 140
C           IDIMY - DIMENSION OF Y (EQUAL TO IDIMX-1)                   PDER 150
C           X     - VECTOR OF COEFFICIENTS FOR ORIGINAL POLYNOMIAL,     PDER 160
C                   ORDERED FROM SMALLEST TO LARGEST POWER              PDER 170
C           IDIMX - DIMENSION OF X                                      PDER 180
C                                                                       PDER 190
C        REMARKS                                                        PDER 200
C           NONE                                                        PDER 210
C                                                                       PDER 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PDER 230
C           NONE                                                        PDER 240
C                                                                       PDER 250
C        METHOD                                                         PDER 260
C           DIMENSION OF Y IS SET AT DIMENSION OF X LESS ONE. DERIVATIVEPDER 270
C           IS THEN CALCULATED BY MULTIPLYING COEFFICIENTS BY THEIR     PDER 280
C           RESPECTIVE EXPONENTS.                                       PDER 290
C                                                                       PDER 300
C     ...................................................................PDER 310
C                                                                       PDER 320
      SUBROUTINE PDER(Y,IDIMY,X,IDIMX)                                  PDER 330
      DIMENSION X(1),Y(1)                                               PDER 340
C                                                                       PDER 350
C     TEST OF DIMENSION                                                 PDER 360
      IF (IDIMX-1) 3,3,1                                                PDER 370
    1 IDIMY=IDIMX-1                                                     PDER 380
      EXPT=0.                                                           PDER 390
      DO 2 I=1,IDIMY                                                    PDER 400
      EXPT=EXPT+1.                                                      PDER 410
    2 Y(I)=X(I+1)*EXPT                                                  PDER 420
      GO TO 4                                                           PDER 430
    3 IDIMY=0                                                           PDER 440
    4 RETURN                                                            PDER 450
      END                                                               PDER 460
```

Subroutine PINT

```
C
C     ..................................................................
C
C        SUBROUTINE PINT
C
C        PURPOSE
C           FIND INTEGRAL OF A POLYNOMIAL WITH CONSTANT OF INTEGRATION
C           EQUAL TO ZERO
C
C        USAGE
C           CALL PINT(Y,IDIMY,X,IDIMX)
C
C        DESCRIPTION OF PARAMETERS
C           Y      - VECTOR OF COEFFICIENTS FOR INTEGRAL, ORDERED FROM
C                    SMALLEST TO LARGEST POWER
C           IDIMY - DIMENSION OF Y (EQUAL TO IDIMX+1)
C           X      - VECTOR OF COEFFICIENTS FOR ORIGINAL POLYNOMIAL,
C                    ORDERED FROM SMALLEST TO LARGEST POWER
C           IDIMX - DIMENSION OF X
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DIMENSION OF Y IS SET AT DIMENSION OF X PLUS ONE, AND THE
C           CONSTANT TERM IS SET TO ZERO. INTEGRAL IS THEN CALCULATED
C           BY DIVIDING COEFFICIENTS BY THEIR RESPECTIVE EXPONENTS.
C
C     ..................................................................
      SUBROUTINE PINT(Y,IDIMY,X,IDIMX)
      DIMENSION X(1),Y(1)
C
      IDIMY=IDIMX+1
      Y(1)=0.
      IF(IDIMX)1,1,2
    1 RETURN
    2 EXPT=1.
      DO 3 I=2,IDIMY
      Y(I)=X(I-1)/EXPT
    3 EXPT=EXPT+1.
      GO TO 1
      END
```

Subroutine PQSD

```
C
C     ..................................................................
C
C        SUBROUTINE PQSD
C
C        PURPOSE
C           PERFORM QUADRATIC SYNTHETIC DIVISION
C
C        USAGE
C           CALL PQSD(A,B,P,Q,X,IDIMX)
C
C        DESCRIPTION OF PARAMETERS
C           A      - COEFFICIENT OF Z IN REMAINDER (CALCULATED)
C           B      - CONSTANT TERM IN REMAINDER (CALCULATED)
C           P      - COEFFICIENT OF Z IN QUADRATIC POLYNOMIAL
C           Q      - CONSTANT TERM IN QUADRATIC POLYNOMIAL
C           X      - COEFFICIENT VECTOR FOR GIVEN POLYNOMIAL, ORDERED
C                    FROM SMALLEST TO LARGEST POWER
C           IDIMX - DIMENSION OF X
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           POLYNOMIAL IS DIVIDED BY THE QUADRATIC Z**2-P*Z-Q GIVING
C           THE LINEAR REMAINDER A*Z+B
C
C     ..................................................................
      SUBROUTINE PQSD(A,B,P,Q,X,IDIMX)
      DIMENSION X(1)
C
      A=0.
      B=0.
      J=IDIMX
    1 IF(J)3,3,2
    2 Z=P*A+B
      B=Q*A+X(J)
      A=Z
      J=J-1
      GO TO 1
    3 RETURN
      END
```

Subroutine PCLD

```
C                                                                       PCLD  10
C     ...................................................................PCLD  20
C                                                                       PCLD  30
C        SUBROUTINE PCLD                                                PCLD  40
C                                                                       PCLD  50
C        PURPOSE                                                        PCLD  60
C           SHIFT OF ORIGIN (COMPLETE LINEAR SYNTHETIC DIVISION)        PCLD  70
C                                                                       PCLD  80
C        USAGE                                                          PCLD  90
C           CALL PCLD(X,IDIMX,U)                                        PCLD 100
C                                                                       PCLD 110
C        DESCRIPTION OF PARAMETERS                                      PCLD 120
C           X      - VECTOR OF COEFFICIENTS, ORDERED FROM SMALLEST TO   PCLD 130
C                    LARGEST POWER. IT IS REPLACED BY VECTOR OF         PCLD 140
C                    TRANSFORMED COEFFICIENTS.                          PCLD 150
C           IDIMX - DIMENSION OF X                                      PCLD 160
C           U     - SHIFT PARAMETER                                     PCLD 170
C                                                                       PCLD 180
C        REMARKS                                                        PCLD 190
C           NONE                                                        PCLD 200
C                                                                       PCLD 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PCLD 220
C           NONE                                                        PCLD 230
C                                                                       PCLD 240
C        METHOD                                                         PCLD 250
C           COEFFICIENT VECTOR X(I) OF POLYNOMIAL P(Z) IS TRANSFORMED   PCLD 260
C           SUCH THAT Q(Z)=P(Z-U) WHERE Q(Z) DENOTES THE POLYNOMIAL     PCLD 270
C           WITH TRANSFORMED COEFFICIENT VECTOR.                        PCLD 280
C                                                                       PCLD 290
C     ...................................................................PCLD 300
C                                                                       PCLD 310
      SUBROUTINE PCLD (X,IDIMX,U)                                       PCLD 320
      DIMENSION X(1)                                                    PCLD 330
C                                                                       PCLD 340
      K=1                                                               PCLD 350
    1 J=IDIMX                                                           PCLD 360
    2 IF (J-K) 4,4,3                                                    PCLD 370
    3 X(J-1)=X(J-1)+U*X(J)                                              PCLD 380
      J=J-1                                                             PCLD 390
      GO TO 2                                                           PCLD 400
    4 K=K+1                                                             PCLD 410
      IF (IDIMX-K) 5,5,1                                                PCLD 420
    5 RETURN                                                            PCLD 430
      END                                                               PCLD 440
```

Subroutine PGCD

```
C                                                                       PGCD  10
C     ...................................................................PGCD  20
C                                                                       PGCD  30
C        SUBROUTINE PGCD                                                PGCD  40
C                                                                       PGCD  50
C        PURPOSE                                                        PGCD  60
C           DETERMINE GREATEST COMMON DIVISOR OF TWO POLYNOMIALS        PGCD  70
C                                                                       PGCD  80
C        USAGE                                                          PGCD  90
C           CALL PGCD(X,IDIMX,Y,IDIMY,WORK,EPS,IER)                     PGCD 100
C                                                                       PGCD 110
C        DESCRIPTION OF PARAMETERS                                      PGCD 120
C           X      -  VECTOR OF COEFFICIENTS FOR FIRST POLYNOMIAL,      PGCD 130
C                     ORDERED FROM SMALLEST TO LARGEST POWER            PGCD 140
C           IDIMX -  DIMENSION OF X                                     PGCD 150
C           Y      -  VECTOR OF COEFFICIENTS FOR SECOND POLYNOMIAL,     PGCD 160
C                     ORDERED FROM SMALLEST TO LARGEST POWER.           PGCD 170
C                     THIS IS REPLACED BY GREATEST COMMON DIVISOR       PGCD 180
C           IDIMY -  DIMENSION OF Y                                     PGCD 190
C           WORK  -  WORKING STORAGE ARRAY                              PGCD 200
C           EPS   -  TOLERANCE VALUE BELOW WHICH COEFFICIENT IS         PGCD 210
C                    ELIMINATED DURING NORMALIZATION                    PGCD 220
C           IER   -  RESULTANT ERROR CODE WHERE                         PGCD 230
C                    IER=0  NO ERROR                                    PGCD 240
C                    IER=1  X OR Y IS ZERO POLYNOMIAL                   PGCD 250
C                                                                       PGCD 260
C        REMARKS                                                        PGCD 270
C           IDIMX MUST BE GREATER THAN IDIMY                            PGCD 280
C           IDIMY=1 ON RETURN MEANS X AND Y ARE PRIME, THE GCD IS A     PGCD 290
C           CONSTANT. IDIMX IS DESTROYED DURING COMPUTATION.            PGCD 300
C                                                                       PGCD 310
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PGCD 320
C           PDIV                                                        PGCD 330
C           PNORM                                                       PGCD 340
C                                                                       PGCD 350
C        METHOD                                                         PGCD 360
C           GREATEST COMMON DIVISOR OF TWO POLYNOMIALS X AND Y IS       PGCD 370
C           DETERMINED BY MEANS OF EUCLIDEAN ALGORITHM. COEFFICIENT     PGCD 380
C           VECTORS X AND Y ARE DESTROYED AND GREATEST COMMON           PGCD 390
C           DIVISOR IS GENERATED IN Y.                                  PGCD 400
C                                                                       PGCD 410
C     ...................................................................PGCD 420
C                                                                       PGCD 430
      SUBROUTINE PGCD(X,IDIMX,Y,IDIMY,WORK,EPS,IER)                     PGCD 440
      DIMENSION X(1),Y(1),WORK(1)                                       PGCD 450
C                                                                       PGCD 460
C     DIMENSION REQUIRED FOR VECTOR NAMED  WORK  IS   IDIMX-IDIMY+1     PGCD 470
C                                                                       PGCD 480
    1 CALL PDIV(WORK,NDIM,X,IDIMX,Y,IDIMY,EPS,IER)                      PGCD 490
      IF(IER) 5,2,5                                                     PGCD 500
    2 IF(IDIMX) 5,5,3                                                   PGCD 510
C                                                                       PGCD 520
C     INTERCHANGE X AND Y                                               PGCD 530
C                                                                       PGCD 540
    3 DO 4 J=1,IDIMY                                                    PGCD 550
      WORK(1)=X(J)                                                      PGCD 560
      X(J)=Y(J)                                                         PGCD 570
    4 Y(J)=WORK(1)                                                      PGCD 580
      NDIM=IDIMX                                                        PGCD 590
      IDIMX=IDIMY                                                       PGCD 600
      IDIMY=NDIM                                                        PGCD 610
      GO TO 1                                                           PGCD 620
    5 RETURN                                                            PGCD 630
      END                                                               PGCD 640
```

Subroutine PNORM

```
C                                                                       PNOR  10
C     ..................................................................PNOR  20
C                                                                       PNOR  30
C        SUBROUTINE PNORM                                               PNOR  40
C                                                                       PNOR  50
C        PURPOSE                                                        PNOR  60
C           NORMALIZE COEFFICIENT VECTOR OF A POLYNOMIAL                PNOR  70
C                                                                       PNOR  80
C        USAGE                                                          PNOR  90
C           CALL PNORM(X,IDIMX,EPS)                                     PNOR 100
C                                                                       PNOR 110
C        DESCRIPTION OF PARAMETERS                                      PNOR 120
C           X      - VECTOR OF ORIGINAL COEFFICIENTS, ORDERED FROM      PNOR 130
C                    SMALLEST TO LARGEST POWER. IT REMAINS UNCHANGED    PNOR 140
C           IDIMX  - DIMENSION OF X. IT IS REPLACED BY FINAL DIMENSION  PNOR 150
C           EPS    - TOLERANCE BELOW WHICH COEFFICIENT IS ELIMINATED    PNOR 160
C                                                                       PNOR 170
C        REMARKS                                                        PNOR 180
C           IF ALL COEFFICIENTS ARE LESS THAN EPS, RESULT IS A ZERO     PNOR 190
C           POLYNOMIAL WITH IDIMX=0 BUT VECTOR X REMAINS INTACT         PNOR 200
C                                                                       PNOR 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PNOR 220
C           NONE                                                        PNOR 230
C                                                                       PNOR 240
C        METHOD                                                         PNOR 250
C           DIMENSION OF VECTOR X IS REDUCED BY ONE FOR EACH TRAILING   PNOR 260
C           COEFFICIENT WITH AN ABSOLUTE VALUE LESS THAN OR EQUAL TO EPSPNOR 270
C                                                                       PNOR 280
C     ..................................................................PNOR 290
C                                                                       PNOR 300
      SUBROUTINE PNORM(X,IDIMX,EPS)                                     PNOR 310
      DIMENSION X(1)                                                    PNOR 320
C                                                                       PNOR 330
    1 IF(IDIMX) 4,4,2                                                   PNOR 340
    2 IF(ABS(X(IDIMX))-EPS) 3,3,4                                       PNOR 350
    3 IDIMX=IDIMX-1                                                     PNOR 360
      GO TO 1                                                           PNOR 370
    4 RETURN                                                            PNOR 380
      END                                                               PNOR 390
```

Subroutines PECN and DPECN

These subroutines perform the economization of a polynomial for symmetric range.

A given polynomial P(x) with coefficient vector $P = (p_1, \ldots, p_n)$, representing a function f(x) with a maximum error eps over the range (-bound, bound) -- that is, $\left| f(x) - \sum_{i=1}^{n} p_i x^{i-1} \right| \leqq \text{eps}$ for $|x| \leqq$ bound -- is reduced to a shorter coefficient vector, if possible, representing f(x) over the same range with a maximal error absolutely less than tol by means of telescoping.

The Chebyshev polynomial $T_{n-1}(t)$ has 2^{n-2} as coefficient of t^{n-1}. As a consequence, t^{n-1} may be replaced by $T_{n-1}(t)/2^{n-2}$ plus a polynomial in t of degree n-2.

This means that the telescoped polynomial:

$$P_1(x) = \sum_{i=1}^{n} p_i x^{i-1} - (p_n \cdot \text{bound}^{n-1} / 2^{n-2}) \; T_{n-1}(x/\text{bound})$$

is of degree n - 2 representing f(x) over the range $|x| \leqq$ bound with a maximal error err absolutely less than $\text{eps}_1 = \text{eps} + |p_n| \cdot \text{bound}^{n-1}/2^{n-2}$, since $|T_{n-1}(t)| \leqq 1$ for $|t| \leqq 1$.

If eps_1 is less than the tolerance tol, the procedure may be repeated with P_1 instead of P, and eps_1 instead of eps.

This iterative scheme results in a polynomial approximation to f(x) which possibly consists of fewer terms and thus may allow the computation of f(x) with less effort than the original approximation P(x).

Naturally, tol should be greater than eps.

For reference see K.A. Brons, Algorithm 38, Telescope 2, CACM vol. 4, 1961, no. 3, pp. 151-152.

```
C
C     ..................................................................
C
C        SUBROUTINE PECN
C
C        PURPOSE
C           ECONOMIZE A POLYNOMIAL FOR SYMMETRIC RANGE
C
C        USAGE
C           CALL PECN (P,N,BOUND,EPS,TOL,WORK)
C
C        DESCRIPTION OF PARAMETERS
C           P     - COEFFICIENT VECTOR OF GIVEN POLYNOMIAL
C                   ON RETURN P CONTAINS THE ECONOMIZED POLYNOMIAL
C           N     - DIMENSION OF COEFFICIENT VECTOR P
C                   ON RETURN N CONTAINS DIMENSION OF ECONOMIZED
C                   POLYNOMIAL
C           BOUND - RIGHT HAND BOUNDARY OF RANGE
C           EPS   - INITIAL ERROR BOUND
C                   ON RETURN EPS CONTAINS AN ERROR BOUND FOR THE
C                   ECONOMIZED POLYNOMIAL
C           TOL   - TOLERANCE FOR ERROR
C                   FINAL VALUE OF EPS MUST BE LESS THAN TOL
C           WORK  - WORKING STORAGE OF DIMENSION N (STARTING VALUE
C                   OF N RATHER THAN FINAL VALUE)
C
C        REMARKS
C           THE OPERATION IS BYPASSED IN CASE OF N LESS THAN 1.
C           IN CASE OF AN ARBITRARY INTERVAL (XL,XR) IT IS NECESSARY
C           FIRST TO CALCULATE THE EXPANSION OF THE GIVEN POLYNOMIAL
C           WITH ARGUMENT X IN POWERS OF T = (X-(XR-XL)/2).
C           THIS IS ACCOMPLISHED THROUGH SUBROUTINE PCLD.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SUBROUTINE PECN TAKES AN (N-1)ST DEGREE POLYNOMIAL
C           APPROXIMATION TO A FUNCTION F(X) VALID WITHIN A TOLERANCE
C           EPS OVER THE INTERVAL (-BOUND,BOUND) AND REDUCES IT IF
C           POSSIBLE TO A POLYNOMIAL OF LOWER DEGREE VALID WITHIN
C           THE GIVEN TOLERANCE TOL.
C           THE INITIAL COEFFICIENT VECTOR P IS REPLACED BY THE FINAL
C           VECTOR. THE INITIAL ERROR BOUND EPS IS REPLACED BY A FINAL
C           ERROR BOUND.
C           N IS REPLACED BY THE DIMENSION OF THE REDUCED POLYNOMIAL.
C           THE COEFFICIENT VECTOR OF THE N-TH CHEBYSHEV POLYNOMIAL
C           IS CALCULATED FROM THE RECURSION FORMULA
C           A(K-1)=-A(K+1)*K*L*L*(K-1)/((N+K-2)*(N-K+2))
C           REFERENCE
C           K. A. BRONS, ALGORITHM 38, TELESCOPE 2, CACM VOL. 4, 1961,
C           NO. 3, PP. 151-152.
C
C     ..................................................................
C
      SUBROUTINE PECN(P,N,BOUND,EPS,TOL,WORK)
C
      DIMENSION P(1),WORK(1)
      FL=BOUND*BOUND
C
C     TEST OF DIMENSION
C
    1 IF(N-1)2,3,6
    2 RETURN
    3 IF(EPS+ABS(P(1))-TOL)4,4,5
    4 N=0
      EPS=EPS+ABS(P(1))
    5 RETURN
C
C     CALCULATE EXPANSION OF CHEBYSHEV POLYNOMIAL
C
    6 NEND=N-2
      WORK(N)=-P(N)
      DO 7 J=1,NEND,2
      K=N-J
      FN=(NEND-1+K)*(NEND+3-K)
      FK=K*(K-1)
    7 WORK(K-1)=-WORK(K+1)*FK*FL/FN
C
C     TEST FOR FEASIBILITY OF REDUCTION
C
      IF(K-2)8,8,9
    8 FN=ABS(WORK(1))
      GOTO 10
    9 FN=N-1
      FN=ABS(WORK(2)/FN)
   10 IF(EPS+FN-TOL)11,11,5
C
C     REDUCE POLYNOMIAL
C
   11 EPS=EPS+FN
      N=N-1
      DO 12 J=K,N,2
   12 P(J-1)=P(J-1)+WORK(J-1)
      GOTO 1
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DPECN
C
C        PURPOSE
C           ECONOMIZE A POLYNOMIAL FOR SYMMETRIC RANGE
C
C        USAGE
C           CALL DPECN(P,N,BOUND,EPS,TOL,WORK)
C
C        DESCRIPTION OF PARAMETERS
C           P     - DOUBLE PRECISION COEFFICIENT VECTOR OF GIVEN
C                   POLYNOMIAL
C                   ON RETURN P CONTAINS THE ECONOMIZED POLYNOMIAL
C           N     - DIMENSION OF COEFFICIENT VECTOR P
C                   ON RETURN N CONTAINS DIMENSION OF ECONOMIZED
C                   POLYNOMIAL
C           BOUND - SINGLE PRECISION RIGHT HAND BOUNDARY OF RANGE
C           EPS   - SINGLE PRECISION INITIAL ERROR BOUND
C                   ON RETURN EPS CONTAINS AN ERROR BOUND FOR THE
C                   ECONOMIZED POLYNOMIAL
C           TOL   - SINGLE PRECISION TOLERANCE FOR ERROR
C                   FINAL VALUE OF EPS MUST BE LESS THAN TOL
C           WORK  - DOUBLE PRECISION WORKING STORAGE OF DIMENSION N
C                   (STARTING VALUE OF N RATHER THAN FINAL VALUE)
C
C        REMARKS
C           THE OPERATION IS BYPASSED IN CASE OF N LESS THAN 1.
C           IN CASE OF AN ARBITRARY INTERVAL (XL,XR) IT IS NECESSARY
C           FIRST TO CALCULATE THE EXPANSION OF THE GIVEN POLYNOMIAL
C           WITH ARGUMENT X IN POWERS OF T = (X-(XR-XL)/2).
C           THIS IS ACCOMPLISHED THROUGH SUBROUTINE DPCLD.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SUBROUTINE DPECN TAKES AN (N-1)ST DEGREE POLYNOMIAL
C           APPROXIMATION TO A FUNCTION F(X) VALID WITHIN A TOLERANCE
C           EPS OVER THE INTERVAL (-BOUND,BOUND) AND REDUCES IT IF
C           POSSIBLE TO A POLYNOMIAL OF LOWER DEGREE VALID WITHIN
C           THE GIVEN TOLERANCE TOL.
C           THE INITIAL COEFFICIENT VECTOR P IS REPLACED BY THE FINAL
C           VECTOR. THE INITIAL ERROR BOUND EPS IS REPLACED BY A FINAL
C           ERROR BOUND.
C           N IS REPLACED BY THE DIMENSION OF THE REDUCED POLYNOMIAL.
C           THE COEFFICIENT VECTOR OF THE N-TH CHEBYSHEV POLYNOMIAL
C           IS CALCULATED FROM THE RECURSION FORMULA
C           A(K-1)=-A(K+1)*K*L*L*(K-1)/((N+K-2)*(N-K+2))
C           REFERENCE
C           K. A. BRONS, ALGORITHM 38, TELESCOPE 2, CACM VOL. 4, 1961,
C           NO. 3, PP. 151-152.
C
C     ..................................................................
C
      SUBROUTINE DPECN(P,N,BOUND,EPS,TOL,WORK)
C
      DIMENSION P(1),WORK(1)
      DOUBLE PRECISION P,WORK
C
      FL=BOUND*BOUND
C
C     TEST OF DIMENSION
C
    1 IF(N-1)2,3,6
    2 RETURN
C
    3 IF(EPS+ABS(SNGL(P(1)))-TOL)4,4,5
    4 N=0
      EPS=EPS+ABS(SNGL(P(1)))
    5 RETURN
C
C     CALCULATE EXPANSION OF CHEBYSHEV POLYNOMIAL
C
    6 NEND=N-2
      WORK(N)=-P(N)
      DO 7 J=1,NEND,2
      K=N-J
      FN=(NEND-1+K)*(NEND+3-K)
      FK=K*(K-1)
    7 WORK(K-1)=-WORK(K+1)*DBLE(FK*FL/FN)
C
C     TEST FOR FEASIBILITY OF REDUCTION
C
      IF(K-2)8,8,9
    8 FN=DABS(WORK(1))
      GOTO 10
    9 FN=N-1
      FN=ABS(SNGL(WORK(2))/FN)
   10 IF(EPS+FN-TOL)11,11,5
C
C     REDUCE POLYNOMIAL
C
   11 EPS=EPS+FN
      N=N-1
      DO 12 J=K,N,2
   12 P(J-1)=P(J-1)+WORK(J-1)
      GOTO 1
      END
```

Subroutines PECS and DPECS

These subroutines perform the economization of a polynomial for unsymmetric range.

A given polynomial P(x) with coefficient vector $P = (p_1, \ldots, p_n)$, representing a function f(x) with a maximal error eps over the range (0, bound) -- that is, $\left| f(x) - \sum_{i=1}^{n} p_i x^{i-1} \right| \leqq \text{eps}$ for $0 \leqq x \leqq$ bound -- is reduced to a shorter coefficient vector, if possible, representing f(x) over the same range with a maximal error absolutely less than tol by means of telescoping.

The shifted Chebyshev polynomial $T^s_{n-1}(t) = T_{n-1}(2t - 1)$ has 2^{2n-3} as coefficient of t^{n-1}. As a consequence, t^{n-1} may be replaced by $T^s_{n-1}(t)/2^{2n-3}$ plus a polynomial in t of degree n - 2.

This means that the telescoped polynomial

$$P_1(x) = \sum_{i=1}^{n} p_i x^{i-1} - (p_n \cdot (\text{bound})^{n-1} / 2^{2n-3})\, T^s_{n-1}(x/\text{bound})$$

is of degree n - 2 representing f(x) over the range $0 \leqq x \leqq$ bound with a maximal error err absolutely less than $\text{eps}_1 = \text{eps} + |p_n| \cdot \text{bound}^{n-1} / 2^{2n-3}$, since $|T^s_{n-1}(t)| \leqq 1$ for $0 \leqq t \leqq 1$.

If eps_1 is less than the tolerance tol, the procedure may be repeated with P_1 instead of P, and eps_1 instead of eps.

This iterative scheme results in a polynomial approximation to f(x) which possibly consists of fewer terms and thus may allow the computation of f(x) with less effort than the original approximation P(x).

Naturally, tol should be greater than eps.

For reference see K.A. Brons, Algorithm 37, Telescope 1, CACM vol. 4, 1961, no. 3, p. 151.

```
C
C     ..................................................................
C
C        SUBROUTINE PECS
C
C        PURPOSE
C           ECONOMIZATION OF A POLYNOMIAL FOR UNSYMMETRIC RANGE
C
C        USAGE
C           CALL PECS (P,N,BOUND,EPS,TOL,WORK)
C
C        DESCRIPTION OF PARAMETERS
C           P     - COEFFICIENT VECTOR OF GIVEN POLYNOMIAL
C           N     - DIMENSION OF COEFFICIENT VECTOR
C           BOUND - RIGHT HAND BOUNDARY OF INTERVAL
C           EPS   - INITIAL ERROR BOUND
C           TOL   - TOLERANCE FOR ERROR
C           WORK  - WORKING STORAGE OF DIMENSION N
C
C        REMARKS
C           THE INITIAL COEFFICIENT VECTOR P IS REPLACED BY THE
C           ECONOMIZED VECTOR.
C           THE INITIAL ERROR BOUND EPS IS REPLACED BY A FINAL
C           ERROR BOUND.
C           N IS REPLACED BY THE DIMENSION OF THE REDUCED POLYNOMIAL.
C           IN CASE OF AN ARBITRARY INTERVAL (XL,XR) IT IS NECESSARY
C           FIRST TO CALCULATE THE EXPANSION OF THE GIVEN POLYNOMIAL
C           WITH ARGUMENT X IN POWERS OF T = (X-XL).
C           THIS IS ACCOMPLISHED THROUGH SUBROUTINE PCLD.
C           OPERATION IS BYPASSED IN CASE OF N LESS THAN 1.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SUBROUTINE PECS TAKES AN (N-1)ST DEGREE POLYNOMIAL
C           APPROXIMATION TO A FUNCTION F(X) VALID WITHIN A TOLERANCE
C           EPS OVER THE INTERVAL (0,BOUND) AND REDUCES IT IF POSSIBLE
C           TO A POLYNOMIAL OF LOWER DEGREE VALID WITHIN TOLERANCE
C           TOL.
C           THE COEFFICIENT VECTOR OF THE N-TH SHIFTED CHEBYSHEV
C           POLYNOMIAL IS CALCULATED FROM THE RECURSION FORMULA
C           A(K) = -A(K+1)*K*L*(2*K-1)/(2*(N+K-1)*(N-K+1)).
C           REFERENCE
C           K. A. BRONS, ALGORITHM 37, TELESCOPE 1, CACM VOL. 4, 1961,
C           NO. 3, PP. 151.
C
C     ..................................................................
C
      SUBROUTINE PECS(P,N,BOUND,EPS,TOL,WORK)
C
      DIMENSION P(1),WORK(1)
      FL=BOUND*0.5
C
C     TEST OF DIMENSION
C
    1 IF(N-1)2,3,6
    2 RETURN
    3 IF(EPS+ABS(P(1))-TOL)4,4,5
    4 N=0
      EPS=EPS+ABS(P(1))
    5 RETURN
C
C     CALCULATE EXPANSION OF CHEBYSHEV POLYNOMIAL
C
    6 NEND=N-1
      WORK(N)=-P(N)
      DO 7 J=1,NEND
      K=N-J
      FN=(NEND-1+K)*(N-K)
      FK=K*(K+K-1)
    7 WORK(K)=-WORK(K+1)*FK*FL/FN
C
C        TEST FOR FEASIBILITY OF REDUCTION
C
      FN=ABS(WORK(1))
      IF(EPS+FN-TOL)8,8,5
C
C     REDUCE POLYNOMIAL
C
    8 EPS=EPS+FN
      N=NEND
      DO 9 J=1,NEND
    9 P(J)=P(J)+WORK(J)
      GOTO 1
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DPECS
C
C        PURPOSE
C           ECONOMIZATION OF A POLYNOMIAL FOR UNSYMMETRIC RANGE
C
C        USAGE
C           CALL DPECS(P,N,BOUND,EPS,TOL,WORK)
C
C        DESCRIPTION OF PARAMETERS
C           P      - DOUBLE PRECISION COEFFICIENT VECTOR OF GIVEN
C                    POLYNOMIAL
C           N      - DIMENSION OF COEFFICIENT VECTOR P
C           BOUND  - SINGLE PRECISION RIGHT HAND BOUNDARY OF INTERVAL
C           EPS    - SINGLE PRECISION INITIAL ERROR BOUND
C           TOL    - SINGLE PRECISION TOLERANCE FOR ERROR
C           WORK   - DOUBLE PRECISION WORKING STORAGE OF DIMENSION N
C
C        REMARKS
C           THE INITIAL COEFFICIENT VECTOR P IS REPLACED BY THE
C           ECONOMIZED VECTOR.
C           THE INITIAL ERROR BOUND EPS IS REPLACED BY A FINAL
C           ERROR BOUND.
C           N IS REPLACED BY THE DIMENSION OF THE REDUCED POLYNOMIAL.
C           IN CASE OF AN ARBITRARY INTERVAL (XL,XR) IT IS NECESSARY
C           FIRST TO CALCULATE THE EXPANSION OF THE GIVEN POLYNOMIAL
C           WITH ARGUMENT X IN POWERS OF T = (X-XL).
C           THIS IS ACCOMPLISHED THROUGH SUBROUTINE DPCLD.
C           OPERATION IS BYPASSED IN CASE OF N LESS THAN 1.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SUBROUTINE DPECS TAKES AN (N-1)ST DEGREE POLYNOMIAL
C           APPROXIMATION TO A FUNCTION F(X) VALID WITHIN A TOLERANCE
C           EPS OVER THE INTERVAL (0,BOUND) AND REDUCES IT IF POSSIBLE
C           TO A POLYNOMIAL OF LOWER DEGREE VALID WITHIN TOLERANCE
C           TOL.
C           THE COEFFICIENT VECTOR OF THE N-TH SHIFTED CHEBYSHEV
C           POLYNOMIAL IS CALCULATED FROM THE RECURSION FORMULA
C           A(K) = -A(K+1)*K*L*(2*K-1)/(2*(N+K-1)*(N-K+1)).
C           REFERENCE
C           K. A. BRONS, ALGORITHM 37, TELESCOPE 1, CACM VOL. 4, 1961,
C           NO. 3, PP. 151.
C
C     ..................................................................
C
      SUBROUTINE DPECS(P,N,BOUND,EPS,TOL,WORK)
C
      DIMENSION P(1),WORK(1)
      DOUBLE PRECISION P,WORK
C
      FL=BOUND*0.5
C
C        TEST OF DIMENSION
C
    1 IF(N-1)2,3,6
    2 RETURN
C
    3 IF(EPS+ABS(SNGL(P(1)))-TOL)4,4,5
    4 N=0
      EPS=EPS+ABS(SNGL(P(1)))
    5 RETURN
C
C        CALCULATE EXPANSION OF CHEBYSHEV POLYNOMIAL
C
    6 NEND=N-1
      WORK(N)=-P(N)
      DO 7 J=1,NEND
      K=N-J
      FN=(NEND-1+K)*(N-K)
      FK=K*(K+K-1)
    7 WORK(K)=-WORK(K+1)*DBLE(FK)*DBLE(FL)/DBLE(FN)
C
C        TEST FOR FEASIBILITY OF REDUCTION
C
      FN=DABS(WORK(1))
      IF(EPS+FN-TOL)8,8,5
C
C        REDUCE POLYNOMIAL
C
    8 EPS=EPS+FN
      N=NEND
      DO 9 J=1,NEND
    9 P(J)=P(J)+WORK(J)
      GOTO 1
      END
```

Polynomials: Roots

Subroutine POLRT

This subroutine computes the real and complex roots of a real polynomial.

Given a polynomial

$$f(z) = \sum_{n=0}^{N} a_n z^n \tag{1}$$

let

$Z = X + iY$ be a starting value for a root of f(z).

Then:

$$Z^n = (X + iY)^n \tag{2}$$

Define X_n as real terms of expanded equation (2).

Define Y_n as imaginary terms of expanded equation (2).

Then for:

$$n = 0$$

$$X_o = 1.0$$

$$Y_o = 0.0$$

$$n > 0$$

$$X_n = X \cdot X_{n-1} - Y \cdot Y_{n-1} \tag{3}$$

$$Y_n = X \cdot Y_{n-1} + Y \cdot X_{n-1} \tag{4}$$

Let:

U be the real terms of (1)
V be the imaginary terms of (1)

Then:

$$U = \sum_{n=0}^{N} a_n X_n \tag{5}$$

$$V = \sum_{n=0}^{N} a_n Y_n \tag{6}$$

or:

$$U = a_o + \sum_{n=1}^{N} a_n X_n \tag{7}$$

$$V = \sum_{n=1}^{N} a_n Y_n \tag{8}$$

$$\frac{\partial U}{\partial X} = \sum_{n=1}^{N} n \cdot X_{n-1} \cdot a_n \qquad (9)$$

$$\frac{\partial U}{\partial Y} = - \sum_{n=1}^{N} n\, Y_{n-1}\, a_n \qquad (10)$$

Note that equations (3), (4), (7), (8), (9), and (10) can be performed iteratively for n = 1 to N by saving X_{n-1} and Y_{n-1}.

Using the Newton-Raphson method for computing ΔX, ΔY, the result is:

$$\Delta X = \left(V\frac{\partial U}{\partial Y} - U\frac{\partial U}{\partial X}\right) \Bigg/ \left[\left(\frac{\partial U}{\partial X}\right)^2 + \left(\frac{\partial U}{\partial Y}\right)^2\right] \qquad (11)$$

$$\Delta Y = - \left(U\frac{\partial U}{\partial Y} + V\frac{\partial U}{\partial X}\right) \Bigg/ \left[\left(\frac{\partial U}{\partial X}\right)^2 + \left(\frac{\partial U}{\partial Y}\right)^2\right] \qquad (12)$$

after applying the Cauchy-Riemann equations.

Thus, for the next iteration:

$$X' = X + \Delta X$$

$$Y' = Y + \Delta Y$$

```
C
C     ..................................................................
C
C        SUBROUTINE POLRT
C
C        PURPOSE
C           COMPUTES THE REAL AND COMPLEX ROOTS OF A REAL POLYNOMIAL
C
C        USAGE
C           CALL POLRT(XCOF,COF,M,ROOTR,ROOTI,IER)
C
C        DESCRIPTION OF PARAMETERS
C           XCOF -VECTOR OF M+1 COEFFICIENTS OF THE POLYNOMIAL
C                 ORDERED FROM SMALLEST TO LARGEST POWER
C           COF  -WORKING VECTOR OF LENGTH M+1
C           M    -ORDER OF POLYNOMIAL
C           ROOTR-RESULTANT VECTOR OF LENGTH M CONTAINING REAL ROOTS
C                 OF THE POLYNOMIAL
C           ROOTI-RESULTANT VECTOR OF LENGTH M CONTAINING THE
C                 CORRESPONDING IMAGINARY ROOTS OF THE POLYNOMIAL
C           IER  -ERROR CODE WHERE
C                 IER=0  NO ERROR
C                 IER=1  M LESS THAN ONE
C                 IER=2  M GREATER THAN 36
C                 IER=3  UNABLE TO DETERMINE ROOT WITH 500 INTERATIONS
C                        ON 5 STARTING VALUES
C                 IER=4  HIGH ORDER COEFFICIENT IS ZERO
C
C        REMARKS
C           LIMITED TO 36TH ORDER POLYNOMIAL OR LESS.
C           FLOATING POINT OVERFLOW MAY OCCUR FOR HIGH ORDER
C           POLYNOMIALS BUT WILL NOT AFFECT THE ACCURACY OF THE RESULTS.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           NEWTON-RAPHSON ITERATIVE TECHNIQUE.  THE FINAL ITERATIONS
C           ON EACH ROOT ARE PERFORMED USING THE ORIGINAL POLYNOMIAL
C           RATHER THAN THE REDUCED POLYNOMIAL TO AVOID ACCUMULATED
C           ERRORS IN THE REDUCED POLYNOMIAL.
C
C     ..................................................................
      SUBROUTINE POLRT(XCOF,COF,M,ROOTR,ROOTI,IER)
      DIMENSION XCOF(1),COF(1),ROOTR(1),ROOTI(1)
      DOUBLE PRECISION XO,YO,X,Y,XPR,YPR,UX,UY,V,YT,XT,U,XT2,YT2,SUMSQ,
     1 OX,DY,TEMP,ALPHA
C
C     ..................................................................
C
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
C        STATEMENT WHICH FOLLOWS.
C
C     DOUBLE PRECISION XCOF,COF,ROOTR,ROOTI
C
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
C        ROUTINE.
C
C        THE DOUBLE PRECISION VERSION MAY BE MODIFIED BY CHANGING THE
C        CONSTANT IN STATEMENT 78 TO 1.0D-12 AND IN STATEMENT 122 TO
C        1.0D-10.  THIS WILL PROVIDE HIGHER PRECISION RESULTS AT THE
C        COST OF EXECUTION TIME
C
C        ..................................................................
      IFIT=0
      N=M
      IER=0
      IF(XCOF(N+1))10,25,10
   10 IF(N) 15,15,32
C
C        SET ERROR CODE TO 1
C
   15 IER=1
   20 RETURN
C
C        SET ERROR CODE TO 4
C
   25 IER=4
      GO TO 20
C
C        SET ERROR CODE TO 2
C
   30 IER=2
      GO TO 20
   32 IF(N-36) 35,35,30
   35 NX=N
      NXX=N+1
      N2=1
      KJ1 = N+1
      DO 40 L=1,KJ1
      MT=KJ1-L+1
   40 COF(MT)=XCOF(L)
C
C        SET INITIAL VALUES
C
   45 XO=.00500101
      YO=0.01000101
C
C        ZERO INITIAL VALUE COUNTER
C
      IN=0
   50 X=XO
C
C        INCREMENT INITIAL VALUES AND COUNTER
C
      XO=-10.0*YO
      YO=-10.0*X
C
C        SET X AND Y TO CURRENT VALUE
C
      X=XO
      Y=YO
      IN=IN+1
      GO TO 59
   55 IFIT=1
      XPR=X
      YPR=Y
C
C        EVALUATE POLYNOMIAL AND DERIVATIVES
C
   59 ICT=0
   60 UX=0.0
      UY=0.0
      V =0.0
      YT=0.0
      XT=1.0
      U=COF(N+1)
      IF(U) 65,130,65
   65 DO 70 I=1,N
      L =N-I+1
      TEMP=COF(L)
      XT2=X*XT-Y*YT
      YT2=X*YT+Y*XT
      U=U+TEMP*XT2
      V=V+TEMP*YT2
      FI=I
      UX=UX+FI*XT*TEMP
      UY=UY-FI*YT*TEMP
      XT=XT2
   70 YT=YT2
      SUMSQ=UX*UX+UY*UY
      IF(SUMSQ) 75,110,75
   75 DX=(V*UY-U*UX)/SUMSQ
      X=X+DX
      DY=-(U*UY+V*UX)/SUMSQ
      Y=Y+DY
   78 IF(DABS(DY)+DABS(DX)-1.0D-05) 100,80,80
C
C        STEP ITERATION COUNTER
C
   80 ICT=ICT+1
      IF(ICT-500) 60,85,85
   85 IF(IFIT)100,90,100
   90 IF(IN-5) 50,95,95
C
C        SET ERROR CODE TO 3
C
   95 IER=3
      GO TO 20
  100 DO 105 L=1,NXX
      MT=KJ1-L+1
      TEMP=XCOF(MT)
      XCOF(MT)=COF(L)
  105 COF(L)=TEMP
      ITEMP=N
      N=NX
      NX=ITEMP
      IF(IFIT) 120,55,120
  110 IF(IFIT) 115,50,115
  115 X=XPR
      Y=YPR
  120 IFIT=0
  122 IF(DABS(Y)-1.0D-4*DABS(X)) 135,125,125
  125 ALPHA=X+X
      SUMSQ=X*X+Y*Y
      N=N-2
      GO TO 140
  130 X=0.0
      NX=NX-1
      NXX=NXX-1
  135 Y=0.0
      SUMSQ=0.0
      ALPHA=X
      N=N-1
  140 COF(2)=COF(2)+ALPHA*COF(1)
```

```
145 DO 150 L=2,N                                                    PLRT1890
150 COF(L+1)=COF(L+1)+ALPHA*COF(L)-SUMSQ*COF(L-1)                   PLRT1900
155 ROOTI(N2)=Y                                                     PLRT1910
    ROOTR(N2)=X                                                     PLRT1920
    N2=N2+1                                                         PLRT1930
    IF(SUMSQ) 160,165,160                                           PLRT1940
160 Y=-Y                                                            PLRT1950
    SUMSQ=0.0                                                       PLRT1960
    GO TO 155                                                       PLRT1970
165 IF(N) 20,20,45                                                  PLRT1980
    END                                                             PLRT1990
```

Subroutines PRQD and DPRQD

These subroutines find the roots of a real polynomial by means of the QD-algorithm with displacement. The progressive QD-algorithm is a fast device for calculation of all roots of a polynomial P(x) with real coefficients when there are no approximations to the roots available.

The roots of $P(x) = 0$ are determined by solving for the poles of $Q(x)/P(x)$ where $Q(x)$ is some polynomial of smaller degree than $P(x)$. In the following, n is used as the symbol for the degree of $P(x)$. Using the derivative $P'(x)$ for $Q(x)$ has the advantage that the poles are simple even in case of multiple roots of $P(x)$.

Start of the QD-algorithm requires the continued S-fraction:

$$\cfrac{c}{x-q_1 \cfrac{}{1-e_1 \cfrac{}{x-q_2 \cfrac{}{1-\ddots \cfrac{q_n}{1}}}}} \quad \text{of } P'(x)/P(x)$$

This calculation is performed using the Euclidean algorithm. By means of normalization such that the highest coefficient is 1, $Q_0(x)$ is obtained from $P(x)$, and $P_1(x)$ from $P'(x)$. Then e_k and q_k are calculated from the iteration scheme:

$$q_k Q_k(x) = x \cdot P_k(x) - Q_{k-1}(x), \quad k = 1, 2, \ldots, n \qquad (1)$$

$$e_k P_{k+1}(x) = Q_k(x) - P_k(x), \quad k = 1, 2, \ldots, n-1 \qquad (2)$$

using the fact that all elements $Q_i(x)$ and $P_i(x)$ are normalized by definition.

The following example is an illustration for this iteration scheme:

$$P(x) = 1 + 3x + 3x^2 + x^3 \quad , n = 3$$

$$Q_0(x) = x^3 + 3x^2 + 3x + 1$$

$$P_1(x) = x^2 + 2x + 1 \quad , q_1 Q_1(x) = -x^2 - 2x - 1, \; q_1 = -1$$

$$Q_1(x) = x^2 + 2x + 1 \quad , e_1 P_2(x) = 0 \quad , e_1 = 0$$

where $Q_1(x)$ is a common divisor of $P'(x)/P(x)$ and $q_1 = -1$ is the value of the real root which factors out first.

The above iteration scheme breaks down if some intermediate value of q_i with $0 < i < n$ is equal or approximately equal to zero. In this case an error code is set to 4, which indicates that no S-fraction exists for $P'(x)/P(x)$.

If Q_0 and P_1 have a common divisor of degree j, this common divisor is obtained as Q_{n-j}, and necessarily $e_{n-j} P_{n-j+1} = 0$.

Due to roundoff errors, all coefficients of $e_{n-j} P_{n-j+1}$ cannot be expected to vanish exactly; therefore, allowance is made for coefficients of small absolute value.

If the highest coefficient of $e_{n-j} P_{n-j+1}$ is small in absolute value but some lower coefficient is not, the error code is set to 4, and the subroutine is abandoned.

The coefficients e_k, q_k obtained by the Euclidean algorithm form the first row of the QD-array indicated by superscript 0.

e_0	q_1	e_1	q_2	e_2	...	q_m	e_m
0	$q_1^{(0)}$	$e_1^{(0)}$	$q_2^{(0)}$	$e_2^{(0)}$	...	$q_m^{(0)}$	$e_m^{(0)}$
0	$q_1^{(1)}$	$e_1^{(1)}$	$q_2^{(1)}$	$e_2^{(1)}$	...	$q_m^{(1)}$	$e_m^{(1)}$
. . . . .							

Normally m is equal to n but it may be smaller if a common factor exists.

The following rows are obtained by means of the relationships:

$$q_i^{(v+1)} = q_i^{(v)} + e_i^{(v)} - e_{i-1}^{(v+1)}, \quad i = 1, \ldots, m \tag{3}$$

$$e_i^{(v+1)} = \frac{e_i^{(v)} \cdot q_{i+1}^{(v)}}{q_i^{(v+1)}}, \quad i = 1, \ldots, m-1 \tag{4}$$

together with $e_0^{(v)} = e_m^{(v)} = 0$.

Let x_i denote the roots of P(x) ordered in decreasing absolute value. If P(x) has only roots of different absolute value, there is convergence of the q_k-column to the value x_k. Complex roots are indicated by oscillation of certain q-values.

If x_k and x_{k+1} are of equal absolute value, then the roots of

$$x^2 - (q_k^{(v+1)} + q_{k+1}^{(v)})\, x + q_k^{(v)} q_{k+1}^{(v)} = 0 \tag{5}$$

converge to x_k and x_{k+1}. The same reasoning applies if the relation $|x_k| = |x_{k+1}|$ holds only approximately.

The QD-algorithm in the form (3), (4) suffers from the fact that convergence is rather slow. By means of displacement of the origin a form is obtained which is asymptotically of quadratic convergence.

The strategy is as follows. Start with displacement $t = 0$.

I. As soon as convergence is indicated to a real root -- that is, $e_{m-1}^{(v)}$ is sufficiently small (internal test value 0.01) -- perform a displacement by the amount $q_m^{(v)}$.

Starting with $q_k^{(v)}$, $e_k^{(v)}$, use instead of (3), (4) the iteration scheme:

$$t = t + q_m^{(v)} \tag{6}$$

$$q_k^{(v+1)} = q_k^{(v)} + e_k^{(v)} - e_{k-1}^{(v+1)} - q_m^{(v)}, \quad k = 1, 2, \ldots, m \tag{7}$$

$$e_k^{(v+1)} = q_{k+1}^{(v)} \cdot e_k^{(v)} \Big/ q_k^{(v+1)}, \quad k = 1, 2, \ldots, m-1 \tag{8}$$

Values $e_0^{(v+1)}$ and $e_m^{(v)}$ are thereby set equal to zero.

II. If convergence is indicated to a root pair -- that is, $e_{m-2}^{(v)}$ is less than $e_{m-1}^{(v)}$ and sufficiently small (internal test value 0.01) -- calculate the discriminant of the quadratic equation (5):

$$D = p * p - q_m^{(v)} * q_{m-1}^{(v)}$$

with

$$P = 0.5\,(q_{m-1}^{(v)} + e_{m-1}^{(v)} + q_m^{(v)})$$

If D is positive, provide for a real displacement of amount $P - \sqrt{D}$ or $P + \sqrt{D}$, whichever has the smaller absolute value, and proceed according to (7), (8).

If D is negative, three complex displacements are applied in sequence: the first by amount $P + i\sqrt{|D|}$, the second by $-2i\sqrt{|D|}$, and the third by $V + i\sqrt{|D|}$, resulting in a total real displacement of amount P.

Instead of (6), (7), (8) the following iteration scheme is then obtained. Starting with:

$$t = t + P \tag{9}$$

$$q_1^* = q_1^{(v)} + e_1^{(v)} - P \tag{10}$$

$$P_1 = -D/q_1^{*\,2} \tag{11}$$

$$e_1^* = e_1^{(v)} q_2^{(v)} / \left[q_1^* (1 + p_1)\right] \tag{12}$$

$$q_1^{**} = q_1^* + e_1^* \tag{13}$$

calculate for k = 2, 3, . . . , m:

$$q_k^* = q_k^{(v)} + e_k^{(v)} - e_{k-1}^* - P \tag{14}$$

$$e_{k-1}^{**} = e_{k-1}^* q_k^* / q_{k-1}^{**} \tag{15}$$

$$p_k = p_{k-1} (q_{k-1}^{**} / q_k^*)^2 \tag{16}$$

$$q_{k-1}^{(v+3)} = q_{k-1}^{**} + e_{k-1}^{**} - e_{k-2}^{(v+3)} \tag{17}$$

$$e_k^* = q_{k+1}^{(v)} e_k^{(v)} / \left[q_k^* (1+p_k)\right] \tag{18}$$

$$q_k^{**} = q_k^* + e_k^* - e_{k-1}^{**} \tag{19}$$

$$e_{k-1}^{(v+3)} = (e_{k-1}^{**} q_k^{**} / q_{k-1}^{(v+3)}) (1 + p_k) \tag{20}$$

Finally set:

$$q_m^{(v+3)} = q_m^{**} - e_{m-1}^{(v+3)} \tag{21}$$

Values $e_0^{(v+3)}$, $e_m^{(v)}$ and $q_{m+1}^{(v)}$ are thereby set equal to zero.

III. If none of the values $e_{m-1}^{(v)}$ and $e_{m-2}^{(v)}$ is sufficiently small, the relationship (3), (4) is used with no displacement at all.

Regarding termination of the iterative scheme given by I., II., III. there are two possibilities:

1. If $e_{m-1}^{(v)}$ is negligible (internal test value is 10^{-6} in single precision and 10^{-16} in double precision), a real root is factored out.

2. If $e_{m-2}^{(v)}$ is negligible (with the same internal test values), a pair of roots is factored out.

A maximum of ten times the number of coefficients using I., II. or III. is allowed. At every iteration step for one and the same root or root pair, the internal test value for convergence and the internal test value for acceptance of a displacement are increased by ten percent.

In case of convergence:

1. For a real root --
   real part of root = $t + q_m$
   complex part of root = 0
2. For a real root pair (characterized by $D > 0$) --
   real part of first root = $t + P + \sqrt{D}$
   complex part of first root = 0
   real part of second root = $t + P - \sqrt{D}$
   complex part of second root = 0
3. For a complex root pair (characterized by $D < 0$) --
   real part of first root = $t + P$
   complex part of first root = $\sqrt{-D}$
   real part of second root = $t + P$
   complex part of second root = $-\sqrt{-D}$

As soon as a root or root pair has been factored out, m is reduced by 1 or 2 respectively and the whole procedure I., II., III. is repeated with original values of internal test values, until m = 0 -- that is, all roots have been calculated -- or m = 1, when the last real root is factored immediately. If $P'(x)$, $P(x)$ have a common divisor, the whole process is repeated for this common divisor. Thus, the complete factorization of the original polynomial $P(x)$ is obtained.

Some remarks are in order:

1. The QD-algorithm is a nonlinear relationship and therefore sensitive to roundoff errors. Small intermediate q-values cause loss of accuracy. Therefore, all divisors are checked before division is actually performed. If a divisor is small in

absolute value due to loss of significant digits, the error parameter is set to 3 indicating possible instability of calculation, and further calculation is bypassed.

2. If a maximum of 10 * n iteration steps is insufficient for calculation of all roots, the error parameter is set to 1, indicating poor convergence. Further calculation is bypassed again.

3. If the original polynomial has a degree less than one, the error parameter is set to 2 indicating that no root exists.

4. If, in case of a complex displacement, q_{m-1}^{**} or q_m^* gets very small due to loss of significant digits, and the calculated $e_{m-2}^{(v+3)}$ is less than the internal test value TOL, a pair of roots is calculated from the quadratic equation:

$$x^2 - x \cdot (q_{n-1}^{**} + q_n^* - e_{n-2}^{(v+3)}) + (q_{n-1}^{**} - e_{n-1}^*)$$

$$\left[q_n^* - e_{n-1}^* \; p_{n-1} - q_n^* \frac{e_{n-2}^{**}}{q_{n-2}^{(v+3)}} (1 + p_{n-1})\right] \qquad (22)$$

which is easily obtained from (5).

5. A final test of accuracy is made after calculation of all roots of the given polynomial. The calculated roots are used for calculation of the corresponding coefficient vector. If this calculated coefficient vector has less than a specified number of correct digits (the maximal relative error must be less than TOL), the error parameter is set to -1 indicating that calculation of all roots was successful only with poor accuracy of results.

```
C
C     ..................................................................
C
C        SUBROUTINE PRQD
C
C        PURPOSE
C           CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN POLYNOMIAL
C           WITH REAL COEFFICIENTS.
C
C        USAGE
C           CALL PRQD(C,IC,Q,E,POL,IR,IER)
C
C        DESCRIPTION OF PARAMETERS
C           C      - COEFFICIENT VECTOR OF GIVEN POLYNOMIAL
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH
C                    THE GIVEN COEFFICIENT VECTOR GETS DIVIDED BY THE
C                    LAST NONZERO TERM
C           IC     - DIMENSION OF VECTOR C
C           Q      - WORKING STORAGE OF DIMENSION IC
C                    ON RETURN Q CONTAINS REAL PARTS OF ROOTS
C           E      - WORKING STORAGE OF DIMENSION IC
C                    ON RETURN E CONTAINS COMPLEX PARTS OF ROOTS
C           POL    - WORKING STORAGE OF DIMENSION IC
C                    ON RETURN POL CONTAINS THE COEFFICIENTS OF THE
C                    POLYNOMIAL WITH CALCULATED ROOTS
C                    THIS RESULTING COEFFICIENT VECTOR HAS DIMENSION IR+1
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH
C           IR     - NUMBER OF CALCULATED ROOTS
C                    NORMALLY IR IS EQUAL TO DIMENSION IC MINUS ONE
C           IER    - RESULTING ERROR PARAMETER. SEE REMARKS
C
C        REMARKS
C           THE REAL PART OF THE ROOTS IS STORED IN Q(1) UP TO Q(IR)
C           CORRESPONDING COMPLEX PARTS ARE STORED IN E(1) UP TO E(IR).
C           IER = 0 MEANS NO ERRORS
C           IER = 1 MEANS NO CONVERGENCE WITH FEASIBLE TOLERANCE
C           IER = 2 MEANS POLYNOMIAL IS DEGENERATE (CONSTANT OR ZERO)
C           IER = 3 MEANS SUBROUTINE WAS ABANDONED DUE TO ZERO DIVISOR
C           IER = 4 MEANS THERE EXISTS NO S-FRACTION
C           IER =-1 MEANS CALCULATED COEFFICIENT VECTOR REVEALS POOR
C                   ACCURACY OF THE CALCULATED ROOTS.
C                   THE CALCULATED COEFFICIENT VECTOR HAS LESS THAN
C                   3 CORRECT DIGITS.
C           THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED
C           COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE BEEN
C           CALCULATED.
C           THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT VECTOR IS
C           RECORDED IN Q(IR+1).
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF
C           THE QUOTIENT-DIFFERENCE ALGORITHM WITH DISPLACEMENT.
C           REFERENCE
C           H.RUTISHAUSER, DER QUOTIENTEN-DIFFERENZEN-ALGORITHMUS,
C           BIRKHAEUSER, BASEL/STUTTGART, 1957.
C
C     ..................................................................
C
      SUBROUTINE PRQD(C,IC,Q,E,POL,IR,IER)
C
C     DIMENSIONED DUMMY VARIABLES
      DIMENSION E(1),Q(1),C(1),POL(1)
C
C        NORMALIZATION OF GIVEN POLYNOMIAL
C           TEST OF DIMENSION
C        IR CONTAINS INDEX OF HIGHEST COEFFICIENT
      IER=0
      IR=IC
      EPS=1.E-6
      TOL=1.E-3
      LIMIT=10*IC
      KOUNT=0
    1 IF(IR-1)79,79,2
C
C        DROP TRAILING ZERO COEFFICIENTS
    2 IF(C(IR))4,3,4
    3 IR=IR-1
      GOTO 1
C
C           REARRANGEMENT OF GIVEN POLYNOMIAL
C        EXTRACTION OF ZERO ROOTS
    4 O=1./C(IR)
      IEND=IR-1
      ISTA=1
      NSAV=IR+1
      JBEG=1
C
C        Q(J)=1.
C        Q(J+1)=C(IR-I)/C(IR)
C        Q(IR)=C(J)/C(IR)
C        WHERE J IS THE INDEX OF THE LOWEST NONZERO COEFFICIENT
      DO 9 I=1,IR
      J=NSAV-I
      IF(C(I))7,5,7
    5 GOTO(6,8),JBEG
    6 NSAV=NSAV+1
      Q(ISTA)=0.
      E(ISTA)=0.
      ISTA=ISTA+1
      GOTO 9
    7 JBEG=2
    8 Q(J)=C(I)*O
      C(I)=Q(J)
    9 CONTINUE
C
C           INITIALIZATION
      ESAV=0.
      Q(ISTA)=0.
   10 NSAV=IR
C
C        COMPUTATION OF DERIVATIVE
      EXPT=IR-ISTA
      E(ISTA)=EXPT
      DO 11 I=ISTA,IEND
      EXPT=EXPT-1.0
      POL(I+1)=EPS*ABS(Q(I+1))+EPS
   11 E(I+1)=Q(I+1)*EXPT
C
C        TEST OF REMAINING DIMENSION
      IF(ISTA-IEND)12,20,60
   12 JEND=IEND-1
C
C        COMPUTATION OF S-FRACTION
      DO 19 I=ISTA,JEND
      IF(I-ISTA)13,16,13
   13 IF(ABS(E(I))-POL(I+1))14,14,16
C
C        THE GIVEN POLYNOMIAL HAS MULTIPLE ROOTS, THE COEFFICIENTS OF
C        THE COMMON FACTOR ARE STORED FROM Q(NSAV) UP TO Q(IR)
   14 NSAV=I
      DO 15 K=I,JEND
      IF(ABS(E(K))-POL(K+1))15,15,80
   15 CONTINUE
      GOTO 21
C
C           EUCLIDEAN ALGORITHM
   16 DO 19 K=I,IEND
      E(K+1)=E(K+1)/E(I)
      Q(K+1)=E(K+1)-Q(K+1)
      IF(K-I)18,17,18
C
C        TEST FOR SMALL DIVISOR
   17 IF(ABS(Q(I+1))-POL(I+1))80,80,19
   18 Q(K+1)=Q(K+1)/Q(I+1)
      POL(K+1)=POL(K+1)/ABS(Q(I+1))
      E(K)=Q(K+1)-E(K)
   19 CONTINUE
   20 Q(IR)=-Q(IR)
C
C           THE DISPLACEMENT EXPT IS SET TO 0 AUTOMATICALLY.
C           E(ISTA)=0.,Q(ISTA+1),....,E(NSAV-1),Q(NSAV),E(NSAV)=0.,
C           FORM A DIAGONAL OF THE QD-ARRAY.
C        INITIALIZATION OF BOUNDARY VALUES
   21 E(ISTA)=0.
      NRAN=NSAV-1
   22 E(NRAN+1)=0.
C
C           TEST FOR LINEAR OR CONSTANT FACTOR
C        NRAN-ISTA IS DEGREE-1
      IF(NRAN-ISTA)24,23,31
C
C        LINEAR FACTOR
   23 Q(ISTA+1)=Q(ISTA+1)+EXPT
      E(ISTA+1)=0.
C
```

```
C         TEST FOR UNFACTORED COMMON DIVISOR                            PRQD1690
   24 E(ISTA)=ESAV                                                      PRQD1700
      IF(IR-NSAV)60,60,25                                               PRQD1710
C                                                                       PRQD1720
C         INITIALIZE QD-ALGORITHM FOR COMMON DIVISOR                    PRQD1730
   25 ISTA=NSAV                                                         PRQD1740
      ESAV=E(ISTA)                                                      PRQD1750
      GOTO 10                                                           PRQD1760
C                                                                       PRQD1770
C         COMPUTATION OF ROOT PAIR                                      PRQD1780
   26 P=P+EXPT                                                          PRQD1790
C                                                                       PRQD1800
C            TEST FOR REALITY                                           PRQD1810
      IF(Q)27,28,28                                                     PRQD1820
C                                                                       PRQD1830
C            COMPLEX ROOT PAIR                                          PRQD1840
   27 Q(NRAN)=P                                                         PRQD1850
      Q(NRAN+1)=P                                                       PRQD1860
      E(NRAN)=T                                                         PRQD1870
      E(NRAN+1)=-T                                                      PRQD1880
      GOTO 29                                                           PRQD1890
C                                                                       PRQD1900
C            REAL ROOT PAIR                                             PRQD1910
   28 Q(NRAN)=P-T                                                       PRQD1920
      Q(NRAN+1)=P+T                                                     PRQD1930
      E(NRAN)=0.                                                        PRQD1940
C                                                                       PRQD1950
C            REDUCTION OF DEGREE BY 2 (DEFLATION)                       PRQD1960
   29 NRAN=NRAN-2                                                       PRQD1970
      GOTO 22                                                           PRQD1980
C                                                                       PRQD1990
C         COMPUTATION OF REAL ROOT                                      PRQD2000
   30 Q(NRAN+1)=EXPT+P                                                  PRQD2010
C                                                                       PRQD2020
C            REDUCTION OF DEGREE BY 1 (DEFLATION)                       PRQD2030
      NRAN=NRAN-1                                                       PRQD2040
      GOTO 22                                                           PRQD2050
C                                                                       PRQD2060
C         START QD-ITERATION                                            PRQD2070
   31 JBEG=ISTA+1                                                       PRQD2080
      JEND=NRAN-1                                                       PRQD2090
      TEPS=EPS                                                          PRQD2100
      TDELT=1.E-2                                                       PRQD2110
   32 KOUNT=KOUNT+1                                                     PRQD2120
      P=Q(NRAN+1)                                                       PRQD2130
      R=ABS(E(NRAN))                                                    PRQD2140
C                                                                       PRQD2150
C            TEST FOR CONVERGENCE                                       PRQD2160
      IF(R-TEPS)30,30,33                                                PRQD2170
   33 S=ABS(E(JEND))                                                    PRQD2180
C                                                                       PRQD2190
C         IS THERE A REAL ROOT NEXT                                     PRQD2200
      IF(S-R)38,38,34                                                   PRQD2210
C                                                                       PRQD2220
C         IS DISPLACEMENT SMALL ENOUGH                                  PRQD2230
   34 IF(R-TDELT)36,35,35                                               PRQD2240
   35 P=0.                                                              PRQD2250
   36 Q=P                                                               PRQD2260
      DO 37 J=JBEG,NRAN                                                 PRQD2270
      Q(J)=Q(J)+E(J)-E(J-1)-Q                                           PRQD2280
C                                                                       PRQD2290
C            TEST FOR SMALL DIVISOR                                     PRQD2300
      IF(ABS(Q(J))-POL(J))81,81,37                                      PRQD2310
   37 E(J)=Q(J+1)*E(J)/Q(J)                                             PRQD2320
      Q(NRAN+1)=-E(NRAN)+Q(NRAN+1)-Q                                    PRQD2330
      GOTO 54                                                           PRQD2340
C                                                                       PRQD2350
C         CALCULATE DISPLACEMENT FOR DOUBLE ROOTS                       PRQD2360
C            QUADRATIC EQUATION FOR DOUBLE ROOTS                        PRQD2370
C            X**2-(Q(NRAN)+Q(NRAN+1)+E(NRAN))*X+Q(NRAN)*Q(NRAN+1)=0     PRQD2380
   38 P=0.5*(Q(NRAN)+E(NRAN)+Q(NRAN+1))                                 PRQD2390
      Q=P*P-Q(NRAN)*Q(NRAN+1)                                           PRQD2400
      T=SQRT(ABS(Q))                                                    PRQD2410
C                                                                       PRQD2420
C         TEST FOR CONVERGENCE                                          PRQD2430
      IF(S-TEPS)26,26,39                                                PRQD2440
C                                                                       PRQD2450
C         ARE THERE COMPLEX ROOTS                                       PRQD2460
   39 IF(Q)43,40,40                                                     PRQD2470
   40 IF(P)42,41,41                                                     PRQD2480
   41 T=-T                                                              PRQD2490
   42 P=P+T                                                             PRQD2500
      R=S                                                               PRQD2510
      GOTO 34                                                           PRQD2520
C                                                                       PRQD2530
C         MODIFICATION FOR COMPLEX ROOTS                                PRQD2540
C         IS DISPLACEMENT SMALL ENOUGH                                  PRQD2550
   43 IF(S-TDELT)44,35,35                                               PRQD2560
C                                                                       PRQD2570
C         INITIALIZATION                                                PRQD2580
   44 Q=Q(JBEG)+E(JBEG)-P                                               PRQD2590
C                                                                       PRQD2600
C         TEST FOR SMALL DIVISOR                                        PRQD2610
      IF(ABS(Q)-POL(JBEG))81,81,45                                      PRQD2620
   45 T=(T/Q)**2                                                        PRQD2630
      U=E(JBEG)*Q(JBEG+1)/(Q*(1.+T))                                    PRQD2640
      V=Q+U                                                             PRQD2650
      KOUNT=KOUNT+2                                                     PRQD2660
C                                                                       PRQD2670
C         THREEFOLD LOOP FOR COMPLEX DISPLACEMENT                       PRQD2680
      DO 53 J=JBEG,NRAN                                                 PRQD2690
      Q=Q(J+1)+E(J+1)-U-P                                               PRQD2700
C                                                                       PRQD2710
C         TEST FOR SMALL DIVISOR                                        PRQD2720
      IF(ABS(V)-POL(J))46,46,49                                         PRQD2730
   46 IF(J-NRAN)81,47,81                                                PRQD2740
   47 EXPT=EXPT+P                                                       PRQD2750
      IF(ABS(E(JEND))-TOL)48,48,81                                      PRQD2760
   48 P=0.5*(V+Q-E(JEND))                                               PRQD2770
      Q=P*P-(V-U)*(Q-U*T-Q*W*(1.+T)/Q(JEND))                            PRQD2780
      T=SQRT(ABS(Q))                                                    PRQD2790
      GOTO 26                                                           PRQD2800
C                                                                       PRQD2810
C            TEST FOR SMALL DIVISOR                                     PRQD2820
   49 IF(ABS(Q)-POL(J+1))46,46,50                                       PRQD2830
   50 W=U*Q/V                                                           PRQD2840
      T=T*(V/Q)**2                                                      PRQD2850
      Q(J)=V+W-E(J-1)                                                   PRQD2860
      U=0.                                                              PRQD2870
      IF(J-NRAN)51,52,52                                                PRQD2880
   51 U=Q(J+2)*E(J+1)/(Q*(1.+T))                                        PRQD2890
   52 V=Q+U-W                                                           PRQD2900
C                                                                       PRQD2910
C         TEST FOR SMALL DIVISOR                                        PRQD2920
      IF(ABS(Q(J))-POL(J))81,81,53                                      PRQD2930
   53 E(J)=W*V*(1.+T)/Q(J)                                              PRQD2940
      Q(NRAN+1)=V-E(NRAN)                                               PRQD2950
   54 EXPT=EXPT+P                                                       PRQD2960
      TEPS=TEPS*1.1                                                     PRQD2970
      TDELT=TDELT*1.1                                                   PRQD2980
      IF(KOUNT-LIMIT)32,55,55                                           PRQD2990
C                                                                       PRQD3000
C         NO CONVERGENCE WITH FEASIBLE TOLERANCE                        PRQD3010
C            ERROR RETURN IN CASE OF UNSATISFACTORY CONVERGENCE         PRQD3020
   55 IER=1                                                             PRQD3030
C                                                                       PRQD3040
C         REARRANGE CALCULATED ROOTS                                    PRQD3050
   56 IEND=NSAV-NRAN-1                                                  PRQD3060
      E(ISTA)=ESAV                                                      PRQD3070
      IF(IEND)59,59,57                                                  PRQD3080
   57 DO 58 I=1,IEND                                                    PRQD3090
      J=ISTA+I                                                          PRQD3100
      K=NRAN+1+I                                                        PRQD3110
      E(J)=E(K)                                                         PRQD3120
   58 Q(J)=Q(K)                                                         PRQD3130
   59 IR=ISTA+IEND                                                      PRQD3140
C                                                                       PRQD3150
C         NORMAL RETURN                                                 PRQD3160
   60 IR=IR-1                                                           PRQD3170
      IF(IR)78,78,61                                                    PRQD3180
C                                                                       PRQD3190
C         REARRANGE CALCULATED ROOTS                                    PRQD3200
   61 DO 62 I=1,IR                                                      PRQD3210
      Q(I)=Q(I+1)                                                       PRQD3220
   62 E(I)=E(I+1)                                                       PRQD3230
C                                                                       PRQD3240
C         CALCULATE COEFFICIENT VECTOR FROM ROOTS                       PRQD3250
      POL(IR+1)=1.                                                      PRQD3260
      IEND=IR-1                                                         PRQD3270
      JBEG=1                                                            PRQD3280
      DO 69 J=1,IR                                                      PRQD3290
      ISTA=IR+1-J                                                       PRQD3300
      Q=0.                                                              PRQD3310
      P=Q(ISTA)                                                         PRQD3320
      T=E(ISTA)                                                         PRQD3330
      IF(T)65,63,65                                                     PRQD3340
C                                                                       PRQD3350
C         MULTIPLY WITH LINEAR FACTOR                                   PRQD3360
   63 DO 64 I=ISTA,IR                                                   PRQD3370
      POL(I)=Q-P*POL(I+1)                                               PRQD3380
   64 Q=POL(I+1)                                                        PRQD3390
      GOTO 69                                                           PRQD3400
   65 GOTO(66,67),JBEG                                                  PRQD3410
   66 JBEG=2                                                            PRQD3420
      POL(ISTA)=0.                                                      PRQD3430
      GOTO 69                                                           PRQD3440
C                                                                       PRQD3450
C         MULTIPLY WITH QUADRATIC FACTOR                                PRQD3460
   67 JBEG=1                                                            PRQD3470
      U=P*P+T*T                                                         PRQD3480
      P=P+P                                                             PRQD3490
      DO 68 I=ISTA,IEND                                                 PRQD3500
      POL(I)=Q-P*POL(I+1)+U*POL(I+2)                                    PRQD3510
   68 Q=POL(I+1)                                                        PRQD3520
      POL(IR)=Q-P                                                       PRQD3530
   69 CONTINUE                                                          PRQD3540
      IF(IER)78,70,78                                                   PRQD3550
C                                                                       PRQD3560
C         COMPARISON OF COEFFICIENT VECTORS, IE. TEST OF ACCURACY       PRQD3570
   70 P=0.                                                              PRQD3580
      DO 75 I=1,IR                                                      PRQD3590
      IF(C(I))72,71,72                                                  PRQD3600
   71 Q=ABS(POL(I))                                                     PRQD3610
      GOTO 73                                                           PRQD3620
   72 Q=ABS((POL(I)-C(I))/C(I))                                         PRQD3630
   73 IF(P-Q)74,75,75                                                   PRQD3640
   74 P=Q                                                               PRQD3650
   75 CONTINUE                                                          PRQD3660
      IF(P-TOL)77,76,76                                                 PRQD3670
   76 IER=-1                                                            PRQD3680
   77 Q(IR+1)=P                                                         PRQD3690
      E(IR+1)=0.                                                        PRQD3700
   78 RETURN                                                            PRQD3710
C                                                                       PRQD3720
C         ERROR RETURNS                                                 PRQD3730
C            ERROR RETURN FOR POLYNOMIALS OF DEGREE LESS THAN 1         PRQD3740
   79 IER=2                                                             PRQD3750
      IR=0                                                              PRQD3760
      RETURN                                                            PRQD3770
C                                                                       PRQD3780
C            ERROR RETURN IF THERE EXISTS NO S-FRACTION                 PRQD3790
   80 IER=4                                                             PRQD3800
      IR=ISTA                                                           PRQD3810
      GOTO 60                                                           PRQD3820
C                                                                       PRQD3830
C            ERROR RETURN IN CASE OF INSTABLE QD-ALGORITHM              PRQD3840
   81 IER=3                                                             PRQD3850
      GOTO 56                                                           PRQD3860
      END                                                               PRQD3870
```

```
C                                                                       DPRQ  10
C     ..................................................................DPRQ  20
C                                                                       DPRQ  30
C        SUBROUTINE DPRQD                                               DPRQ  40
C                                                                       DPRQ  50
C        PURPOSE                                                        DPRQ  60
C           CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN POLYNOMIAL  DPRQ  70
C           WITH REAL COEFFICIENTS.                                     DPRQ  80
C                                                                       DPRQ  90
C        USAGE                                                          DPRQ 100
C           CALL DPRQD(C,IC,Q,E,POL,IR,IER)                             DPRQ 110
C                                                                       DPRQ 120
C        DESCRIPTION OF PARAMETERS                                      DPRQ 130
C           C      - COEFFICIENT VECTOR OF GIVEN POLYNOMIAL             DPRQ 140
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DPRQ 150
C                    THE GIVEN COEFFICIENT VECTOR GETS DIVIDED BY THE   DPRQ 160
C                    LAST NONZERO TERM                                  DPRQ 170
C                    DOUBLE PRECISION ARRAY                             DPRQ 180
C           IC     - DIMENSION OF VECTOR C                              DPRQ 190
C           Q      - WORKING STORAGE OF DIMENSION IC                    DPRQ 200
C                    ON RETURN Q CONTAINS REAL PARTS OF ROOTS           DPRQ 210
C                    DOUBLE PRECISION ARRAY                             DPRQ 220
C           E      - WORKING STORAGE OF DIMENSION IC                    DPRQ 230
C                    ON RETURN E CONTAINS COMPLEX PARTS OF ROOTS        DPRQ 240
C                    DOUBLE PRECISION ARRAY                             DPRQ 250
C           POL    - WORKING STORAGE OF DIMENSION IC                    DPRQ 260
C                    ON RETURN POL CONTAINS THE COEFFICIENTS OF THE     DPRQ 270
```

```
C                 POLYNOMIAL WITH CALCULATED ROOTS                      DPRQ 280
C                 THIS RESULTING COEFFICIENT VECTOR HAS DIMENSION IR+1DPRQ 290
C                 COEFFICIENTS ARE ORDERED FROM LOW TO HIGH             DPRQ 300
C                 DOUBLE PRECISION ARRAY                                DPRQ 310
C          IR     - NUMBER OF CALCULATED ROOTS                          DPRQ 320
C                 NORMALLY IR IS EQUAL TO DIMENSION IC MINUS ONE        DPRQ 330
C          IER    - RESULTING ERROR PARAMETER. SEE REMARKS              DPRQ 340
C                                                                       DPRQ 350
C       REMARKS                                                         DPRQ 360
C          THE REAL PART OF THE ROOTS IS STORED IN Q(1) UP TO Q(IR)     DPRQ 370
C          CORRESPONDING COMPLEX PARTS ARE STORED IN E(1) UP TO E(IR).  DPRQ 380
C          IER = 0 MEANS NO ERRORS                                      DPRQ 390
C          IER = 1 MEANS NO CONVERGENCE WITH FEASIBLE TOLERANCE         DPRQ 400
C          IER = 2 MEANS POLYNOMIAL IS DEGENERATE (CONSTANT OR ZERO)    DPRQ 410
C          IER = 3 MEANS SUBROUTINE WAS ABANDONED DUE TO ZERO DIVISOR   DPRQ 420
C          IER = 4 MEANS THERE EXISTS NO S-FRACTION                     DPRQ 430
C          IER =-1 MEANS CALCULATED COEFFICIENT VECTOR REVEALS POOR     DPRQ 440
C                  ACCURACY OF THE CALCULATED ROOTS.                    DPRQ 450
C                  THE CALCULATED COEFFICIENT VECTOR HAS LESS THAN      DPRQ 460
C                  6 CORRECT DIGITS.                                    DPRQ 470
C          THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED            DPRQ 480
C          COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE BEEN  DPRQ 490
C          CALCULATED.                                                  DPRQ 500
C          THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT VECTOR IS      DPRQ 510
C          RECORDED IN Q(IR+1).                                         DPRQ 520
C                                                                       DPRQ 530
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DPRQ 540
C          NONE                                                         DPRQ 550
C                                                                       DPRQ 560
C       METHOD                                                          DPRQ 570
C          THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF       DPRQ 580
C          THE QUOTIENT-DIFFERENCE ALGORITHM WITH DISPLACEMENT.         DPRQ 590
C          REFERENCE                                                    DPRQ 600
C          H.RUTISHAUSER, DER QUOTIENTEN-DIFFERENZEN-ALGORITHMUS,       DPRQ 610
C          BIRKHAEUSER, BASEL/STUTTGART, 1957.                          DPRQ 620
C                                                                       DPRQ 630
C     ..................................................................DPRQ 640
C                                                                       DPRQ 650
      SUBROUTINE DPRQD(C,IC,Q,E,POL,IR,IER)                             DPRQ 660
C                                                                       DPRQ 670
C      DIMENSIONED DUMMY VARIABLES                                      DPRQ 680
      DIMENSION E(1),Q(1),C(1),POL(1)                                   DPRQ 690
      DOUBLE PRECISION Q,E,O,P,T,EXPT,ESAV,U,V,W,C,POL,EPS              DPRQ 700
C                                                                       DPRQ 710
C        NORMALIZATION OF GIVEN POLYNOMIAL                              DPRQ 720
C           TEST OF DIMENSION                                           DPRQ 730
C        IR CONTAINS INDEX OF HIGHEST COEFFICIENT                       DPRQ 740
      IR=IC                                                             DPRQ 750
      IER=0                                                             DPRQ 760
      EPS=1.D-16                                                        DPRQ 770
      TOL=1.E-6                                                         DPRQ 780
      LIMIT=10*IC                                                       DPRQ 790
      KOUNT=0                                                           DPRQ 800
    1 IF(IR-1)79,79,2                                                   DPRQ 810
C                                                                       DPRQ 820
C        DROP TRAILING ZERO COEFFICIENTS                                DPRQ 830
    2 IF(C(IR))4,3,4                                                    DPRQ 840
    3 IR=IR-1                                                           DPRQ 850
      GOTO 1                                                            DPRQ 860
C                                                                       DPRQ 870
C           REARRANGEMENT OF GIVEN POLYNOMIAL                           DPRQ 880
C        EXTRACTION OF ZERO ROOTS                                       DPRQ 890
    4 O=1.D00/C(IR)                                                     DPRQ 900
      IEND=IR-1                                                         DPRQ 910
      ISTA=1                                                            DPRQ 920
      NSAV=IR+1                                                         DPRQ 930
      JBEG=1                                                            DPRQ 940
C                                                                       DPRQ 950
C        Q(J)=1.                                                        DPRQ 960
C        Q(J+1)=C(IR-I)/C(IR)                                           DPRQ 970
C        Q(IR)=C(J)/C(IR)                                               DPRQ 980
C        WHERE J IS THE INDEX OF THE LOWEST NONZERO COEFFICIENT         DPRQ 990
      DO 9 I=1,IR                                                       DPRQ1000
      J=NSAV-I                                                          DPRQ1010
      IF(C(I))7,5,7                                                     DPRQ1020
    5 GOTO(6,8),JBEG                                                    DPRQ1030
    6 NSAV=NSAV+1                                                       DPRQ1040
      Q(ISTA)=0.D0                                                      DPRQ1050
      E(ISTA)=0.D0                                                      DPRQ1060
      ISTA=ISTA+1                                                       DPRQ1070
      GOTO 9                                                            DPRQ1080
    7 JBEG=2                                                            DPRQ1090
    8 Q(J)=C(I)*O                                                       DPRQ1100
      C(I)=Q(J)                                                         DPRQ1110
    9 CONTINUE                                                          DPRQ1120
C                                                                       DPRQ1130
C           INITIALIZATION                                              DPRQ1140
      ESAV=0.D0                                                         DPRQ1150
      Q(ISTA)=0.D0                                                      DPRQ1160
   10 NSAV=IR                                                           DPRQ1170
C                                                                       DPRQ1180
C        COMPUTATION OF DERIVATIVE                                      DPRQ1190
      EXPT=IR-ISTA                                                      DPRQ1200
      E(ISTA)=EXPT                                                      DPRQ1210
      DO 11 I=ISTA,IEND                                                 DPRQ1220
      EXPT=EXPT-1.D0D                                                   DPRQ1230
      POL(I+1)=EPS*DABS(Q(I+1))+EPS                                     DPRQ1240
   11 E(I+1)=Q(I+1)*EXPT                                                DPRQ1250
C                                                                       DPRQ1260
C        TEST OF REMAINING DIMENSION                                    DPRQ1270
      IF(ISTA-IEND)12,20,60                                             DPRQ1280
   12 JEND=IEND-1                                                       DPRQ1290
C                                                                       DPRQ1300
C        COMPUTATION OF S-FRACTION                                      DPRQ1310
      DO 19 I=ISTA,JEND                                                 DPRQ1320
      IF(I-ISTA)13,16,13                                                DPRQ1330
   13 IF(DABS(E(I))-POL(I+1))14,14,16                                   DPRQ1340
C                                                                       DPRQ1350
C        THE GIVEN POLYNOMIAL HAS MULTIPLE ROOTS. THE COEFFICIENTS OF   DPRQ1360
C        THE COMMON FACTOR ARE STORED FROM Q(NSAV) UP TO Q(IR)          DPRQ1370
   14 NSAV=I                                                            DPRQ1380
      DO 15 K=I,JEND                                                    DPRQ1390
      IF(DABS(E(K))-POL(K+1))15,15,80                                   DPRQ1400
   15 CONTINUE                                                          DPRQ1410
      GOTO 21                                                           DPRQ1420
C                                                                       DPRQ1430
C           EUCLIDEAN ALGORITHM                                         DPRQ1440
   16 DO 19 K=I,IEND                                                    DPRQ1450
      E(K+1)=E(K+1)/E(I)                                                DPRQ1460
      Q(K+1)=E(K+1)-Q(K+1)                                              DPRQ1470
      IF(K-I)18,17,18                                                   DPRQ1480
C                                                                       DPRQ1490
C        TEST FOR SMALL DIVISOR                                         DPRQ1500
   17 IF(DABS(Q(I+1))-POL(I+1))80,80,19                                 DPRQ1510
   18 Q(K+1)=Q(K+1)/Q(I+1)                                              DPRQ1520
      POL(K+1)=POL(K+1)/DABS(Q(I+1))                                    DPRQ1530
      E(K)=Q(K+1)-E(K)                                                  DPRQ1540
   19 CONTINUE                                                          DPRQ1550
   20 Q(IR)=-Q(IR)                                                      DPRQ1560
C                                                                       DPRQ1570
C           THE DISPLACEMENT EXPT IS SET TO 0 AUTOMATICALLY.            DPRQ1580
C           E(ISTA)=0.,Q(ISTA+1),...,E(NSAV-1),Q(NSAV),E(NSAV)=0.,      DPRQ1590
C           FORM A DIAGONAL OF THE QD-ARRAY.                            DPRQ1600
C        INITIALIZATION OF BOUNDARY VALUES                              DPRQ1610
   21 E(ISTA)=0.D0                                                      DPRQ1620
      NRAN=NSAV-1                                                       DPRQ1630
   22 E(NRAN+1)=0.D0                                                    DPRQ1640
C                                                                       DPRQ1650
C           TEST FOR LINEAR OR CONSTANT FACTOR                          DPRQ1660
C        NRAN-ISTA IS DEGREE-1                                          DPRQ1670
      IF(NRAN-ISTA)24,23,31                                             DPRQ1680
C                                                                       DPRQ1690
C        LINEAR FACTOR                                                  DPRQ1700
   23 Q(ISTA+1)=Q(ISTA+1)+EXPT                                          DPRQ1710
      E(ISTA+1)=0.D0                                                    DPRQ1720
C                                                                       DPRQ1730
C        TEST FOR UNFACTORED COMMON DIVISOR                             DPRQ1740
   24 E(ISTA)=ESAV                                                      DPRQ1750
      IF(IR-NSAV)60,60,25                                               DPRQ1760
C                                                                       DPRQ1770
C        INITIALIZE QD-ALGORITHM FOR COMMON DIVISOR                     DPRQ1780
   25 ISTA=NSAV                                                         DPRQ1790
      ESAV=E(ISTA)                                                      DPRQ1800
      GOTO 10                                                           DPRQ1810
C                                                                       DPRQ1820
C        COMPUTATION OF ROOT PAIR                                       DPRQ1830
   26 P=P+EXPT                                                          DPRQ1840
C                                                                       DPRQ1850
C        TEST FOR REALITY                                               DPRQ1860
      IF(O)27,28,28                                                     DPRQ1870
C                                                                       DPRQ1880
C        COMPLEX ROOT PAIR                                              DPRQ1890
   27 Q(NRAN)=P                                                         DPRQ1900
      Q(NRAN+1)=P                                                       DPRQ1910
      E(NRAN)=T                                                         DPRQ1920
      E(NRAN+1)=-T                                                      DPRQ1930
      GOTO 29                                                           DPRQ1940
C                                                                       DPRQ1950
C        REAL ROOT PAIR                                                 DPRQ1960
   28 Q(NRAN)=P-T                                                       DPRQ1970
      Q(NRAN+1)=P+T                                                     DPRQ1980
      E(NRAN)=0.D0                                                      DPRQ1990
C                                                                       DPRQ2000
C           REDUCTION OF DEGREE BY 2 (DEFLATION)                        DPRQ2010
   29 NRAN=NRAN-2                                                       DPRQ2020
      GOTO 22                                                           DPRQ2030
C                                                                       DPRQ2040
C        COMPUTATION OF REAL ROOT                                       DPRQ2050
   30 Q(NRAN+1)=EXPT+P                                                  DPRQ2060
C                                                                       DPRQ2070
C           REDUCTION OF DEGREE BY 1 (DEFLATION)                        DPRQ2080
      NRAN=NRAN-1                                                       DPRQ2090
      GOTO 22                                                           DPRQ2100
C                                                                       DPRQ2110
C        START QD-ITERATION                                             DPRQ2120
   31 JBEG=ISTA+1                                                       DPRQ2130
      JEND=NRAN-1                                                       DPRQ2140
      TEPS=EPS                                                          DPRQ2150
      TDELT=1.E-2                                                       DPRQ2160
   32 KOUNT=KOUNT+1                                                     DPRQ2170
      P=Q(NRAN+1)                                                       DPRQ2180
      R=ABS(SNGL(E(NRAN)))                                              DPRQ2190
C                                                                       DPRQ2200
C           TEST FOR CONVERGENCE                                        DPRQ2210
      IF(R-TEPS)30,30,33                                                DPRQ2220
   33 S=ABS(SNGL(E(JEND)))                                              DPRQ2230
C                                                                       DPRQ2240
C        IS THERE A REAL ROOT NEXT                                      DPRQ2250
      IF(S-R)38,38,34                                                   DPRQ2260
C                                                                       DPRQ2270
C        IS DISPLACEMENT SMALL ENOUGH                                   DPRQ2280
   34 IF(R-TDELT)36,35,35                                               DPRQ2290
   35 P=0.D0                                                            DPRQ2300
   36 O=P                                                               DPRQ2310
      DO 37 J=JBEG,NRAN                                                 DPRQ2320
      Q(J)=Q(J)+E(J)-E(J-1)-O                                           DPRQ2330
C                                                                       DPRQ2340
C           TEST FOR SMALL DIVISOR                                      DPRQ2350
      IF(DABS(Q(J))-POL(J))81,81,37                                     DPRQ2360
   37 E(J)=Q(J+1)*E(J)/Q(J)                                             DPRQ2370
      Q(NRAN+1)=-E(NRAN)+Q(NRAN+1)-O                                    DPRQ2380
      GOTO 34                                                           DPRQ2390
C                                                                       DPRQ2400
C        CALCULATE DISPLACEMENT FOR DOUBLE ROOTS                        DPRQ2410
C           QUADRATIC EQUATION FOR DOUBLE ROOTS                         DPRQ2420
C           X**2-(Q(NRAN)+Q(NRAN+1)+E(NRAN))*X+Q(NRAN)*Q(NRAN+1)=0      DPRQ2430
   38 P=0.5D0*(Q(NRAN)+E(NRAN)+Q(NRAN+1))                               DPRQ2440
      O=P*P-Q(NRAN)*Q(NRAN+1)                                           DPRQ2450
      T=DSQRT(DABS(O))                                                  DPRQ2460
C                                                                       DPRQ2470
C        TEST FOR CONVERGENCE                                           DPRQ2480
      IF(S-TEPS)26,26,39                                                DPRQ2490
C                                                                       DPRQ2500
C        ARE THERE COMPLEX ROOTS                                        DPRQ2510
   39 IF(O)43,40,40                                                     DPRQ2520
   40 IF(P)42,41,41                                                     DPRQ2530
   41 T=-T                                                              DPRQ2540
   42 P=P+T                                                             DPRQ2550
      R=S                                                               DPRQ2560
      GOTO 34                                                           DPRQ2570
C                                                                       DPRQ2580
C        MODIFICATION FOR COMPLEX ROOTS                                 DPRQ2590
C        IS DISPLACEMENT SMALL ENOUGH                                   DPRQ2600
   43 IF(S-TDELT)44,35,35                                               DPRQ2610
C                                                                       DPRQ2620
C           INITIALIZATION                                              DPRQ2630
   44 O=Q(JBEG)+E(JBEG)-P                                               DPRQ2640
C                                                                       DPRQ2650
C           TEST FOR SMALL DIVISOR                                      DPRQ2660
      IF(DABS(O)-POL(JBEG))81,81,45                                     DPRQ2670
   45 T=(T/O)**2                                                        DPRQ2680
      U=E(JBEG)*Q(JBEG+1)/(O*(1.0D0+T))                                 DPRQ2690
      V=O+U                                                             DPRQ2700
C                                                                       DPRQ2710
C        THREEFOLD LOOP FOR COMPLEX DISPLACEMENT                        DPRQ2720
      KOUNT=KOUNT+2                                                     DPRQ2730
      DO 53 J=JBEG,NRAN                                                 DPRQ2740
      D=Q(J+1)+E(J+1)-U-P                                               DPRQ2750
C                                                                       DPRQ2760
C           TEST FOR SMALL DIVISOR                                      DPRQ2770
      IF(DABS(V)-POL(J))46,46,49                                        DPRQ2780
   46 IF(J-NRAN)81,47,81                                                DPRQ2790
   47 EXPT=EXPT+P                                                       DPRQ2800
      IF(ABS(SNGL(E(JEND)))-TOL)48,48,81                                DPRQ2810
   48 P=0.5D0*(V+O-E(JEND))                                             DPRQ2820
      O=P*P-(V-U)*(O-U*T-O*W*(1.D0+T)/Q(JEND))                          DPRQ2830
      T=DSQRT(DABS(O))                                                  DPRQ2840
      GOTO 26                                                           DPRQ2850
```

```
C
C           TEST FOR SMALL DIVISOR
   49 IF(DABS(O)-POL(J+1))46,46,50
   50 W=U*O/V
      T=T*(V/O)**2
      Q(J)=V+W-E(J-1)
      U=0.D0
      IF(J-NRAN)51,52,52
   51 U=Q(J+2)*E(J+1)/(O*(1.D0+T))
   52 V=O+U-W
C
C           TEST FOR SMALL DIVISOR
      IF(DABS(Q(J))-POL(J))81,81,53
   53 E(J)=W*V*(1.D0+T)/Q(J)
      Q(NRAN+1)=V-E(NRAN)
   54 EXPT=EXPT+P
      TEPS=TEPS*1.1
      TDELT=TDELT*1.1
      IF(KOUNT-LIMIT)32,55,55
C
C         NO CONVERGENCE WITH FEASIBLE TOLERANCE
C             ERROR RETURN IN CASE OF UNSATISFACTORY CONVERGENCE
   55 IER=1
C
C         REARRANGE CALCULATED ROOTS
   56 IEND=NSAV-NRAN-1
      E(ISTA)=ESAV
      IF(IEND)59,59,57
   57 DO 58 I=1,IEND
      J=ISTA+I
      K=NRAN+1+I
      E(J)=E(K)
   58 Q(J)=Q(K)
   59 IR=ISTA+IEND
C
C         NORMAL RETURN
   60 IR=IR-1
      IF(IR)78,78,61
C
C         REARRANGE CALCULATED ROOTS
   61 DO 62 I=1,IR
      Q(I)=Q(I+1)
   62 E(I)=E(I+1)
C
C         CALCULATE COEFFICIENT VECTOR FROM ROOTS
      POL(IR+1)=1.D0
      IEND=IR-1
      JBEG=1
      DO 69 J=1,IR
      ISTA=IR+1-J
      O=0.D0
      P=Q(ISTA)
      T=E(ISTA)
      IF(T)65,63,65
C
C         MULTIPLY WITH LINEAR FACTOR
   63 DO 64 I=ISTA,IR
      POL(I)=O-P*POL(I+1)
   64 O=POL(I+1)
      GOTO 69
   65 GOTO(66,67),JBEG
   66 JBEG=2
      POL(ISTA)=0.D0
      GOTO 69
C
C         MULTIPLY WITH QUADRATIC FACTOR
   67 JBEG=1
      U=P*P+T*T
      P=P+P
      DO 68 I=ISTA,IEND
      POL(I)=O-P*POL(I+1)+U*POL(I+2)
   68 O=POL(I+1)
      POL(IR)=O-P
   69 CONTINUE
      IF(IER)78,70,78
C
C         COMPARISON OF COEFFICIENT VECTORS, IE. TEST OF ACCURACY
   70 P=0.D0
      DO 75 I=1,IR
      IF(C(I))72,71,72
   71 O=DABS(POL(I))
      GOTO 73
   72 O=DABS((POL(I)-C(I))/C(I))
   73 IF(P-O)74,75,75
   74 P=O
   75 CONTINUE
      IF(SNGL(P)-TOL)77,76,76
   76 IER=-1
   77 Q(IR+1)=P
      E(IR+1)=0.D0
   78 RETURN
C
C         ERROR RETURNS
C             ERROR RETURN FOR POLYNOMIALS OF DEGREE LESS THAN 1
   79 IER=2
      IR=0
      RETURN
C
C           ERROR RETURN IF THERE EXISTS NO S-FRACTION
   80 IER=4
      IR=ISTA
      GOTO 60
C
C           ERROR RETURN IN CASE OF INSTABLE QD-ALGORITHM
   81 IER=3
      GOTO 56
      END
```

Subroutines PRBM and DPRBM

These subroutines compute the real and complex roots of a real polynomial

$$p(x) = c_1 + c_2x + \ldots + c_{n+1}x^n$$

using Bairstow's iterative method of quadratic factorization.

1. Mathematical background

Every real polynomial of degree greater than one can be factored in the form $p(x) = q(x)r(x)$ where $q(x) = x^2 + q_2x + q_1$ is quadratic. If $q(x)$ is reducible, that is, if $q(x)$ is a product of two real linear factors, $p(x)$ has a pair of real roots; and if $q(x)$ is irreducible, $p(x)$ has a complex conjugate pair of roots. If $r(x)$ has degree exceeding one, it too may be factored as above, and so on. The quadratic factors of $p(x)$ are determined by subroutines PQFB and DPQFB. See those routines for a detailed description of Bairstow's method.

2. Programming considerations

The subroutines need only the coefficient vector $C = (c_1, \ldots, c_{IC})$ and the number of coefficients $IC = n+1$ as input. They give as output an error parameter IER, the number of computed roots IR, the real and complex parts of the computed roots in vectors RR and RC, and -- if all the roots have been computed -- the coefficients of the polynomial with the computed roots stored in the vector POL.

If $p(x)$ is a constant, that is, if $c_2 = c_3 = \ldots = 0$, we set IR = 0, IER = 2, and return to the call program.

Otherwise, let $IR = \max \left\{ j \mid c_j \neq 0 \right\}$. Then we may write:

$$p(x) = c_{IR}\, \bar{p}(x)\, \bar{\bar{p}}(x)$$

$$\bar{p}(x) = p_1 + p_2x + \ldots + p_Jx^{J-1}$$

$$\bar{\bar{p}}(x) = x^K$$

where $K \geq 0$, $J = IR - K$, and $p_j = c_{K+j}/c_{IR}$, $j = 1, 2, \ldots, J$, and $p_1 \neq 0$. The coefficients of $\bar{p}$ are stored in POL(1) up to POL(J) and POL(J+1) up to POL(IR) are initialized to zero. If an exponential overflow occurs during the calculation of the p_i we again set IR = 0, IER = 2 and return control to the calling program.

The following example illustrates the procedure so far:

$$p(x) = 0+0\cdot x+50\cdot x^2 + 40\cdot x^3 + 30\cdot x^4 + 20\cdot x^5$$

$$+ 10\cdot x^6 + 0\cdot x^7 + 0\cdot x^8 + 0\cdot x^9$$

IC = 10 IR = 7, J = 5

	C		POL	RR	RC
1:	0	1:	5	0	0
	0		4	0	0
	50		3	-	-
	40		2	-	-
	30	J:	1	-	-
	20		0		
	10	IR:	0		
	0				
	0				
IC:	0				

$$\bar{p}(x) = 5+4x+3x^2+2x^3+x^4$$

$$\bar{\bar{p}}(x) = x^2$$

3. The Bairstow iteration

Upon entering the main part of the routine we assume

$$p(x) = c_{IR}\, \bar{p}(x)\, \bar{\bar{p}}(x)$$

where

$$\bar{p}(x) = p_1 + p_2 x + \ldots + p_j x^{j-1} \qquad (p_j = 1)$$

is the polynomial with the unknown roots, and

$$\bar{\bar{p}}(x) = p_{j+1} + p_{j+2}x + \ldots + p_{IR}x^{IR-j-1} + x^{IR-j}$$

is the polynomial generated from the computed roots. The coefficients $p_1, p_2, \ldots, p_{IR}$ are stored in POL(1), POL (2), . . ., POL(IR). Now there are three possibilities:

(1) j = 1. All the roots have been computed. IR is decreased by 1 in order to record the number of computed roots and control is passed on to the last phase of the subroutine (the appendix).

(2) j = 2. The last root is $x = -p_1$. This is recorded by setting RR(IR - 1) = $-p_1$ and RC(IR - 1) = 0. The polynomial $\bar{\bar{p}}(x)$ is multiplied by the linear factor $x + p_1$, occupying after multiplication the locations POL(2) up to POL(IR). IR is decreased by 1, thus recording the number of computed roots, and control is passed on to the appendix.

(3) $j \geq 3$. In this case the polynomial $\bar{p}(x)$ is factored by means of subroutine PQFB or DPQFB, where the quadratic factor $q_1 + q_2x + x^2$ of the previous step is used as the initial guess (in the first factorization x^2 is used). If factorization is not successful, it is restarted with the initial guesses $-q_1+q_2x+x^2$, $-q_1-q_2x+x^2$ and $q_1-q_2x+x^2$. If none of these trials leads to a successful factorization, q_1 and q_2 are increased by 1 (at most nine times). If after a maximum of 40 trials factorization is still unsuccessful, IR is decreased by J, thus recording the number of computed roots, and the rest of the subroutine is bypassed giving the error message IER = 3.

If any one of these trials is successful, $\bar{p}(x)$ is divided by the determined quadratic factor, occupying after division the locations POL(1) up to POL (J-2). Then the polynomial $\bar{\bar{p}}(x)$ is multiplied by the same quadratic factor and occupies after multiplication the locations POL(J-1) up to POL(IR). At least the quadratic equation is solved, and the real and complex parts of the two roots are stored in the corresponding locations of RR and RC. After decreasing j by 2, the main part of the subroutine is reentered.

Assuming that all the roots of the polynomial have been computed, the coefficients of the polynomial $\bar{\bar{p}}(x)$ are shifted into the locations POL(1) up to POL(IR), and POL(IR+1) is set to 1 (IR is the number of computed roots).

Furthermore, the absolutely greatest error of the coefficients of the resulting polynomial is generated using the formula

$$a = \max_i |\, \delta_i \,|$$

$$\text{where} \quad \delta_i = \begin{cases} (p_i - c_i)/c_i & \text{if } c_i \neq 0 \\ p_i & \text{if } c_i = 0 \end{cases} \qquad i = 1, \ldots, IR$$

The number a is stored in RR(IR+1) giving the user some indication of the accuracy reached by the subroutines. The warning IER = -1 is given if $a > \epsilon$ ($\epsilon = 10^{-3}$ in single precision and $\epsilon = 10^{-6}$ in double precision). The warning IER = -1 is overridden by the warning IER = 1, which is given if any quadratic factorization fails to converge. (See PQFB.)

For reference see J. H. Wilkinson, "The Evaluation of the Zeros of Ill-Conditioned Polynomials (Parts 1 and 2)", Numerische Mathematik, vol. 1 (1959), pp 150-180.

```
C                                                                       PRBM  10
C     ..................................................................PRBM  20
C                                                                       PRBM  30
C        SUBROUTINE PRBM                                                PRBM  40
C                                                                       PRBM  50
C        PURPOSE                                                        PRBM  60
C           TO CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN          PRBM  70
C           POLYNOMIAL WITH REAL COEFFICIENTS.                          PRBM  80
C                                                                       PRBM  90
C        USAGE                                                          PRBM 100
C           CALL PRBM (C,IC,RR,RC,POL,IR,IER)                           PRBM 110
C                                                                       PRBM 120
C        DESCRIPTION OF PARAMETERS                                      PRBM 130
C           C       - INPUT VECTOR CONTAINING THE COEFFICIENTS OF THE   PRBM 140
C                     GIVEN POLYNOMIAL. COEFFICIENTS ARE ORDERED FROM   PRBM 150
C                     LOW TO HIGH. ON RETURN COEFFICIENTS ARE DIVIDED   PRBM 160
C                     BY THE LAST NONZERO TERM.                         PRBM 170
C           IC      - DIMENSION OF VECTORS C, RR, RC, AND POL.          PRBM 180
C           RR      - RESULTANT VECTOR OF REAL PARTS OF THE ROOTS.      PRBM 190
C           RC      - RESULTANT VECTOR OF COMPLEX PARTS OF THE ROOTS.   PRBM 200
C           POL     - RESULTANT VECTOR OF COEFFICIENTS OF THE POLYNOMIAL PRBM 210
C                     WITH CALCULATED ROOTS. COEFFICIENTS ARE ORDERED   PRBM 220
C                     FROM LOW TO HIGH (SEE REMARK 4).                  PRBM 230
C           IR      - OUTPUT VALUE SPECIFYING THE NUMBER OF CALCULATED  PRBM 240
C                     ROOTS. NORMALLY IR IS EQUAL TO IC-1.              PRBM 250
C           IER     - RESULTANT ERROR PARAMETER CODED AS FOLLOWS        PRBM 260
C                       IER=0  - NO ERROR,                              PRBM 270
C                       IER=1  - SUBROUTINE PQFB RECORDS POOR CONVERGENCE PRBM 280
C                                AT SOME QUADRATIC FACTORIZATION WITHIN PRBM 290
C                                50 ITERATION STEPS,                    PRBM 300
C                       IER=2  - POLYNOMIAL IS DEGENERATE, I.E. ZERO OR PRBM 310
C                                CONSTANT,                              PRBM 320
C                                OR OVERFLOW IN NORMALIZATION OF GIVEN  PRBM 330
C                                POLYNOMIAL,                            PRBM 340
C                       IER=3  - THE SUBROUTINE IS BYPASSED DUE TO      PRBM 350
C                                SUCCESSIVE ZERO DIVISORS OR OVERFLOWS  PRBM 360
C                                IN QUADRATIC FACTORIZATION OR DUE TO   PRBM 370
C                                COMPLETELY UNSATISFACTORY ACCURACY,    PRBM 380
C                       IER=-1 - CALCULATED COEFFICIENT VECTOR HAS LESS PRBM 390
C                                THAN THREE CORRECT SIGNIFICANT DIGITS. PRBM 400
C                                THIS REVEALS POOR ACCURACY OF CALCULATED PRBM 410
C                                ROOTS.                                 PRBM 420
C                                                                       PRBM 430
C        REMARKS                                                        PRBM 440
C           (1) REAL PARTS OF THE ROOTS ARE STORED IN RR(1) UP TO RR(IR)PRBM 450
C               AND CORRESPONDING COMPLEX PARTS IN RC(1) UP TO RC(IR).  PRBM 460
C           (2) ERROR MESSAGE IER=1 INDICATES POOR CONVERGENCE WITHIN   PRBM 470
C               50 ITERATION STEPS AT SOME QUADRATIC FACTORIZATION      PRBM 480
C               PERFORMED BY SUBROUTINE PQFB.                           PRBM 490
C           (3) NO ACTION BESIDES ERROR MESSAGE IER=2 IN CASE OF A ZERO PRBM 500
C               OR CONSTANT POLYNOMIAL. THE SAME ERROR MESSAGE IS GIVEN PRBM 510
C               IN CASE OF AN OVERFLOW IN NORMALIZATION OF GIVEN        PRBM 520
C               POLYNOMIAL.                                             PRBM 530
C           (4) ERROR MESSAGE IER=3 INDICATES SUCCESSIVE ZERO DIVISORS  PRBM 540
C               OR OVERFLOWS OR COMPLETELY UNSATISFACTORY ACCURACY AT   PRBM 550
C               ANY QUADRATIC FACTORIZATION PERFORMED BY                PRBM 560
C               SUBROUTINE PQFB. IN THIS CASE CALCULATION IS BYPASSED.  PRBM 570
C               IR RECORDS THE NUMBER OF CALCULATED ROOTS.              PRBM 580
C               POL(1),...,POL(J-IR) ARE THE COEFFICIENTS OF THE        PRBM 590
C               REMAINING POLYNOMIAL, WHERE J IS THE ACTUAL NUMBER OF   PRBM 600
C               COEFFICIENTS IN VECTOR C (NORMALLY J=IC).               PRBM 610
C           (5) IF CALCULATED COEFFICIENT VECTOR HAS LESS THAN THREE    PRBM 620
C               CORRECT SIGNIFICANT DIGITS THOUGH ALL QUADRATIC         PRBM 630
C               FACTORIZATIONS SHOWED SATISFACTORY ACCURACY, THE ERROR  PRBM 640
C               MESSAGE IER=-1 IS GIVEN.                                PRBM 650
C           (6) THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED       PRBM 660
C               COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE  PRBM 670
C               BEEN CALCULATED. IN THIS CASE THE NUMBER OF ROOTS IR IS PRBM 680
C               EQUAL TO THE ACTUAL DEGREE OF THE POLYNOMIAL (NORMALLY  PRBM 690
C               IR=IC-1). THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT PRBM 700
C               VECTOR IS RECORDED IN RR(IR+1).                         PRBM 710
C                                                                       PRBM 720
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PRBM 730
C           SUBROUTINE PQFB      QUADRATIC FACTORIZATION OF A POLYNOMIAL PRBM 740
C                                BY BAIRSTOW ITERATION.                 PRBM 750
C                                                                       PRBM 760
C        METHOD                                                         PRBM 770
C           THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF      PRBM 780
C           SUCCESSIVE QUADRATIC FACTORIZATION PERFORMED BY BAIRSTOW    PRBM 790
C           ITERATION. X**2 IS USED AS INITIAL GUESS FOR THE FIRST      PRBM 800
C           QUADRATIC FACTOR, AND FURTHER EACH CALCULATED QUADRATIC     PRBM 810
C           FACTOR IS USED AS INITIAL GUESS FOR THE NEXT ONE. AFTER     PRBM 820
C           COMPUTATION OF ALL ROOTS THE COEFFICIENT VECTOR IS          PRBM 830
C           CALCULATED AND COMPARED WITH THE GIVEN ONE.                 PRBM 840
C           FOR REFERENCE, SEE J. H. WILKINSON, THE EVALUATION OF THE   PRBM 850
C           ZEROS OF ILL-CONDITIONED POLYNOMIALS (PART ONE AND TWO),    PRBM 860
C           NUMERISCHE MATHEMATIK, VOL.1 (1959), PP.150-180.            PRBM 870
C                                                                       PRBM 880
C     ..................................................................PRBM 890
C                                                                       PRBM 900
      SUBROUTINE PRBM(C,IC,RR,RC,POL,IR,IER)                            PRBM 910
C                                                                       PRBM 920
C                                                                       PRBM 930
      DIMENSION C(1),RR(1),RC(1),POL(1),Q(4)                            PRBM 940
C                                                                       PRBM 950
C        TEST ON LEADING ZERO COEFFICIENTS                              PRBM 960
      EPS=1.E-3                                                         PRBM 970
      LIM=50                                                            PRBM 980
      IR=IC+1                                                           PRBM 990
    1 IR=IR-1                                                           PRBM1000
      IF(IR-1)42,42,2                                                   PRBM1010
    2 IF(C(IR))3,1,3                                                    PRBM1020
C                                                                       PRBM1030
C        WORK UP ZERO ROOTS AND NORMALIZE REMAINING POLYNOMIAL          PRBM1040
    3 IER=0                                                             PRBM1050
      J=IR                                                              PRBM1060
      L=0                                                               PRBM1070
      A=C(IR)                                                           PRBM1080
      DO 8 I=1,IR                                                       PRBM1090
      IF(L)4,4,7                                                        PRBM1100
    4 IF(C(I))6,5,6                                                     PRBM1110
    5 RR(I)=0.                                                          PRBM1120
      RC(I)=0.                                                          PRBM1130
      POL(J)=0.                                                         PRBM1140
      J=J-1                                                             PRBM1150
      GO TO 8                                                           PRBM1160
    6 L=1                                                               PRBM1170
      IST=I                                                             PRBM1180
      J=0                                                               PRBM1190
    7 J=J+1                                                             PRBM1200
      C(I)=C(I)/A                                                       PRBM1210
      POL(J)=C(I)                                                       PRBM1220
      CALL OVERFL(N)                                                    PRBM1230
      IF(N-2)42,8,8                                                     PRBM1240
    8 CONTINUE                                                          PRBM1250
C                                                                       PRBM1260
C        START BAIRSTOW ITERATION                                       PRBM1270
      Q1=0.                                                             PRBM1280
      Q2=0.                                                             PRBM1290
    9 IF(J-2)33,10,14                                                   PRBM1300
C                                                                       PRBM1310
C        DEGREE OF RESTPOLYNOMIAL IS EQUAL TO ONE                       PRBM1320
   10 A=POL(1)                                                          PRBM1330
      RR(IST)=-A                                                        PRBM1340
      RC(IST)=0.                                                        PRBM1350
      IR=IR-1                                                           PRBM1360
      Q2=0.                                                             PRBM1370
      IF(IR-1)13,13,11                                                  PRBM1380
   11 DO 12 I=2,IR                                                      PRBM1390
      Q1=Q2                                                             PRBM1400
      Q2=POL(I+1)                                                       PRBM1410
   12 POL(I)=A*Q2+Q1                                                    PRBM1420
   13 POL(IR+1)=A*Q2                                                    PRBM1430
      GO TO 34                                                          PRBM1440
C        THIS IS BRANCH TO COMPARISON OF COEFFICIENT VECTORS C AND POL  PRBM1450
C                                                                       PRBM1460
C        DEGREE OF RESTPOLYNOMIAL IS GREATER THAN ONE                   PRBM1470
   14 DO 22 L=1,10                                                      PRBM1480
      N=1                                                               PRBM1490
   15 Q(1)=Q1                                                           PRBM1500
      Q(2)=Q2                                                           PRBM1510
      CALL PQFB(POL,J,Q,LIM,I)                                          PRBM1520
      IF(I)16,24,23                                                     PRBM1530
   16 IF(Q1)18,17,18                                                    PRBM1540
   17 IF(Q2)18,21,18                                                    PRBM1550
   18 GO TO (19,20,19,21),N                                             PRBM1560
   19 Q1=-Q1                                                            PRBM1570
      N=N+1                                                             PRBM1580
      GO TO 15                                                          PRBM1590
   20 Q2=-Q2                                                            PRBM1600
      N=N+1                                                             PRBM1610
      GO TO 15                                                          PRBM1620
   21 Q1=1.+Q1                                                          PRBM1630
   22 Q2=1.-Q2                                                          PRBM1640
C                                                                       PRBM1650
C        ERROR EXIT DUE TO UNSATISFACTORY RESULTS OF FACTORIZATION      PRBM1660
      IER=3                                                             PRBM1670
      IR=IR-J                                                           PRBM1680
      RETURN                                                            PRBM1690
C                                                                       PRBM1700
C        WORK UP RESULTS OF QUADRATIC FACTORIZATION                     PRBM1710
   23 IER=1                                                             PRBM1720
   24 Q1=Q(1)                                                           PRBM1730
      Q2=Q(2)                                                           PRBM1740
C                                                                       PRBM1750
C        PERFORM DIVISION OF FACTORIZED POLYNOMIAL BY QUADRATIC FACTOR  PRBM1760
      B=0.                                                              PRBM1770
      A=0.                                                              PRBM1780
      I=J                                                               PRBM1790
   25 H=-Q1*B-Q2*A+POL(I)                                               PRBM1800
      POL(I)=B                                                          PRBM1810
      B=A                                                               PRBM1820
      A=H                                                               PRBM1830
      I=I-1                                                             PRBM1840
      IF(I-2)26,26,25                                                   PRBM1850
   26 POL(2)=B                                                          PRBM1860
      POL(1)=A                                                          PRBM1870
C                                                                       PRBM1880
C        MULTIPLY POLYNOMIAL WITH CALCULATED ROOTS BY QUADRATIC FACTOR  PRBM1890
      L=IR-1                                                            PRBM1900
      IF(J-L)27,27,29                                                   PRBM1910
   27 DO 28 I=J,L                                                       PRBM1920
   28 POL(I-1)=POL(I-1)+POL(I)*Q2+POL(I+1)*Q1                           PRBM1930
   29 POL(L)=POL(L)+POL(L+1)*Q2+Q1                                      PRBM1940
      POL(IR)=POL(IR)+Q2                                                PRBM1950
C                                                                       PRBM1960
C        CALCULATE ROOT-PAIR FROM QUADRATIC FACTOR X*X+Q2*X+Q1          PRBM1970
      H=-.5*Q2                                                          PRBM1980
      A=H*H-Q1                                                          PRBM1990
      B=SQRT(ABS(A))                                                    PRBM2000
      IF(A)30,30,31                                                     PRBM2010
   30 RR(IST)=H                                                         PRBM2020
      RC(IST)=B                                                         PRBM2030
      IST=IST+1                                                         PRBM2040
      RR(IST)=H                                                         PRBM2050
      RC(IST)=-B                                                        PRBM2060
      GO TO 32                                                          PRBM2070
   31 B=H+SIGN(B,H)                                                     PRBM2080
      RR(IST)=Q1/B                                                      PRBM2090
      RC(IST)=0.                                                        PRBM2100
      IST=IST+1                                                         PRBM2110
      RR(IST)=B                                                         PRBM2120
      RC(IST)=0.                                                        PRBM2130
   32 IST=IST+1                                                         PRBM2140
      J=J-2                                                             PRBM2150
      GO TO 9                                                           PRBM2160
C                                                                       PRBM2170
C        SHIFT BACK ELEMENTS OF POL BY 1 AND COMPARE VECTORS POL AND C  PRBM2180
   33 IR=IR-1                                                           PRBM2190
   34 A=0.                                                              PRBM2200
      DO 38 I=1,IR                                                      PRBM2210
      Q1=C(I)                                                           PRBM2220
      Q2=POL(I+1)                                                       PRBM2230
      POL(I)=Q2                                                         PRBM2240
      IF(Q1)35,36,35                                                    PRBM2250
   35 Q2=(Q1-Q2)/Q1                                                     PRBM2260
   36 Q2=ABS(Q2)                                                        PRBM2270
      IF(Q2-A)38,38,37                                                  PRBM2280
   37 A=Q2                                                              PRBM2290
   38 CONTINUE                                                          PRBM2300
      I=IR+1                                                            PRBM2310
      POL(I)=1.                                                         PRBM2320
      RR(I)=A                                                           PRBM2330
      RC(I)=0.                                                          PRBM2340
      IF(IER)39,39,41                                                   PRBM2350
   39 IF(A-EPS)41,41,40                                                 PRBM2360
C                                                                       PRBM2370
C        WARNING DUE TO POOR ACCURACY OF CALCULATED COEFFICIENT VECTOR  PRBM2380
   40 IER=-1                                                            PRBM2390
   41 RETURN                                                            PRBM2400
C                                                                       PRBM2410
C        ERROR EXIT DUE TO DEGENERATE POLYNOMIAL OR OVERFLOW IN         PRBM2420
C        NORMALIZATION                                                  PRBM2430
   42 IER=2                                                             PRBM2440
      IR=0                                                              PRBM2450
      RETURN                                                            PRBM2460
      END                                                               PRBM2470
```

```
C                                                                       DPRB  10
C     ..................................................................DPRB  20
C                                                                       DPRB  30
C       SUBROUTINE DPRBM                                                DPRB  40
C                                                                       DPRB  50
C       PURPOSE                                                         DPRB  60
C          TO CALCULATE ALL REAL AND COMPLEX ROOTS OF A GIVEN           DPRB  70
C          POLYNOMIAL WITH REAL COEFFICIENTS.                           DPRB  80
C                                                                       DPRB  90
C       USAGE                                                           DPRB 100
C          CALL DPRBM (C,IC,RR,RC,POL,IR,IER)                           DPRB 110
C                                                                       DPRB 120
C       DESCRIPTION OF PARAMETERS                                       DPRB 130
C          C      - DOUBLE PRECISION INPUT VECTOR CONTAINING THE        DPRB 140
C                   COEFFICIENTS OF THE GIVEN POLYNOMIAL. COEFFICIENTS  DPRB 150
C                   ARE ORDERED FROM LOW TO HIGH. ON RETURN COEFFI-     DPRB 160
C                   CIENTS ARE DIVIDED BY THE LAST NONZERO TERM.        DPRB 170
C          IC     - DIMENSION OF VECTORS C, RR, RC, AND POL.            DPRB 180
C          RR     - RESULTANT DOUBLE PRECISION VECTOR OF REAL PARTS     DPRB 190
C                   OF THE ROOTS.                                       DPRB 200
C          RC     - RESULTANT DOUBLE PRECISION VECTOR OF COMPLEX PARTS  DPRB 210
C                   OF THE ROOTS.                                       DPRB 220
C          POL    - RESULTANT DOUBLE PRECISION VECTOR OF COEFFICIENTS   DPRB 230
C                   OF THE POLYNOMIAL WITH CALCULATED ROOTS.            DPRB 240
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH (SEE      DPRB 250
C                   REMARK 4).                                          DPRB 260
C          IR     - OUTPUT VALUE SPECIFYING THE NUMBER OF CALCULATED    DPRB 270
C                   ROOTS. NORMALLY IR IS EQUAL TO IC-1.                DPRB 280
C          IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS          DPRB 290
C                     IER=0  - NO ERROR,                                DPRB 300
C                     IER=1  - SUBROUTINE DPQFB RECORDS POOR CONVERGENCEDPRB 310
C                              AT SOME QUADRATIC FACTORIZATION WITHIN   DPRB 320
C                              100 ITERATION STEPS,                     DPRB 330
C                     IER=2  - POLYNOMIAL IS DEGENERATE, I.E. ZERO OR  DPRB 340
C                              CONSTANT,                                DPRB 350
C                              OR OVERFLOW IN NORMALIZATION OF GIVEN    DPRB 360
C                              POLYNOMIAL,                              DPRB 370
C                     IER=3  - THE SUBROUTINE IS BYPASSED DUE TO        DPRB 380
C                              SUCCESSIVE ZERO DIVISORS OR OVERFLOWS    DPRB 390
C                              IN QUADRATIC FACTORIZATION OR DUE TO     DPRB 400
C                              COMPLETELY UNSATISFACTORY ACCURACY,      DPRB 410
C                     IER=-1 - CALCULATED COEFFICIENT VECTOR HAS LESS   DPRB 420
C                              THAN SIX CORRECT SIGNIFICANT DIGITS.     DPRB 430
C                              THIS REVEALS POOR ACCURACY OF CALCULATED DPRB 440
C                              ROOTS.                                   DPRB 450
C                                                                       DPRB 460
C       REMARKS                                                         DPRB 470
C          (1) REAL PARTS OF THE ROOTS ARE STORED IN RR(1) UP TO RR(IR)DPRB 480
C              AND CORRESPONDING COMPLEX PARTS IN RC(1) UP TO RC(IR).   DPRB 490
C          (2) ERROR MESSAGE IER=1 INDICATES POOR CONVERGENCE WITHIN    DPRB 500
C              100 ITERATION STEPS AT SOME QUADRATIC FACTORIZATION      DPRB 510
C              PERFORMED BY SUBROUTINE DPQFB.                           DPRB 520
C          (3) NO ACTION BESIDES ERROR MESSAGE IER=2 IN CASE OF A ZERO  DPRB 530
C              OR CONSTANT POLYNOMIAL. THE SAME ERROR MESSAGE IS GIVEN  DPRB 540
C              IN CASE OF AN OVERFLOW IN NORMALIZATION OF GIVEN         DPRB 550
C              POLYNOMIAL.                                              DPRB 560
C          (4) ERROR MESSAGE IER=3 INDICATES SUCCESSIVE ZERO DIVISORS   DPRB 570
C              OR OVERFLOWS OR COMPLETELY UNSATISFACTORY ACCURACY AT    DPRB 580
C              ANY QUADRATIC FACTORIZATION PERFORMED BY                 DPRB 590
C              SUBROUTINE DPQFB. IN THIS CASE CALCULATION IS BYPASSED.  DPRB 600
C              IR RECORDS THE NUMBER OF CALCULATED ROOTS.               DPRB 610
C              POL(1),...,POL(J-IR) ARE THE COEFFICIENTS OF THE         DPRB 620
C              REMAINING POLYNOMIAL, WHERE J IS THE ACTUAL NUMBER OF    DPRB 630
C              COEFFICIENTS IN VECTOR C (NORMALLY J=IC).                DPRB 640
C          (5) IF CALCULATED COEFFICIENT VECTOR HAS LESS THAN SIX       DPRB 650
C              CORRECT SIGNIFICANT DIGITS THOUGH ALL QUADRATIC          DPRB 660
C              FACTORIZATIONS SHOWED SATISFACTORY ACCURACY, THE ERROR   DPRB 670
C              MESSAGE IER=-1 IS GIVEN.                                 DPRB 680
C          (6) THE FINAL COMPARISON BETWEEN GIVEN AND CALCULATED        DPRB 690
C              COEFFICIENT VECTOR IS PERFORMED ONLY IF ALL ROOTS HAVE   DPRB 700
C              BEEN CALCULATED. IN THIS CASE THE NUMBER OF ROOTS IR IS  DPRB 710
C              EQUAL TO THE ACTUAL DEGREE OF THE POLYNOMIAL (NORMALLY   DPRB 720
C              IR=IC-1). THE MAXIMAL RELATIVE ERROR OF THE COEFFICIENT  DPRB 730
C              VECTOR IS RECORDED IN RR(IR+1).                          DPRB 740
C                                                                       DPRB 750
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DPRB 760
C          SUBROUTINE DPQFB    QUADRATIC FACTORIZATION OF A POLYNOMIAL DPRB 770
C                              BY BAIRSTOW ITERATION.                   DPRB 780
C                                                                       DPRB 790
C       METHOD                                                          DPRB 800
C          THE ROOTS OF THE POLYNOMIAL ARE CALCULATED BY MEANS OF       DPRB 810
C          SUCCESSIVE QUADRATIC FACTORIZATION PERFORMED BY BAIRSTOW     DPRB 820
C          ITERATION. X**2 IS USED AS INITIAL GUESS FOR THE FIRST       DPRB 830
C          QUADRATIC FACTOR, AND FURTHER EACH CALCULATED QUADRATIC      DPRB 840
C          FACTOR IS USED AS INITIAL GUESS FOR THE NEXT ONE. AFTER      DPRB 850
C          COMPUTATION OF ALL ROOTS THE COEFFICIENT VECTOR IS           DPRB 860
C          CALCULATED AND COMPARED WITH THE GIVEN ONE.                  DPRB 870
C          FOR REFERENCE, SEE J. H. WILKINSON, THE EVALUATION OF THE    DPRB 880
C          ZEROS OF ILL-CONDITIONED POLYNOMIALS (PART ONE AND TWO),     DPRB 890
C          NUMERISCHE MATHEMATIK, VOL.1 (1959), PP.150-180.             DPRB 900
C                                                                       DPRB 910
C     ..................................................................DPRB 920
C                                                                       DPRB 930
      SUBROUTINE DPRBM(C,IC,RR,RC,POL,IR,IER)                           DPRB 940
C                                                                       DPRB 950
C                                                                       DPRB 960
      DIMENSION C(1),RR(1),RC(1),POL(1),Q(4)                            DPRB 970
      DOUBLE PRECISION C,RR,RC,POL,Q,EPS,A,B,H,Q1,Q2                    DPRB 980
C                                                                       DPRB 990
C        TEST ON LEADING ZERO COEFFICIENTS                              DPRB1000
      EPS=1.D-6                                                         DPRB1010
      LIM=100                                                           DPRB1020
      IR=IC+1                                                           DPRB1030
    1 IR=IR-1                                                           DPRB1040
      IF(IR-1)42,42,2                                                   DPRB1050
    2 IF(C(IR))3,1,3                                                    DPRB1060
C                                                                       DPRB1070
C        WORK UP ZERO ROOTS AND NORMALIZE REMAINING POLYNOMIAL          DPRB1080
    3 IER=0                                                             DPRB1090
      J=IR                                                              DPRB1100
      L=0                                                               DPRB1110
      A=C(IR)                                                           DPRB1120
      DO 8 I=1,IR                                                       DPRB1130
      IF(L)4,4,7                                                        DPRB1140
    4 IF(C(I))6,5,6                                                     DPRB1150
    5 RR(I)=0.D0                                                        DPRB1160
      RC(I)=0.D0                                                        DPRB1170
      POL(J)=0.D0                                                       DPRB1180
      J=J-1                                                             DPRB1190
      GO TO 8                                                           DPRB1200
    6 L=1                                                               DPRB1210
      IST=I                                                             DPRB1220
      J=0                                                               DPRB1230
    7 J=J+1                                                             DPRB1240
      C(I)=C(I)/A                                                       DPRB1250
      POL(J)=C(I)                                                       DPRB1260
      CALL OVERFL(N)                                                    DPRB1270
      IF(N-2)42,8,8                                                     DPRB1280
    8 CONTINUE                                                          DPRB1290
C                                                                       DPRB1300
C        START BAIRSTOW ITERATION                                       DPRB1310
      Q1=0.D0                                                           DPRB1320
      Q2=0.D0                                                           DPRB1330
    9 IF(J-2)33,10,14                                                   DPRB1340
C                                                                       DPRB1350
C        DEGREE OF RESTPOLYNOMIAL IS EQUAL TO ONE                       DPRB1360
   10 A=POL(1)                                                          DPRB1370
      RR(IST)=-A                                                        DPRB1380
      RC(IST)=0.D0                                                      DPRB1390
      IR=IR-1                                                           DPRB1400
      Q2=0.D0                                                           DPRB1410
      IF(IR-1)13,13,11                                                  DPRB1420
   11 DO 12 I=2,IR                                                      DPRB1430
      Q1=Q2                                                             DPRB1440
      Q2=POL(I+1)                                                       DPRB1450
   12 POL(I)=A*Q2+Q1                                                    DPRB1460
   13 POL(IR+1)=A+Q2                                                    DPRB1470
      GO TO 34                                                          DPRB1480
C        THIS IS BRANCH TO COMPARISON OF COEFFICIENT VECTORS C AND POL  DPRB1490
C                                                                       DPRB1500
C        DEGREE OF RESTPOLYNOMIAL IS GREATER THAN ONE                   DPRB1510
   14 DO 22 L=1,10                                                      DPRB1520
      N=1                                                               DPRB1530
   15 Q(1)=Q1                                                           DPRB1540
      Q(2)=Q2                                                           DPRB1550
      CALL DPQFB(POL,J,Q,LIM,I)                                         DPRB1560
      IF(I)16,24,23                                                     DPRB1570
   16 IF(Q1)18,17,18                                                    DPRB1580
   17 IF(Q2)18,21,18                                                    DPRB1590
   18 GO TO (19,20,19,21),N                                             DPRB1600
   19 Q1=-Q1                                                            DPRB1610
      N=N+1                                                             DPRB1620
      GO TO 15                                                          DPRB1630
   20 Q2=-Q2                                                            DPRB1640
      N=N+1                                                             DPRB1650
      GO TO 15                                                          DPRB1660
   21 Q1=1.D0+Q1                                                        DPRB1670
   22 Q2=1.D0-Q2                                                        DPRB1680
C                                                                       DPRB1690
C        ERROR EXIT DUE TO UNSATISFACTORY RESULTS OF FACTORIZATION      DPRB1700
      IER=3                                                             DPRB1710
      IR=IR-J                                                           DPRB1720
      RETURN                                                            DPRB1730
C                                                                       DPRB1740
C        WORK UP RESULTS OF QUADRATIC FACTORIZATION                     DPRB1750
   23 IER=1                                                             DPRB1760
   24 Q1=Q(1)                                                           DPRB1770
      Q2=Q(2)                                                           DPRB1780
C                                                                       DPRB1790
C        PERFORM DIVISION OF FACTORIZED POLYNOMIAL BY QUADRATIC FACTOR  DPRB1800
      B=0.D0                                                            DPRB1810
      A=0.D0                                                            DPRB1820
      I=J                                                               DPRB1830
   25 H=-Q1*B-Q2*A+POL(I)                                               DPRB1840
      POL(I)=B                                                          DPRB1850
      B=A                                                               DPRB1860
      A=H                                                               DPRB1870
      I=I-1                                                             DPRB1880
      IF(I-2)26,26,25                                                   DPRB1890
   26 POL(2)=B                                                          DPRB1900
      POL(1)=A                                                          DPRB1910
C                                                                       DPRB1920
C        MULTIPLY POLYNOMIAL WITH CALCULATED ROOTS BY QUADRATIC FACTOR  DPRB1930
      L=IR-1                                                            DPRB1940
      IF(J-L)27,27,29                                                   DPRB1950
   27 DO 28 I=J,L                                                       DPRB1960
   28 POL(I-1)=POL(I-1)+POL(I)*Q2+POL(I+1)*Q1                           DPRB1970
   29 POL(L)=POL(L)+POL(L+1)*Q2+Q1                                      DPRB1980
      POL(IR)=POL(IR)+Q2                                                DPRB1990
C                                                                       DPRB2000
C        CALCULATE ROOT-PAIR FROM QUADRATIC FACTOR X*X+Q2*X+Q1          DPRB2010
      H=-.5D0*Q2                                                        DPRB2020
      A=H*H-Q1                                                          DPRB2030
      B=DSQRT(DABS(A))                                                  DPRB2040
      IF(A)30,30,31                                                     DPRB2050
   30 RR(IST)=H                                                         DPRB2060
      RC(IST)=B                                                         DPRB2070
      IST=IST+1                                                         DPRB2080
      RR(IST)=H                                                         DPRB2090
      RC(IST)=-B                                                        DPRB2100
      GO TO 32                                                          DPRB2110
   31 B=H+DSIGN(B,H)                                                    DPRB2120
      RR(IST)=Q1/B                                                      DPRB2130
      RC(IST)=0.D0                                                      DPRB2140
      IST=IST+1                                                         DPRB2150
      RR(IST)=B                                                         DPRB2160
      RC(IST)=0.D0                                                      DPRB2170
   32 IST=IST+1                                                         DPRB2180
      J=J-2                                                             DPRB2190
      GO TO 9                                                           DPRB2200
C                                                                       DPRB2210
C        SHIFT BACK ELEMENTS OF POL BY 1 AND COMPARE VECTORS POL AND C  DPRB2220
   33 IR=IR-1                                                           DPRB2230
   34 A=0.D0                                                            DPRB2240
      DO 38 I=1,IR                                                      DPRB2250
      Q1=C(I)                                                           DPRB2260
      Q2=POL(I+1)                                                       DPRB2270
      POL(I)=Q2                                                         DPRB2280
      IF(Q1)35,36,35                                                    DPRB2290
   35 Q2=(Q1-Q2)/Q1                                                     DPRB2300
   36 Q2=DABS(Q2)                                                       DPRB2310
      IF(Q2-A)38,38,37                                                  DPRB2320
   37 A=Q2                                                              DPRB2330
   38 CONTINUE                                                          DPRB2340
      I=IR+1                                                            DPRB2350
      POL(I)=1.D0                                                       DPRB2360
      RR(I)=A                                                           DPRB2370
      RC(I)=0.D0                                                        DPRB2380
      IF(IER)39,39,41                                                   DPRB2390
   39 IF(A-EPS)41,41,40                                                 DPRB2400
C                                                                       DPRB2410
C        WARNING DUE TO POOR ACCURACY OF CALCULATED COEFFICIENT VECTOR  DPRB2420
   40 IER=-1                                                            DPRB2430
   41 RETURN                                                            DPRB2440
C                                                                       DPRB2450
C        ERROR EXIT DUE TO DEGENERATE POLYNOMIAL OR OVERFLOW IN         DPRB2460
C        NORMALIZATION                                                  DPRB2470
   42 IER=2                                                             DPRB2480
      IR=0                                                              DPRB2490
      RETURN                                                            DPRB2500
      END                                                               DPRB2510
```

Subroutines PQFB and DPQFB

These subroutines use Bairstow's iterative method to find an approximation to a quadratic factor of a given polynomial $p(x) = c_1+c_2x+\ldots+c_{n+1}x^n$ (deg $p(x) = n \geq 2$, c_i real).

1. Mathematical background

Let $Q(x) = x^2 + Q_2x+Q_1$ be any quadratic polynomial with real coefficients. Then $p(x) = R(x)Q(x) + B(Q_1, Q_2)x + C(Q_1, Q_2)$, where $C(Q_1, Q_2) = a_1$, $B(Q_1, Q_2) = a_2$, $R(x) = a_3+a_4x+\ldots+a_{n+1}x^{n-2}$, and

$$a_i = \begin{cases} 0 & \text{if } i > n+1 \\ c_i - Q_2 a_{i+1} - Q_1 a_{i+2} & \text{if } i=n+1,\ldots,2 \\ c_1 - Q_1 a_3 & \text{if } i=1 \end{cases}$$

Of course, if $q(x)=x^2 + q_2x+q_1$ is a factor of $p(x)$, then:

$$\left.\begin{aligned} C(q_1, q_2) &= 0 \\ \text{and} \quad B(q_1, q_2) &= 0 \end{aligned}\right\}$$

Suppose that $Q(x)$ is an approximation to $q(x)$; then Q_1 and Q_2 are approximations to q_1 and q_2 respectively, and if ΔQ_1 and ΔQ_2 are such that:

$$\left.\begin{aligned} q_1 &= Q_1 + \Delta Q_1 \\ \text{and} \quad q_2 &= Q_2 + \Delta Q_2 \end{aligned}\right\}$$

Then:

$$\left.\begin{aligned} C(Q_1 + \Delta Q_1, Q_2 + \Delta Q_2) &= 0 \\ \text{and} \quad B(Q_1 + \Delta Q_1, Q_2 + \Delta Q_2) &= 0 \end{aligned}\right\} \qquad (1)$$

Instead of solving (1) for ΔQ_1 and ΔQ_2, we solve:

$$\left.\begin{aligned} & C(Q_1, Q_2) + \frac{\partial}{\partial Q_1} C(Q_1, Q_2)\Delta Q_1 \\ & \qquad + \frac{\partial}{\partial Q_2} C(Q_1, Q_2)\Delta Q_2 = 0 \\ \text{and} \quad & B(Q_1, Q_2) + \frac{\partial}{\partial Q_1} B(Q_1, Q_2)\Delta Q_1 \\ & \qquad + \frac{\partial}{\partial Q_2} B(Q_1, Q_2)\Delta Q_2 = 0 \end{aligned}\right\} \qquad (2)$$

for ΔQ_1 and ΔQ_2. Equation (2) is the linearization of (1). To solve (2) we must find $\frac{\partial}{\partial Q_i} B(Q_1, Q_2)$ and $\frac{\partial}{\partial Q_i} C(Q_1, Q_2)$, $i = 1, 2$. To do this we define

$$r_i = \begin{cases} a_i & \text{if } i>1 \\ a_1 - Q_2 B & \text{if } i=1 \end{cases}$$

and then

$$s_i = \begin{cases} 0 & \text{if } i>n+1 \\ r_i - Q_2 s_{i+1} - Q_1 s_{i+2} & \text{if } i = n+1, n, n-1, \ldots, \end{cases}$$

Now:

$$\left.\begin{aligned} \frac{\partial r_i}{\partial Q_2} &= -r_{i+1} - Q_2 \frac{\partial r_{i+1}}{\partial Q_e} - Q_1 \frac{\partial r_{i+2}}{\partial Q_2} \\ \text{and} \quad \frac{\partial r_i}{\partial Q_1} &= -r_{i+2} - Q_2 \frac{\partial r_{i+1}}{\partial Q_1} - Q_1 \frac{\partial r_{i+2}}{\partial Q_1} \end{aligned}\right\}$$

By an easy induction on i it follows that:

$$\frac{\partial r_i}{\partial Q_2} = -S_{i+1} \quad \text{and} \quad \frac{\partial r_i}{\partial Q_1} = -S_{i+2}, \qquad i=1,\ldots, n+1$$

In particular:

$$\left.\begin{aligned} \frac{\partial B}{\partial Q_1} &= \frac{\partial a_2}{\partial Q_1} = \frac{\partial r_2}{\partial Q_1} = -S_4 \\ \frac{\partial B}{\partial Q_2} &= \frac{\partial a_2}{\partial Q_2} = \frac{\partial r_2}{\partial Q_2} = -S_3 \\ \frac{\partial C}{\partial Q_1} &= \frac{\partial a_1}{\partial Q_1} = \frac{\partial r_1}{\partial Q_1} + Q_2 \frac{\partial r_2}{\partial Q_1} = \\ & \quad -S_3 - Q_2 S_4 \\ \frac{\partial C}{\partial Q_2} &= \frac{\partial a_1}{\partial Q_2} = \frac{\partial r_1}{\partial Q_2} + Q_2 \frac{\partial r_2}{\partial Q_2} + r_2 = \\ & \quad -S_2 - Q_2 S_3 + r_4 \end{aligned}\right\}$$

ınd if we let:

$$\left.\begin{aligned} C_1 &= s_4 \\ B_1 &= s_3 \\ A_1 &= s_2 - r_2 \\ A &= r_1 \end{aligned}\right\}$$

hen we can rewrite (2) as:

$$\left.\begin{aligned} (B_1 + Q_2 C_1)\,\Delta Q_1 + (A_1 + Q_2 B_1)\,\Delta Q_2 &= A + Q_2 B \\ C_1\,\Delta Q_1 + \qquad B_1\,\Delta Q_2 &= B \end{aligned}\right\}$$

/hich simplifies to:

$$\left.\begin{aligned} B_1\,\Delta Q_1 + A_1\,\Delta Q_2 &= A \\ C_1\,\Delta Q_1 + B_1\,\Delta Q_2 &= B \end{aligned}\right\} \qquad (3)$$

olving (3) we find that:

$$\left.\begin{aligned} \Delta Q_1 &= \frac{A_1 B - B_1 A}{A_1 C_1 - B_1^{\,2}} \\ \Delta Q_2 &= \frac{C_1 A - B_1 B}{A_1 C_1 - B_1^{\,2}} \end{aligned}\right\}$$

/e can now replace Q_i by $Q_i + \Delta Q_i$, $i = 1, 2$, and epeat the above procedure.

. Programming considerations

'he input consists of a vector C of polynomial oefficients (C(1) is the constant term), the dimenion IC of C, a vector Q of dimension 4 with Q(1) nd Q(2) containing initial guesses for the coeffiients of a quadratic factor, and an upper bound ,IM for the number of iterations to be performed.

First $J \le IC$ is determined such that $C(J) \ne 0$, '(K)=0 for $K > J$. If $J \le 1$, there is no computation nd the error parameter IER is set to -1. If $J > 1$, ıe polynomial is normalized by dividing all the oefficients of C by C(J), the coefficient of the eading term. If overflow occurs during the norıalization, computation is terminated and IER is et to -1. If no overflow occurs, the degree (J-1) of the polynomial p(x) is tested; if p is of degree 1, there is no further computation and IER is set to -2; if p is of degree 2, Q(1) and Q(2) are set to C(1) and C(2) respectively, Q(3) and Q(4) are set to zero, IER is set to zero, and computation is terminated; if p is of degree greater than 2, the main part of the subroutine--the Bairstow iterative procedure--is entered.

Every approximation $Q(x) = x^2 + Q_2 x + Q_1$ to a quadratic factor of p has associated with it a constant CC, called the norm of the modified linear remainder and hereafter referred to as the norm. It is given by $CC = CA \cdot |A| + CB \cdot |B|$ where:

$$CA = \begin{cases} 1 & \text{if } |C(1)| \le |C(2)| \\ \left|\dfrac{C(2)}{C(1)}\right| & \text{if } |C(1)| > |C(2)| \end{cases}$$

and

$$CB = \begin{cases} \left|\dfrac{C(1)}{C(2)}\right| & \text{if } |C(1)| < |C(2)| \\ 1 & \text{if } |C(1)| \ge |C(2)| \end{cases}$$

Since there is no guarantee that the Bairstow method will converge, several accuracy tests are performed at the end of each iteration. Depending on the outcome of these tests, which in part involve CC, either the computation is terminated or another iteration is performed.

Letting:

$$EPS = \begin{cases} 10^{-6} & \text{in single precision} \\ 10^{-14} & \text{in double precision} \end{cases}$$

and:

$$EPS1 = \begin{cases} 10^{-3} & \text{in single precision} \\ 10^{-16} & \text{in double precision} \end{cases}$$

the tests are described below.

a. If $|A| \le EPS^* |C(1)|$ and $|B| \le EPS^* |C(2)|$, or if $|\Delta Q_1| \le EPS^* |Q_1|$ and $|\Delta Q_2| \le EPS^* |Q_2|$, computation is terminated after setting IER = 0, $Q(1) = Q_1$, $Q(2) = Q_2$, $Q(3) = A$, and $Q(4) = B$.

b. If $|\Delta Q_1| \le EPS1^* |Q_1|$ and $|\Delta Q_2| \le EPS1^* |Q_2|$, and if $|\Delta Q_1/Q_1|$ or $|\Delta Q_2/Q_2|$ is not smaller than its value for the previous iteration (this is generally due to roundoff errors), computation is terminated after setting IER = 0, $Q(1) = QQ_1$, $Q(2) = QQ_2$, $Q(3) = AA$, and $Q(4) = BB$ where QQ_1, QQ_2, AA, and BB are the values of Q_1, Q_2, A, and B belonging to

a previous iteration--the iteration with the smallest norm.

c. If no convergence occurs within LIM iterations and if the last norm is not greater than CD/10, where CD is the norm associated with the quadratic factor x^2, results are generated as in b. with the exception that IER is set to 1. If the last norm is greater than CD/10:

(1) If the initial guess for the quadratic factor was x^2, the results are generated as in b. with the exception that IER is set to -3.

(2) If the initial guess for the quadratic factor was not x^2, the subroutine restarts with x^2 as the initial guess.

d. If exponential overflow or division by 0 occurs during any iteration, results are generated as in b. with the exception that IER is set to -3.

For reference see:

(1) J. H. Wilkinson, "The Evaluation of the Zeros of Ill-Conditioned Polynomials (Part I and II)," Numerische Mathematik, vol. 1 (1959), pp. 150-180.

(2) F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 472-476.


```
C                                                                       PQFB  10
C     ...................................................................PQFB  20
C                                                                       PQFB  30
C        SUBROUTINE PQFB                                                PQFB  40
C                                                                       PQFB  50
C        PURPOSE                                                        PQFB  60
C           TO FIND AN APPROXIMATION Q(X)=Q1+Q2*X+X*X TO A QUADRATIC    PQFB  70
C           FACTOR OF A GIVEN POLYNOMIAL P(X) WITH REAL COEFFICIENTS.   PQFB  80
C                                                                       PQFB  90
C        USAGE                                                          PQFB 100
C           CALL PQFB(C,IC,Q,LIM,IER)                                   PQFB 110
C                                                                       PQFB 120
C        DESCRIPTION OF PARAMETERS                                      PQFB 130
C           C   - INPUT VECTOR CONTAINING THE COEFFICIENTS OF P(X) -    PQFB 140
C                 C(1) IS THE CONSTANT TERM (DIMENSION IC)              PQFB 150
C           IC  - DIMENSION OF C                                        PQFB 160
C           Q   - VECTOR OF DIMENSION 4 - ON INPUT Q(1) AND Q(2) MUST   PQFB 170
C                 CONTAIN INITIAL GUESSES FOR Q1 AND Q2 - ON RETURN Q(1)PQFB 180
C                 AND Q(2) CONTAIN THE REFINED COEFFICIENTS Q1 AND Q2 OFPQFB 190
C                 Q(X), WHILE Q(3) AND Q(4) CONTAIN THE COEFFICIENTS A  PQFB 200
C                 AND B OF A+B*X, WHICH IS THE REMAINDER OF THE QUOTIENTPQFB 210
C                 OF P(X) BY Q(X)                                       PQFB 220
C           LIM - INPUT VALUE SPECIFYING THE MAXIMUM NUMBER OF          PQFB 230
C                 ITERATIONS TO BE PERFORMED                            PQFB 240
C           IER - RESULTING ERROR PARAMETER (SEE REMARKS)               PQFB 250
C                 IER= 0 - NO ERROR                                     PQFB 260
C                 IER= 1 - NO CONVERGENCE WITHIN LIM ITERATIONS         PQFB 270
C                 IER=-1 - THE POLYNOMIAL P(X) IS CONSTANT OR UNDEFINED PQFB 280
C                        - OR OVERFLOW OCCURRED IN NORMALIZING P(X)     PQFB 290
C                 IER=-2 - THE POLYNOMIAL P(X) IS OF DEGREE 1           PQFB 300
C                 IER=-3 - NO FURTHER REFINEMENT OF THE APPROXIMATION TOPQFB 310
C                          A QUADRATIC FACTOR IS FEASIBLE, DUE TO EITHERPQFB 320
C                          DIVISION BY 0, OVERFLOW OR AN INITIAL GUESS  PQFB 330
C                          THAT IS NOT SUFFICIENTLY CLOSE TO A FACTOR OFPQFB 340
C                          P(X)                                         PQFB 350
C                                                                       PQFB 360
C        REMARKS                                                        PQFB 370
C           (1)  IF IER=-1 THERE IS NO COMPUTATION OTHER THAN THE       PQFB 380
C                POSSIBLE NORMALIZATION OF C.                           PQFB 390
C           (2)  IF IER=-2 THERE IS NO COMPUTATION OTHER THAN THE       PQFB 400
C                NORMALIZATION OF C.                                    PQFB 410
C           (3)  IF IER =-3  IT IS SUGGESTED THAT A NEW INITIAL GUESS BEPQFB 420
C                MADE FOR A QUADRATIC FACTOR.  Q, HOWEVER, WILL CONTAIN PQFB 430
C                THE VALUES ASSOCIATED WITH THE ITERATION THAT YIELDED  PQFB 440
C                THE SMALLEST NORM OF THE MODIFIED LINEAR REMAINDER.    PQFB 450
C           (4)  IF IER=1, THEN, ALTHOUGH THE NUMBER OF ITERATIONS LIM  PQFB 460
C                WAS TOO SMALL TO INDICATE CONVERGENCE, NO OTHER PROB-  PQFB 470
C                LEMS HAVE BEEN DETECTED, AND Q WILL CONTAIN THE VALUES PQFB 480
C                ASSOCIATED WITH THE ITERATION THAT YIELDED THE SMALLESTPQFB 490
C                NORM OF THE MODIFIED LINEAR REMAINDER.                 PQFB 500
C           (5)  FOR COMPLETE DETAIL SEE THE DOCUMENTATION FOR          PQFB 510
C                SUBROUTINES PQFB AND DPQFB.                            PQFB 520
C                                                                       PQFB 530
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PQFB 540
C           NONE                                                        PQFB 550
C                                                                       PQFB 560
C                                                                       PQFB 570
C        METHOD                                                         PQFB 580
C           COMPUTATION IS BASED ON BAIRSTOW'S ITERATIVE METHOD.  (SEE  PQFB 590
C           WILKINSON, J.H., THE EVALUATION OF THE ZEROS OF ILL-CON-    PQFB 600
C           DITIONED POLYNOMIALS (PART ONE AND TWO), NUMERISCHE MATHE-  PQFB 610
C           MATIK, VOL.1 (1959), PP. 150-180, OR HILDEBRAND, F.B.,      PQFB 620
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ PQFB 630
C           TORONTO/LONDON, 1956, PP. 472-476.)                         PQFB 640
C                                                                       PQFB 650
C     ...................................................................PQFB 660
```

```
      SUBROUTINE PQFB(C,IC,Q,LIM,IER)
C
C
      DIMENSION C(1),Q(1)
C
C        TEST ON LEADING ZERO COEFFICIENTS
      IER=0
      J=IC+1
    1 J=J-1
      IF(J-1)40,40,2
    2 IF(C(J))3,1,3
C
C        NORMALIZATION OF REMAINING COEFFICIENTS
    3 A=C(J)
      IF(A-1.)4,6,4
    4 DO 5 I=1,J
      C(I)=C(I)/A
      CALL OVERFL(N)
      IF(N-2)40,5,5
    5 CONTINUE
C
C        TEST ON NECESSITY OF BAIRSTOW ITERATION
    6 IF(J-3)41,38,7
C
C        PREPARE BAIRSTOW ITERATION
    7 EPS=1.E-6
      EPS1=1.E-3
      L=0
      LL=0
      Q1=Q(1)
      Q2=Q(2)
      QQ1=0.
      QQ2=0.
      AA=C(1)
      BB=C(2)
      CB=ABS(AA)
      CA=ABS(BB)
      IF(CB-CA)8,9,10
    8 CC=CB+CB
      CB=CB/CA
      CA=1.
      GO TO 11
    9 CC=CA+CA
      CA=1.
      CB=1.
      GO TO 11
   10 CC=CA+CA
      CA=CA/CB
      CB=1.
   11 CD=CC*.1
C
C        START BAIRSTOW ITERATION
C        PREPARE NESTED MULTIPLICATION
   12 A=0.
      B=A
      A1=A
      B1=A
      I=J
      QQ01=Q1
      QQQ2=Q2
      DQ1=HH
      DQ2=H
C
C        START NESTED MULTIPLICATION
   13 H=-Q1*B-Q2*A+C(I)
      CALL OVERFL(N)
      IF(N-2)42,14,14
   14 B=A
      A=H
      I=I-1
      IF(I-1)18,15,16
   15 H=0.
   16 H=-Q1*B1-Q2*A1+H
      CALL OVERFL(N)
      IF(N-2)42,17,17
   17 C1=B1
      B1=A1
      A1=H
      GO TO 13
C        END OF NESTED MULTIPLICATION
C
C        TEST ON SATISFACTORY ACCURACY
   18 H=CA*ABS(A1)+CB*ABS(B)
      IF(LL)19,19,39
   19 L=L+1
      IF(ABS(A)-EPS*ABS(C(1)))20,20,21
   20 IF(ABS(B)-EPS*ABS(C(2)))39,39,21
C
C        TEST ON LINEAR REMAINDER OF MINIMUM NORM
   21 IF(H-CC)22,22,23
   22 AA=A
      BB=B
      CC=H
      QQ1=Q1
      QQ2=Q2
C
C        TEST ON LAST ITERATION STEP
   23 IF(L-LIM)28,28,24
C
C        TEST ON RESTART OF BAIRSTOW ITERATION WITH ZERO INITIAL GUESS
   24 IF(H-CD)43,43,25
   25 IF(Q(1))27,26,27
   26 IF(Q(2))27,42,27
   27 Q(1)=0.
      Q(2)=0.
      GO TO 7
C
C        PERFORM ITERATION STEP
   28 HH=AMAX1(ABS(A1),ABS(B1),ABS(C1))
      IF(HH)42,42,29
   29 A1=A1/HH
      B1=B1/HH
      C1=C1/HH
      H=A1*C1-B1*B1
      IF(H)30,42,30
   30 A=A/HH
      B=B/HH
      HH=(B*A1-A*B1)/H
      H=(A*C1-B*B1)/H
      Q1=Q1+HH
      Q2=Q2+H
C        END OF ITERATION STEP
C
C        TEST ON SATISFACTORY RELATIVE ERROR OF ITERATED VALUES
      IF(ABS(HH)-EPS*ABS(Q1))31,31,33
   31 IF(ABS(H)-EPS*ABS(Q2))32,32,33
   32 LL=1
      GO TO 12
C
C        TEST ON DECREASING RELATIVE ERRORS
   33 IF(L-1)12,12,34
   34 IF(ABS(HH)-EPS1*ABS(Q1))35,35,12
   35 IF(ABS(H)-EPS1*ABS(Q2))36,36,12
   36 IF(ABS(QQQ1*HH)-ABS(Q1*DQ1))37,44,44
   37 IF(ABS(QQQ2*H)-ABS(Q2*DQ2))12,44,44
C        END OF BAIRSTOW ITERATION
C
C        EXIT IN CASE OF QUADRATIC POLYNOMIAL
   38 Q(1)=C(1)
      Q(2)=C(2)
      Q(3)=0.
      Q(4)=0.
      RETURN
C
C        EXIT IN CASE OF SUFFICIENT ACCURACY
   39 Q(1)=Q1
      Q(2)=Q2
      Q(3)=A
      Q(4)=B
      RETURN
C
C        ERROR EXIT IN CASE OF ZERO OR CONSTANT POLYNOMIAL
   40 IER=-1
      RETURN
C
C        ERROR EXIT IN CASE OF LINEAR POLYNOMIAL
   41 IER=-2
      RETURN
C
C        ERROR EXIT IN CASE OF NONREFINED QUADRATIC FACTOR
   42 IER=-3
      GO TO 44
C
C        ERROR EXIT IN CASE OF UNSATISFACTORY ACCURACY
   43 IER=1
   44 Q(1)=QQ1
      Q(2)=QQ2
      Q(3)=AA
      Q(4)=BB
      RETURN
      END
```

```
C                                                                       DPQF  10
C     ..................................................................DPQF  20
C                                                                       DPQF  30
C        SUBROUTINE DPQFB                                               DPQF  40
C                                                                       DPQF  50
C        PURPOSE                                                        DPQF  60
C           TO FIND AN APPROXIMATION Q(X)=Q1+Q2*Y*X*X TO A QUADRATIC    DPQF  70
C           FACTOR OF A GIVEN POLYNOMIAL P(X) WITH REAL COEFFICIENTS.   DPQF  80
C                                                                       DPQF  90
C        USAGE                                                          DPQF 100
C           CALL DPQFB(C,IC,Q,LIM,IER)                                  DPQF 110
C                                                                       DPQF 120
C        DESCRIPTION OF PARAMETERS                                      DPQF 130
C           C   - DOUBLE PRECISION INPUT VECTOR CONTAINING THE          DPQF 140
C                 COEFFICIENTS OF P(X) - C(1) IS THE CONSTANT TERM      DPQF 150
C                 (DIMENSION IC)                                        DPQF 160
C           IC  - DIMENSION OF C                                        DPQF 170
C           Q   - DOUBLE PRECISION VECTOR OF DIMENSION 4 - ON INPUT Q(1)DPQF 180
C                 AND Q(2) CONTAIN INITIAL GUESSES FOR Q1 AND Q2 - ON   DPQF 190
C                 RETURN Q(1) AND Q(2) CONTAIN THE REFINED COEFFICIENTS DPQF 200
C                 Q1 AND Q2 OF Q(X), WHILE Q(3) AND Q(4) CONTAIN THE    DPQF 210
C                 COEFFICIENTS A AND B OF A*B*X, WHICH IS THE REMAINDER DPQF 220
C                 OF THE QUOTIENT OF P(X) BY Q(X)                       DPQF 230
C           LIM - INPUT VALUE SPECIFYING THE MAXIMUM NUMBER OF          DPQF 240
C                 ITERATIONS TO BE PERFORMED                            DPQF 250
C           IER - RESULTING ERROR PARAMETER (SEE REMARKS)               DPQF 260
C                 IER= 0 - NO ERROR                                     DPQF 270
C                 IER= 1 - NO CONVERGENCE WITHIN LIM ITERATIONS         DPQF 280
C                 IER=-1 - THE POLYNOMIAL P(X) IS CONSTANT OR UNDEFINED DPQF 290
C                          - OR OVERFLOW OCCURRED IN NORMALIZING P(X)   DPQF 300
C                 IER=-2 - THE POLYNOMIAL P(X) IS OF DEGREE 1           DPQF 310
C                 IER=-3 - NO FURTHER REFINEMENT OF THE APPROXIMATION TODPQF 320
C                          A QUADRATIC FACTOR IS FEASIBLE, DUE TO EITHERDPQF 330
C                          DIVISION BY 0, OVERFLOW OR AN INITIAL GUESS  DPQF 340
C                          THAT IS NOT SUFFICIENTLY CLOSE TO A FACTOR OFDPQF 350
C                          P(X)                                         DPQF 360
C                                                                       DPQF 370
C        REMARKS                                                        DPQF 380
C           (1)  IF IER=-1 THERE IS NO COMPUTATION OTHER THAN THE       DPQF 390
C                POSSIBLE NORMALIZATION OF C.                           DPQF 400
C           (2)  IF IER=-2 THERE IS NO COMPUTATION OTHER THAN THE       DPQF 410
C                NORMALIZATION OF C.                                    DPQF 420
C           (3)  IF IER =-3  IT IS SUGGESTED THAT A NEW INITIAL GUESS BEDPQF 430
C                MADE FOR A QUADRATIC FACTOR.  Q, HOWEVER, WILL CONTAIN DPQF 440
C                THE VALUES ASSOCIATED WITH THE ITERATION THAT YIELDED  DPQF 450
C                THE SMALLEST NORM OF THE MODIFIED LINEAR REMAINDER.    DPQF 460
C           (4)  IF IER=1, THEN, ALTHOUGH THE NUMBER OF ITERATIONS LIM  DPQF 470
C                WAS TOO SMALL TO INDICATE CONVERGENCE, NO OTHER PROB-  DPQF 480
C                LEMS HAVE BEEN DETECTED, AND Q WILL CONTAIN THE VALUES DPQF 490
C                ASSOCIATED WITH THE ITERATION THAT YIELDED THE SMALLESTDPQF 500
C                NORM OF THE MODIFIED LINEAR REMAINDER.                 DPQF 510
C           (5)  FOR COMPLETE DETAIL SEE THE DOCUMENTATION FOR          DPQF 520
C                SUBROUTINES PQFB AND DPQFB.                            DPQF 530
C                                                                       DPQF 540
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DPQF 550
C           NONE                                                        DPQF 560
C                                                                       DPQF 570
C        METHOD                                                         DPQF 580
C           COMPUTATION IS BASED ON BAIRSTOW'S ITERATIVE METHOD.  (SEE  DPQF 590
C           WILKINSON, J.H., THE EVALUATION OF THE ZEROS OF ILL-CON-    DPQF 600
C           DITIONED POLYNOMIALS (PART ONE AND TWO), NUMERISCHE MATHE-  DPQF 610
C           MATIK, VOL.1 (1959), PP. 150-180, OR HILDEBRAND, F.B.,      DPQF 620
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ DPQF 630
C           TORONTO/LONDON, 1956, PP. 472-476.)                         DPQF 640
C                                                                       DPQF 650
C     ..................................................................DPQF 660
C                                                                       DPQF 670
      SUBROUTINE DPQFB(C,IC,Q,LIM,IER)                                  DPQF 680
C                                                                       DPQF 690
C                                                                       DPQF 700
      DIMENSION C(1),Q(1)                                               DPQF 710
      DOUBLE PRECISION A,B,AA,BB,CA,CB,CC,CD,A1,B1,C1,H,HH,Q1,Q2,QQ1,   DPQF 720
     1                 QQ2,QQQ1,QQQ2,DQ1,DQ2,EPS,EPS1,C,Q               DPQF 730
C                                                                       DPQF 740
C        TEST ON LEADING ZERO COEFFICIENTS                              DPQF 750
      IER=0                                                             DPQF 760
      J=IC+1                                                            DPQF 770
    1 J=J-1                                                             DPQF 780
```

```
      IF(J-1)40,40,2
    2 IF(C(J))3,1,3
C
C        NORMALIZATION OF REMAINING COEFFICIENTS
    3 A=C(J)
      IF(A-1.D014,6,4
    4 DO 5 I=1,J
      C(I)=C(I)/A
      CALL OVERFL(N)
      IF(N-2)40,5,5
    5 CONTINUE
C
C        TEST ON NECESSITY OF BAIRSTOW ITERATION
    6 IF(J-3)41,38,7
C
C        PREPARE BAIRSTOW ITERATION
    7 EPS=1.D-14
      EPS1=1.D-6
      L=0
      LL=0
      Q1=Q(1)
      Q2=Q(2)
      QQ1=0.D0
      QQ2=0.D0
      AA=C(1)
      BB=C(2)
      CB=DABS(AA)
      CA=DABS(BB)
      IF(CB-CA)8,9,10
    8 CC=CB+CB
      CB=CB/CA
      CA=1.D0
      GO TO 11
    9 CC=CA+CA
      CA=1.D0
      CB=1.D0
      GO TO 11
   10 CC=CA+CA
      CA=CA/CB
      CB=1.D0
   11 CD=CC*.1D0
C
C        START BAIRSTOW ITERATION
C        PREPARE NESTED MULTIPLICATION
   12 A=0.D0
      B=A
      A1=A
      B1=A
      I=J
      QQQ1=Q1
      QQQ2=Q2
      DQ1=HH
      DQ2=H
C
C        START NESTED MULTIPLICATION
   13 H=-Q1*B-Q2*A+C(I)
      CALL OVERFL(N)
      IF(N-2)42,14,14
   14 B=A
      A=H
      I=I-1
      IF(I-1)18,15,16
   15 H=0.D0
   16 H=-Q1*B1-Q2*A1+H
      CALL OVERFL(N)
      IF(N-2)42,17,17
   17 C1=B1
      B1=A1
      A1=H
      GO TO 13
C        END OF NESTED MULTIPLICATION
C
C        TEST ON SATISFACTORY ACCURACY
   18 H=CA*DABS(A)+CB*DABS(B)
      IF(LL)19,19,39
   19 L=L+1
      IF(DABS(A)-EPS*DABS(C(1)))20,20,21
   20 IF(DABS(B)-EPS*DABS(C(2)))39,39,21
C
C        TEST ON LINEAR REMAINDER OF MINIMUM NORM
   21 IF(H-CC)22,22,23
   22 AA=A
      BB=B
      CC=H
      QQ1=Q1
      QQ2=Q2
C
C        TEST ON LAST ITERATION STEP
   23 IF(L-LIM)28,28,24
C
C        TEST ON RESTART OF BAIRSTOW ITERATION WITH ZERO INITIAL GUESS
   24 IF(H-CD)43,43,25
   25 IF(Q(1))27,26,27
   26 IF(Q(2))27,42,27
   27 Q(1)=0.D0
      Q(2)=0.D0
      GO TO 7
C
C        PERFORM ITERATION STEP
   28 HH=DMAX1(DABS(A1),DABS(B1),DABS(C1))
      IF(HH)42,42,29
   29 A1=A1/HH
      B1=B1/HH
      C1=C1/HH
      H=A1*C1-B1*B1
      IF(H)30,42,30
   30 A=A/HH
      B=B/HH
      HH=(B*A1-A*B1)/H
      H=(A*C1-B*B1)/H
      Q1=Q1+HH
      Q2=Q2+H
C        END OF ITERATION STEP
C
C        TEST ON SATISFACTORY RELATIVE ERROR OF ITERATED VALUES
      IF(DABS(HH)-EPS*DABS(Q1))31,31,33
   31 IF(DABS(H)-EPS*DABS(Q2))32,32,33
   32 LL=1
      GO TO 12
C
C        TEST ON DECREASING RELATIVE ERRORS
   33 IF(L-1)12,12,34
   34 IF(DABS(HH)-EPS1*DABS(Q1))35,35,12
   35 IF(DABS(H)-EPS1*DABS(Q2))36,36,12
   36 IF(DABS(QQQ1*HH)-DABS(Q1*DQ1))37,44,44
   37 IF(DABS(QQQ2*H)-DABS(Q2*DQ2))12,44,44
C        END OF BAIRSTOW ITERATION
C
C        EXIT IN CASE OF QUADRATIC POLYNOMIAL
   38 Q(1)=C(1)
      Q(2)=C(2)
      Q(3)=0.D0
      Q(4)=0.D0
      RETURN
C
C        EXIT IN CASE OF SUFFICIENT ACCURACY
   39 Q(1)=Q1
      Q(2)=Q2
      Q(3)=A
      Q(4)=B
      RETURN
C
C        ERROR EXIT IN CASE OF ZERO OR CONSTANT POLYNOMIAL
   40 IER=-1
      RETURN
C
C        ERROR EXIT IN CASE OF LINEAR POLYNOMIAL
   41 IER=-2
      RETURN
C
C        ERROR EXIT IN CASE OF NONREFINED QUADRATIC FACTOR
   42 IER=-3
      GO TO 44
C
C        ERROR EXIT IN CASE OF UNSATISFACTORY ACCURACY
   43 IER=1
   44 Q(1)=QQ1
      Q(2)=QQ2
      Q(3)=AA
      Q(4)=BB
      RETURN
      END
```

Polynomials: Special Types

Subroutines CNP and DCNP

These subroutines compute the values of the Chebyshev polynomials for a given argument x and orders zero up to N. The Chebyshev polynomial $T_n(x)$ satisfies the recurrence equation

$$T_{n+1}(x) = 2\,x \cdot T_n(x) - T_{n-1}(x)$$

with starting values $T_0(x) = 1$, $T_1(x) = x$. From this equation it follows immediately that roundoff errors grow at worst linearly under the tacit assumption that the argument has an absolute value less than one.

If e_{n+r} is the error in $T_{n+r}(x)$ due to a single roundoff error e in $T_n(x)$:

$$e_{n+r+1} = 2\,x \cdot e_{n+r} - e_{n+r-1}$$

with initial conditions $e_n = e$, $e_{n-1} = 0$.

The solution of this difference equation has its maximum for $|x| = 1$:

$$e_{n-1} = 0,\ e_n = e,\ \left|e_{n+1}\right| = 2\,e,\ \ldots,\ \left|e_{n+r}\right| = (r+1)e$$

The order is assumed to be zero for negative values of N.

```
C                                                                       CNP   10
C     ...................................................................CNP   20
C                                                                       CNP   30
C        SUBROUTINE CNP                                                 CNP   40
C                                                                       CNP   50
C        PURPOSE                                                        CNP   60
C           COMPUTE THE VALUES OF THE CHEBYSHEV POLYNOMIALS T(N,X)      CNP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  CNP   80
C                                                                       CNP   90
C        USAGE                                                          CNP  100
C           CALL CNP(Y,X,N)                                             CNP  110
C                                                                       CNP  120
C        DESCRIPTION OF PARAMETERS                                      CNP  130
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESCNP  140
C                    OF CHEBYSHEV POLYNOMIALS OF ORDER 0 UP TO N        CNP  150
C                    FOR GIVEN ARGUMENT X.                              CNP  160
C           Y      - RESULT VALUE                                      CNP  170
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER          CNP  180
C           X      - ARGUMENT OF CHEBYSHEV POLYNOMIAL                  CNP  190
C           N      - ORDER OF CHEBYSHEV POLYNOMIAL                     CNP  200
C                                                                       CNP  210
C        REMARKS                                                        CNP  220
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     CNP  230
C                                                                       CNP  240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CNP  250
C           NONE                                                        CNP  260
C                                                                       CNP  270
C        METHOD                                                         CNP  280
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          CNP  290
C           CHEBYSHEV POLYNOMIALS T(N,X)                                CNP  300
C           T(N+1,X)=2*X*T(N,X)-T(N-1,X),                               CNP  310
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              CNP  320
C           THE SECOND IS THE ARGUMENT.                                 CNP  330
C           STARTING VALUES ARE T(0,X)=1, T(1,X)=X.                     CNP  340
C                                                                       CNP  350
C     ...................................................................CNP  360
C                                                                       CNP  370
      SUBROUTINE CNP(Y,X,N)                                             CNP  380
C                                                                       CNP  390
      DIMENSION Y(1)                                                    CNP  400
      Y(1)=1.                                                           CNP  410
      IF(N)1,1,2                                                        CNP  420
    1 RETURN                                                            CNP  430
C                                                                       CNP  440
    2 Y(2)=X                                                            CNP  450
      IF(N-1)1,1,3                                                      CNP  460
C                                                                       CNP  470
C        INITIALIZATION                                                 CNP  480
    3 F=X+X                                                             CNP  490
C                                                                       CNP  500
      DO 4 I=2,N                                                        CNP  510
    4 Y(I+1)=F*Y(I)-Y(I-1)                                              CNP  520
      RETURN                                                            CNP  530
      END                                                               CNP  540
```

```
C                                                                       DNP   10
C     ...................................................................DNP   20
C                                                                       DNP   30
C        SUBROUTINE DCNP                                                DNP   40
C                                                                       DNP   50
C        PURPOSE                                                        DNP   60
C           COMPUTE THE VALUES OF THE CHEBYSHEV POLYNOMIALS T(N,X)      DNP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  DNP   80
C                                                                       DNP   90
C        USAGE                                                          DNP  100
C           CALL DCNP,Y,X,N)                                            DNP  110
C                                                                       DNP  120
C        DESCRIPTION OF PARAMETERS                                      DNP  130
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESDNP  140
C                    OF CHEBYSHEV POLYNOMIALS OF ORDER 0 UP TO N        DNP  150
C                    FOR GIVEN ARGUMENT X.                              DNP  160
C                    DOUBLE PRECISION VECTOR.                           DNP  170
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER          DNP  180
C           Y      - RESULT VALUE                                      DNP  190
C                    DOUBLE PRECISION VARIABLE.                         DNP  200
C           X      - ARGUMENT OF CHEBYSHEV POLYNOMIAL                  DNP  210
C           N      - ORDER OF CHEBYSHEV POLYNOMIAL                     DNP  220
C                                                                       DNP  230
C        REMARKS                                                        DNP  240
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     DNP  250
C                                                                       DNP  260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DNP  270
C           NONE                                                        DNP  280
C                                                                       DNP  290
C        METHOD                                                         DNP  300
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          DNP  310
C           CHEBYSHEV POLYNOMIALS T(N,X)                                DNP  320
C           T(N+1,X)=2*X*T(N,X)-T(N-1,X),                               DNP  330
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              DNP  340
C           THE SECOND IS THE ARGUMENT.                                 DNP  350
C           STARTING VALUES ARE T(0,X)=1, T(1,X)=X.                     DNP  360
C                                                                       DNP  370
C     ...................................................................DNP  380
C                                                                       DNP  390
      SUBROUTINE DCNP(Y,X,N)                                            DNP  400
C                                                                       DNP  410
      DIMENSION Y(1)                                                    DNP  420
      DOUBLE PRECISION Y,X,F                                            DNP  430
C                                                                       DNP  440
      Y(1)=1.D0                                                         DNP  450
      IF(N)1,1,2                                                        DNP  460
    1 RETURN                                                            DNP  470
C                                                                       DNP  480
    2 Y(2)=X                                                            DNP  490
      IF(N-1)1,1,3                                                      DNP  500
C                                                                       DNP  510
C        INITIALIZATION                                                 DNP  520
    3 F=X+X                                                             DNP  530
C                                                                       DNP  540
      DO 4 I=2,N                                                        DNP  550
    4 Y(I+1)=F*Y(I)-Y(I-1)                                              DNP  560
      RETURN                                                            DNP  570
      END                                                               DNP  580
```

Subroutines CNPS and DCNPS

These subroutines compute the value of a series expansion in Chebyshev polynomials. The Chebyshev polynomial $T_n(x)$ satisfies the recurrence equation

$$T_{n+1}(x) = 2x \cdot T_n(x) - T_{n-1}(x)$$

with starting values $T_0(x) = 1$, $T_1(x) = x$.

An n-term expansion in Chebyshev polynomials with coefficient vector $C = (c_1, \ldots, c_n)$ is evaluated by means of a backward iteration scheme:

$$\text{Arg} = 2x, \quad H_1 = 0, \quad H_0 = 0$$

$$\left.\begin{aligned} H_2 &= H_1 \\ H_1 &= H_0 \\ H_0 &= \text{Arg} \cdot H_1 - H_2 + c_{n-i+1} \\ Y &= (c_1 - H_2 + H_0)/2. \end{aligned}\right\} \text{ for } i = 1, \ldots, n$$

This gives result $Y(x) = \sum_{i=1}^{n} c_i T_{i-1}(x)$.

Calculation is bypassed in case of a nonpositive value of the dimension n.

```
C                                                                       CNPS  10
C     ...................................................................CNPS  20
C                                                                       CNPS  30
C        SUBROUTINE CNPS                                                CNPS  40
C                                                                       CNPS  50
C        PURPOSE                                                        CNPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN CHEBYSHEV      CNPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. CNPS  80
C                                                                       CNPS  90
C        USAGE                                                          CNPS 100
C           CALL CNPS(Y,X,C,N)                                          CNPS 110
C                                                                       CNPS 120
C        DESCRIPTION OF PARAMETERS                                      CNPS 130
C           Y      - RESULT VALUE                                       CNPS 140
C           X      - ARGUMENT VALUE                                     CNPS 150
C           C      - COEFFICIENT VECTOR OF GIVEN EXPANSION              CNPS 160
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          CNPS 170
C           N      - DIMENSION OF COEFFICIENT VECTOR C                  CNPS 180
C                                                                       CNPS 190
C        REMARKS                                                        CNPS 200
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 CNPS 210
C                                                                       CNPS 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CNPS 230
C           NONE                                                        CNPS 240
C                                                                       CNPS 250
C        METHOD                                                         CNPS 260
C           DEFINITION                                                  CNPS 270
C           Y=SUM(C(I)*T(I-1,X), SUMMED OVER I FROM 1 TO N).            CNPS 280
C           EVALUATION IS DONE BY MEANS OF BACKWARD RECURSION           CNPS 290
C           USING THE RECURRENCE EQUATION FOR CHEBYSHEV POLYNOMIALS     CNPS 300
C           T(N+1,X)=2*X*T(N,X)-T(N-1,X).                               CNPS 310
C                                                                       CNPS 320
C     ...................................................................CNPS 330
C                                                                       CNPS 340
      SUBROUTINE CNPS(Y,X,C,N)                                          CNPS 350
C                                                                       CNPS 360
      DIMENSION C(1)                                                    CNPS 370
C                                                                       CNPS 380
C        TEST OF DIMENSION                                              CNPS 390
      IF(N)1,1,2                                                        CNPS 400
    1 RETURN                                                            CNPS 410
C                                                                       CNPS 420
    2 IF(N-2)3,4,4                                                      CNPS 430
    3 Y=C(1)                                                            CNPS 440
      RETURN                                                            CNPS 450
C                                                                       CNPS 460
C        INITIALIZATION                                                 CNPS 470
    4 ARG=X+X                                                           CNPS 480
      H1=0.                                                             CNPS 490
      H0=0.                                                             CNPS 500
C                                                                       CNPS 510
      DO 5 I=1,N                                                        CNPS 520
      K=N-I                                                             CNPS 530
      H2=H1                                                             CNPS 540
      H1=H0                                                             CNPS 550
    5 H0=ARG*H1-H2+C(K+1)                                               CNPS 560
      Y=0.5*(C(1)-H2+H0)                                                CNPS 570
      RETURN                                                            CNPS 580
      END                                                               CNPS 590
```

```
C                                                                       DNPS  10
C     ...................................................................DNPS  20
C                                                                       DNPS  30
C        SUBROUTINE DCNPS                                               DNPS  40
C                                                                       DNPS  50
C        PURPOSE                                                        DNPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN CHEBYSHEV      DNPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. DNPS  80
C                                                                       DNPS  90
C        USAGE                                                          DNPS 100
C           CALL DCNPS(Y,X,C,N)                                         DNPS 110
C                                                                       DNPS 120
C        DESCRIPTION OF PARAMETERS                                      DNPS 130
C           Y      - RESULT VALUE                                       DNPS 140
C                    DOUBLE PRECISION VARIABLE                          DNPS 150
C           X      - ARGUMENT VALUE                                     DNPS 160
C                    DOUBLE PRECISION VARIABLE                          DNPS 170
C           C      - COEFFICIENT VECTOR OF GIVEN EXPANSION              DNPS 180
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DNPS 190
C                    DOUBLE PRECISION VECTOR                            DNPS 200
C           N      - DIMENSION OF COEFFICIENT VECTOR C                  DNPS 210
C                                                                       DNPS 220
C        REMARKS                                                        DNPS 230
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 DNPS 240
C                                                                       DNPS 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DNPS 260
C           NONE                                                        DNPS 270
C                                                                       DNPS 280
C        METHOD                                                         DNPS 290
C           DEFINITION                                                  DNPS 300
C           Y=SUM(C(I)*T(I-1,X), SUMMED OVER I FROM 1 TO N).            DNPS 310
C           EVALUATION IS DONE BY MEANS OF BACKWARD RECURSION           DNPS 320
C           USING THE RECURRENCE EQUATION FOR CHEBYSHEV POLYNOMIALS     DNPS 330
C           T(N+1,X)=2*X*T(N,X)-T(N-1,X).                               DNPS 340
C                                                                       DNPS 350
C     ...................................................................DNPS 360
C                                                                       DNPS 370
      SUBROUTINE DCNPS(Y,X,C,N)                                         DNPS 380
C                                                                       DNPS 390
      DIMENSION C(1)                                                    DNPS 400
      DOUBLE PRECISION C,Y,X,H0,H1,H2,ARG                               DNPS 410
C                                                                       DNPS 420
C        TEST OF DIMENSION                                              DNPS 430
      IF(N)1,1,2                                                        DNPS 440
    1 RETURN                                                            DNPS 450
C                                                                       DNPS 460
    2 IF(N-2)3,4,4                                                      DNPS 470
    3 Y=C(1)                                                            DNPS 480
      RETURN                                                            DNPS 490
C                                                                       DNPS 500
C        INITIALIZATION                                                 DNPS 510
    4 ARG=X+X                                                           DNPS 520
      H1=0.D0                                                           DNPS 530
      H0=0.D0                                                           DNPS 540
C                                                                       DNPS 550
      DO 5 I=1,N                                                        DNPS 560
      K=N-I                                                             DNPS 570
      H2=H1                                                             DNPS 580
      H1=H0                                                             DNPS 590
    5 H0=ARG*H1-H2+C(K+1)                                               DNPS 600
      Y=0.5D0*(C(1)-H2+H0)                                              DNPS 610
      RETURN                                                            DNPS 620
      END                                                               DNPS 630
```

Subroutines TCNP and DTCNP

These subroutines transform a given series expansion in Chebyshev polynomials to a polynomial. The independent variable x of the given expansion is hereby substituted by the independent variable z of the resulting polynomial using the linear transformation $x = A * z + B$ or $z = (x - B)/A$.

This means:

$$\sum_{i=1}^{N} C_i * T_{i-1}(A * z + B)$$

$$= \sum_{i=1}^{N} POL_i * z^{i-1}$$

Calculation is bypassed in case of a nonpositive value of the dimension N.

The range (-1, +1) in x is transformed to the range (z_1, z_r) in z with $z_1 = -(1+B)/A$ and $z_r = (1-B)/A$, or vice versa, $A = 2/(z_r - z_1)$ and $B = -(z_r + z_1)/(z_r - z_1)$.

The Chebyshev polynomial $T_n(x)$ satisfies the recurrence equation

$$T_{n+1}(x) = 2x \cdot T_n(x) - T_{n-1}(x)$$

with starting values $T_0(x) = 1$, $T_1(x) = x$. The transformation is performed by means of a forward iteration scheme:

1. The coefficient vector of the Chebyshev polynomial $T_i(x)$ is calculated from the coefficient vectors of $T_{i-1}(x)$, $T_{i-2}(x)$ using the recurrence equation, for $i = 3, \ldots, N$.
2. The resulting polynomial coefficient vector is obtained by summation of c_i times coefficient vector of $T_{i-1}(x)$ over i, for $i = 1, \ldots, N$.

```
C     ..................................................................TCNP 490
C                                                                       TCNP 500
      SUBROUTINE TCNP(A,B,POL,N,C,WORK)                                 TCNP 510
C                                                                       TCNP 520
      DIMENSION POL(1),C(1),WORK(1)                                     TCNP 530
C                                                                       TCNP 540
C        TEST OF DIMENSION                                              TCNP 550
      IF(N-1)2,1,3                                                      TCNP 560
C                                                                       TCNP 570
C        DIMENSION LESS THAN 2                                          TCNP 580
    1 POL(1)=C(1)                                                       TCNP 590
    2 RETURN                                                            TCNP 600
C                                                                       TCNP 610
    3 POL(1)=C(1)+C(2)*B                                                TCNP 620
      POL(2)=C(2)*A                                                     TCNP 630
      IF(N-2)2,2,4                                                      TCNP 640
C                                                                       TCNP 650
C        INITIALIZATION                                                 TCNP 660
    4 WORK(1)=1.                                                        TCNP 670
      WORK(2)=B                                                         TCNP 680
      WORK(3)=0.                                                        TCNP 690
      WORK(4)=A                                                         TCNP 700
      XD=A+A                                                            TCNP 710
      XO=B+B                                                            TCNP 720
C                                                                       TCNP 730
C        CALCULATE COEFFICIENT VECTOR OF NEXT CHEBYSHEV POLYNOMIAL      TCNP 740
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              TCNP 750
      DO 6 J=3,N                                                        TCNP 760
      P=0.                                                              TCNP 770
C                                                                       TCNP 780
      DO 5 K=2,J                                                        TCNP 790
      H=P-WORK(2*K-3)+XO*WORK(2*K-2)                                    TCNP 800
      P=WORK(2*K-2)                                                     TCNP 810
      WORK(2*K-2)=H                                                     TCNP 820
      WORK(2*K-3)=P                                                     TCNP 830
      POL(K-1)=POL(K-1)+H*C(J)                                          TCNP 840
    5 P=XD*P                                                            TCNP 850
      WORK(2*J-1)=0.                                                    TCNP 860
      WORK(2*J)=P                                                       TCNP 870
    6 POL(J)=C(J)*P                                                     TCNP 880
      RETURN                                                            TCNP 890
      END                                                               TCNP 900
```

```
C                                                                       DTCN  10
C     ..................................................................DTCN  20
C                                                                       DTCN  30
C        SUBROUTINE DTCNP                                               DTCN  40
C                                                                       DTCN  50
C        PURPOSE                                                        DTCN  60
C           A SERIES EXPANSION IN CHEBYSHEV POLYNOMIALS WITH INDEPENDENTDTCN  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  DTCN  80
C           VARIABLE Z, WHERE X=A*Z+B.                                  DTCN  90
C                                                                       DTCN 100
C        USAGE                                                          DTCN 110
C           CALL DTCNP(A,B,POL,N,C,WORK)                                DTCN 120
C                                                                       DTCN 130
C        DESCRIPTION OF PARAMETERS                                      DTCN 140
C           A      - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONDTCN 150
C                    DOUBLE PRECISION VARIABLE                          DTCN 160
C           B      - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION      DTCN 170
C                    DOUBLE PRECISION VARIABLE                          DTCN 180
C           POL    - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE) DTCN 190
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DTCN 200
C                    DOUBLE PRECISION VECTOR                            DTCN 210
C           N      - DIMENSION OF COEFFICIENT VECTORS POL AND C        DTCN 220
C           C      - GIVEN COEFFICIENT VECTOR OF EXPANSION             DTCN 230
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DTCN 240
C                    POL AND C MAY BE IDENTICALLY LOCATED               DTCN 250
C                    DOUBLE PRECISION VECTOR                            DTCN 260
C           WORK   - WORKING STORAGE OF DIMENSION 2*N                  DTCN 270
C                    DOUBLE PRECISION ARRAY                             DTCN 280
C                                                                       DTCN 290
C        REMARKS                                                        DTCN 300
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    DTCN 310
C           WITH COEFFICIENT VECTOR POL.                                DTCN 320
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                DTCN 330
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSDTCN 340
C           THE RANGE (-1,+1) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE     DTCN 350
C           ZL=-(1+B)/A AND ZR=(1-B)/A.                                 DTCN 360
C           FOR GIVEN ZL, ZR WE HAVE A=2/(ZR-ZL) AND B=-(ZR+ZL)/(ZR-ZL) DTCN 370
C                                                                       DTCN 380
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTCN 390
C           NONE                                                        DTCN 400
C                                                                       DTCN 410
C        METHOD                                                         DTCN 420
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      DTCN 430
C           FOR CHEBYSHEV POLYNOMIALS T(N,X)                            DTCN 440
C           T(N+1,X)=2*X*T(N,X)-T(N-1,X),                               DTCN 450
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              DTCN 460
C           THE SECOND IS THE ARGUMENT.                                 DTCN 470
C           STARTING VALUES ARE T(0,X)=1, T(1,X)=X.                     DTCN 480
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        DTCN 490
C           X = A*Z+B TOGETHER WITH                                     DTCN 500
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             DTCN 510
C           =SUM(C(I)*T(I-1,X), SUMMED OVER I FROM 1 TO N).             DTCN 520
C                                                                       DTCN 530
C     ..................................................................DTCN 540
C                                                                       DTCN 550
      SUBROUTINE DTCNP(A,B,POL,N,C,WORK)                                DTCN 560
C                                                                       DTCN 570
      DIMENSION POL(1),C(1),WORK(1)                                     DTCN 580
      DOUBLE PRECISION A,B,POL,C,WORK,H,P,XD,XO                         DTCN 590
C                                                                       DTCN 600
C        TEST OF DIMENSION                                              DTCN 610
      IF(N-1)2,1,3                                                      DTCN 620
C                                                                       DTCN 630
C        DIMENSION LESS THAN 2                                          DTCN 640
    1 POL(1)=C(1)                                                       DTCN 650
    2 RETURN                                                            DTCN 660
C                                                                       DTCN 670
    3 POL(1)=C(1)+C(2)*B                                                DTCN 680
      POL(2)=C(2)*A                                                     DTCN 690
      IF(N-2)2,2,4                                                      DTCN 700
C                                                                       DTCN 710
C        INITIALIZATION                                                 DTCN 720
    4 WORK(1)=1.D0                                                      DTCN 730
      WORK(2)=B                                                         DTCN 740
      WORK(3)=0.D0                                                      DTCN 750
      WORK(4)=A                                                         DTCN 760
      XD=A+A                                                            DTCN 770
      XO=B+B                                                            DTCN 780
C                                                                       DTCN 790
```

```
C        CALCULATE COEFFICIENT VECTOR OF NEXT CHEBYSHEV POLYNOMIAL      DTCN 800
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              DTCN 810
      DO 6 J=3,N                                                        DTCN 820
      P=0.D0                                                            DTCN 830
C                                                                       DTCN 840
      DO 5 K=2,J                                                        DTCN 850
      H=P-WORK(2*K-3)+X0*WORK(2*K-2)                                    DTCN 860
      P=WORK(2*K-2)                                                     DTCN 870
      WORK(2*K-2)=H                                                     DTCN 880
      WORK(2*K-3)=P                                                     DTCN 890
      POL(K-1)=POL(K-1)+H*C(J)                                          DTCN 900
    5 P=X0*P                                                            DTCN 910
      WORK(2*J-1)=0.D0                                                  DTCN 920
      WORK(2*J)=P                                                       DTCN 930
    6 POL(J)=C(J)*P                                                     DTCN 940
      RETURN                                                            DTCN 950
      END                                                               DTCN 960
```

Subroutines CSP and DCSP

These subroutines compute the values of the shifted Chebyshev polynomials for a given argument x and orders zero up to N. The shifted Chebyshev polynomial satisfies the recurrence equation.

$$T^S_{n+1}(x) = 2\,(2\,x-1)\,T^S_n(x) - T^S_{n-1}(x)$$

with starting values $T^S_0(x) = 1$, $T^S_1(x) = 2 \cdot x - 1$. From this equation, it follows immediately that roundoff errors grow at worst linearly under the tacit assumption that the argument values are between zero and one.

If e_{n+r} is the error in $T^S_{n+r}(x)$ due to a single rounding error e in $T^S_n(x)$:

$$e_{n+r+1} = 2\,(2\,x-1)\,e_{n+r} - e_{n+r-1}$$

with initial conditions $e_n = e$, $e_{n-1} = 0$.

The solution of this difference equation has its maximum for x = 1 or x = 0.

$$e_{n-1} = 0,\ e_n = e,\ \left|e_{n+1}\right| = 2e,\ \ldots,\ \left|e_{n+r}\right| = (1 + r)\,e.$$

The order is assumed to be zero for negative values of N.

```
C                                                                       CSP   10
C     ..................................................................CSP   20
C                                                                       CSP   30
C        SUBROUTINE CSP                                                 CSP   40
C                                                                       CSP   50
C        PURPOSE                                                        CSP   60
C           COMPUTE THE VALUES OF THE SHIFTED CHEBYSHEV POLYNOMIALS     CSP   70
C           TS(N,X) FOR ARGUMENT X AND ORDERS 0 UP TO N.                CSP   80
C                                                                       CSP   90
C        USAGE                                                          CSP  100
C           CALL CSP(Y,X,N)                                             CSP  110
C                                                                       CSP  120
C        DESCRIPTION OF PARAMETERS                                      CSP  130
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESCSP  140
C                    OF SHIFTED CHEBYSHEV POLYNOMIALS OF ORDER 0 UP TO N CSP  150
C                    FOR GIVEN ARGUMENT X.                              CSP  160
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER          CSP  170
C           X      - ARGUMENT OF SHIFTED CHEBYSHEV POLYNOMIAL          CSP  180
C           N      - ORDER OF SHIFTED CHEBYSHEV POLYNOMIAL             CSP  190
C                                                                       CSP  200
C        REMARKS                                                        CSP  210
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     CSP  220
C                                                                       CSP  230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CSP  240
C           NONE                                                        CSP  250
C                                                                       CSP  260
C        METHOD                                                         CSP  270
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          CSP  280
C           SHIFTED CHEBYSHEV POLYNOMIALS TS(N,X)                       CSP  290
C           TS(N+1,X)=(4*X-2)*TS(N,X)-TS(N-1,X),                        CSP  300
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              CSP  310
C           THE SECOND IS THE ARGUMENT.                                 CSP  320
C           STARTING VALUES ARE TS(0,X)=1, TS(1,X)=2*X-1.               CSP  330
C                                                                       CSP  340
C     ..................................................................CSP  350
C                                                                       CSP  360
      SUBROUTINE CSP(Y,X,N)                                             CSP  370
C                                                                       CSP  380
      DIMENSION Y(1)                                                    CSP  390
C                                                                       CSP  400
C        TEST OF ORDER                                                  CSP  410
      Y(1)=1.                                                           CSP  420
      IF(N)1,1,2                                                        CSP  430
    1 RETURN                                                            CSP  440
C                                                                       CSP  450
    2 Y(2)=X+X-1.                                                       CSP  460
      IF(N-1)1,1,3                                                      CSP  470
C                                                                       CSP  480
C        INITIALIZATION                                                 CSP  490
    3 F=Y(2)+Y(2)                                                       CSP  500
C                                                                       CSP  510
      DO 4 I=2,N                                                        CSP  520
    4 Y(I+1)=F*Y(I)-Y(I-1)                                              CSP  530
      RETURN                                                            CSP  540
      END                                                               CSP  550
```

```
C                                                                       DSP  10
C     ..................................................................DSP  20
C                                                                       DSP  30
C        SUBROUTINE DCSP                                                DSP  40
C                                                                       DSP  50
C        PURPOSE                                                        DSP  60
C           COMPUTE THE VALUES OF THE SHIFTED CHEBYSHEV POLYNOMIALS     DSP  70
C           TS(N,X) FOR ARGUMENT X AND ORDERS 0 UP TO N.                DSP  80
C                                                                       DSP  90
C        USAGE                                                          DSP 100
C           CALL DCSP(Y,X,N)                                            DSP 110
C                                                                       DSP 120
C        DESCRIPTION OF PARAMETERS                                      DSP 130
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESDSP 140
C                    OF SHIFTED CHEBYSHEV POLYNOMIALS OF ORDER 0 UP TO N DSP 150
C                    FOR GIVEN ARGUMENT X.                              DSP 160
C                    DOUBLE PRECISION VECTOR.                           DSP 170
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER          DSP 180
C           X      - ARGUMENT OF SHIFTED CHEBYSHEV POLYNOMIAL          DSP 190
C                    DOUBLE PRECISION VARIABLE.                         DSP 200
C           N      - ORDER OF SHIFTED CHEBYSHEV POLYNOMIAL             DSP 210
C                                                                       DSP 220
C        REMARKS                                                        DSP 230
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     DSP 240
C                                                                       DSP 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DSP 260
C           NONE                                                        DSP 270
C                                                                       DSP 280
C        METHOD                                                         DSP 290
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          DSP 300
C           SHIFTED CHEBYSHEV POLYNOMIALS TS(N,X)                       DSP 310
C           TS(N+1,X)=(4*X-2)*TS(N,X)-TS(N-1,X),                        DSP 320
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              DSP 330
C           THE SECOND IS THE ARGUMENT.                                 DSP 340
C           STARTING VALUES ARE TS(0,X)=1, TS(1,X)=2*X-1.               DSP 350
C                                                                       DSP 360
C     ..................................................................DSP 370
C                                                                       DSP 380
      SUBROUTINE DCSP(Y,X,N)                                            DSP 390
C                                                                       DSP 400
      DIMENSION Y(1)                                                    DSP 410
      DOUBLE PRECISION Y,X,F                                            DSP 420
C                                                                       DSP 430
C        TEST OF ORDER                                                  DSP 440
      Y(1)=1.D0                                                         DSP 450
      IF(N)1,1,2                                                        DSP 460
    1 RETURN                                                            DSP 470
C                                                                       DSP 480
    2 Y(2)=X+X-1.D0                                                     DSP 490
      IF(N-1)1,1,3                                                      DSP 500
C                                                                       DSP 510
C        INITIALIZATION                                                 DSP 520
    3 F=Y(2)+Y(2)                                                       DSP 530
C                                                                       DSP 540
      DO 4 I=2,N                                                        DSP 550
    4 Y(I+1)=F*Y(I)-Y(I-1)                                              DSP 560
      RETURN                                                            DSP 570
      END                                                               DSP 580
```

Subroutines CSPS and DCSPS

These subroutines compute the value of a series expansion in shifted Chebyshev polynomials. The shifted Chebyshev polynomial $T_n^s(x)$ satisfies the recurrence equation

$$T_{n+1}^s(x) = 2\,(2\,x-1)\,T_n^s(x) - T_{n-1}^s(x)$$

with starting values $T_0^s(x) = 1$, $T_1^s(x) = 2x - 1$.

An n-term expansion in shifted Chebyshev polynomials with coefficient vector $C = (c_1, \ldots, c_n)$ is evaluated by means of a backward iteration scheme:

$$\mathrm{Arg} = 4\,x - 2,\ H_1 = 0,\ H_0 = 0$$

$$\left.\begin{aligned} H_2 &= H_1 \\ H_1 &= H_0 \\ H_0 &= \mathrm{Arg}\cdot H_1 - H_2 + c_{n-i+1} \\ Y &= (c_1 - H_2 + H_0)/2. \end{aligned}\right\}\ \text{for } i = 1, \ldots, n$$

This gives the result $Y(x) = \sum_{i=1}^{n} c_i\, T_{i-1}^s(x)$.

Calculation is bypassed in case of a nonpositive value of the dimension n.

```
C                                                                       CSPS  10
C     ..................................................................CSPS  20
C                                                                       CSPS  30
C        SUBROUTINE CSPS                                                CSPS  40
C                                                                       CSPS  50
C        PURPOSE                                                        CSPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN SHIFTED        CSPS  70
C           CHEBYSHEV POLYNOMIALS WITH COEFFICIENT VECTOR C             CSPS  80
C           FOR ARGUMENT VALUE X.                                       CSPS  90
C                                                                       CSPS 100
C        USAGE                                                          CSPS 110
C           CALL CSPS(Y,X,C,N)                                          CSPS 120
C                                                                       CSPS 130
C        DESCRIPTION OF PARAMETERS                                      CSPS 140
C           Y      - RESULT VALUE                                      CSPS 150
C           X      - ARGUMENT VALUE                                    CSPS 160
C           C      - COEFFICIENT VECTOR OF GIVEN EXPANSION             CSPS 170
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH         CSPS 180
C           N      - DIMENSION OF COEFFICIENT VECTOR C                 CSPS 190
C                                                                       CSPS 200
C        REMARKS                                                        CSPS 210
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 CSPS 220
C                                                                       CSPS 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  CSPS 240
C           NONE                                                        CSPS 250
C                                                                       CSPS 260
C        METHOD                                                         CSPS 270
C           DEFINITION                                                  CSPS 280
C           Y=SUM(C(I)*TS(I-1,X), SUMMED OVER I FROM 1 TO N).           CSPS 290
C           EVALUATION IS DONE BY MEANS OF BACKWARD RECURSION           CSPS 300
C           USING THE RECURRENCE EQUATION FOR SHIFTED                   CSPS 310
C           CHEBYSHEV POLYNOMIALS                                       CSPS 320
C           TS(N+1,X)=(4*X-2)*TS(N,X)-TS(N-1,X).                        CSPS 330
C                                                                       CSPS 340
C     ..................................................................CSPS 350
C                                                                       CSPS 360
      SUBROUTINE CSPS(Y,X,C,N)                                          CSPS 370
C                                                                       CSPS 380
      DIMENSION C(1)                                                    CSPS 390
C                                                                       CSPS 400
C        TEST OF DIMENSION                                              CSPS 410
      IF(N)1,1,2                                                        CSPS 420
    1 RETURN                                                            CSPS 430
C                                                                       CSPS 440
    2 IF(N-2)3,4,4                                                      CSPS 450
    3 Y=C(1)                                                            CSPS 460
      RETURN                                                            CSPS 470
C                                                                       CSPS 480
C        INITIALIZATION                                                 CSPS 490
    4 ARG=X+X-1.                                                        CSPS 500
      ARG=ARG+ARG                                                       CSPS 510
      H1=0.                                                             CSPS 520
      H0=0.                                                             CSPS 530
C                                                                       CSPS 540
      DO 5 I=1,N                                                        CSPS 550
      K=N-I                                                             CSPS 560
      H2=H1                                                             CSPS 570
      H1=H0                                                             CSPS 580
    5 H0=ARG*H1-H2+C(K+1)                                               CSPS 590
      Y=0.5*(C(1)-H2+H0)                                                CSPS 600
      RETURN                                                            CSPS 610
      END                                                               CSPS 620
```

```
C                                                                       DSPS  10
C     ...................................................................DSPS  20
C                                                                       DSPS  30
C        SUBROUTINE DCSPS                                               DSPS  40
C                                                                       DSPS  50
C        PURPOSE                                                        DSPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN SHIFTED        DSPS  70
C           CHEBYSHEV POLYNOMIALS WITH COEFFICIENT VECTOR C             DSPS  80
C           FOR ARGUMENT VALUE X.                                       DSPS  90
C                                                                       DSPS 100
C        USAGE                                                          DSPS 110
C           CALL DCSPS(Y,X,C,N)                                         DSPS 120
C                                                                       DSPS 130
C        DESCRIPTION OF PARAMETERS                                      DSPS 140
C           Y      - RESULT VALUE                                       DSPS 150
C                    DOUBLE PRECISION VARIABLE                          DSPS 160
C           X      - ARGUMENT VALUE                                     DSPS 170
C                    DOUBLE PRECISION VARIABLE                          DSPS 180
C           C      - COEFFICIENT VECTOR OF GIVEN EXPANSION              DSPS 190
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DSPS 200
C                    DOUBLE PRECISION VECTOR                            DSPS 210
C           N      - DIMENSION OF COEFFICIENT VECTOR C                  DSPS 220
C                                                                       DSPS 230
C        REMARKS                                                        DSPS 240
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 DSPS 250
C                                                                       DSPS 260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DSPS 270
C           NONE                                                        DSPS 280
C                                                                       DSPS 290
C        METHOD                                                         DSPS 300
C           DEFINITION                                                  DSPS 310
C           Y=SUM(C(I)*TS(I-1,X), SUMMED OVER I FROM 1 TO N).           DSPS 320
C           EVALUATION IS DONE BY MEANS OF BACKWARD RECURSION           DSPS 330
C           USING THE RECURRENCE EQUATION FOR SHIFTED                   DSPS 340
C           CHEBYSHEV POLYNOMIALS                                       DSPS 350
C           TS(N+1,X)=(4*X-2)*TS(N,X)-TS(N-1,X).                        DSPS 360
C                                                                       DSPS 370
C     ...................................................................DSPS 380
C                                                                       DSPS 390
      SUBROUTINE DCSPS(Y,X,C,N)                                         DSPS 400
C                                                                       DSPS 410
      DIMENSION C(1)                                                    DSPS 420
      DOUBLE PRECISION C,Y,X,HO,H1,H2,ARG                               DSPS 430
C                                                                       DSPS 440
C        TEST OF DIMENSION                                              DSPS 450
      IF(N)1,1,2                                                        DSPS 460
    1 RETURN                                                            DSPS 470
C                                                                       DSPS 480
    2 IF(N-2)3,4,4                                                      DSPS 490
    3 Y=C(1)                                                            DSPS 500
      RETURN                                                            DSPS 510
C                                                                       DSPS 520
C        INITIALIZATION                                                 DSPS 530
    4 ARG=X+X-1.D0                                                      DSPS 540
      ARG=ARG+ARG                                                       DSPS 550
      H1=0.D0                                                           DSPS 560
      HO=0.D0                                                           DSPS 570
      DO 5 I=1,N                                                        DSPS 580
      K=N-I                                                             DSPS 590
      H2=H1                                                             DSPS 600
      H1=HO                                                             DSPS 610
    5 HO=ARG*H1-H2+C(K+1)                                               DSPS 620
      Y=0.5D0*(C(1)-H2+HO)                                              DSPS 630
      RETURN                                                            DSPS 640
      END                                                               DSPS 650
```

Subroutines TCSP and DTCSP

These subroutines transform a given series expansion in shifted Chebyshev polynomials to a polynomial. The independent variable x of the given expansion is thereby substituted by the independent variable z of the resulting polynomial using the linear transformation $x = A * z + B$ or $z = (x - B)/A$.

This means:
$$\sum_{i=1}^{N} C_i * T^S_{i-1}(A * z + B)$$
$$= \sum_{i=1}^{N} POL_i * z^{i-1}$$

Calculation is bypassed in case of a nonpositive value of the dimension N.

The range (0, +1) in x is transformed to the range (z_l, z_r) in z with $z_l = -B/A$ and $z_r = (1-B)/A$ or vice versa, $A = 1/(z_r - z_l)$ and $B = -z_l/(z_r - z_l)$.

The shifted Chebyshev polynomial $T^S_n(x)$ satisfies the recurrence equation

$$T^S_{n+1}(x) = 4\,x\,T^S_n(x) - 2\,T^S_n(x) - T^S_{n-1}(x)$$

with starting values $T^S_0(x) = 1$, $T^S_1(x) = 2\,x - 1$. The transformation is performed by means of a forward iteration scheme:

1. The coefficient vector of the shifted Chebyshev polynomial $T^S_i(x)$ is calculated from the coefficient vectors of $T^S_{i-1}(x)$, $T^S_{i-2}(x)$ using the recurrence equation, for $i = 3, \ldots, N$.
2. The resulting polynomial coefficient vector is obtained by summation of c_i times $T^S_{i-1}(x)$ over i, for $i = 1, \ldots, N$.

```
C                                                                       TCSP  10
C     ...................................................................TCSP  20
C                                                                       TCSP  30
C        SUBROUTINE TCSP                                                TCSP  40
C                                                                       TCSP  50
C        PURPOSE                                                        TCSP  60
C           A SERIES EXPANSION IN SHIFTED CHEBYSHEV POLYNOMIALS WITH    TCSP  70
C           INDEPENDENT VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH  TCSP  80
C           INDEPENDENT VARIABLE Z, WHERE X=A*Z+B.                      TCSP  90
C                                                                       TCSP 100
C        USAGE                                                          TCSP 110
C           CALL TCSP(A,B,POL,N,C,WORK)                                 TCSP 120
C                                                                       TCSP 130
C        DESCRIPTION OF PARAMETERS                                      TCSP 140
C           A      - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONTCSP 150
C           B      - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION      TCSP 160
C           POL    - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE) TCSP 170
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          TCSP 180
C           N      - DIMENSION OF COEFFICIENT VECTORS POL AND C         TCSP 190
C           C      - GIVEN COEFFICIENT VECTOR OF EXPANSION              TCSP 200
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          TCSP 210
C                    POL AND C MAY BE IDENTICALLY LOCATED               TCSP 220
C           WORK   - WORKING STORAGE OF DIMENSION 2*N                   TCSP 230
C                                                                       TCSP 240
C        REMARKS                                                        TCSP 250
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    TCSP 260
C           WITH COEFFICIENT VECTOR POL.                                TCSP 270
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                TCSP 280
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSTCSP 290
C           THE RANGE (0,1) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE       TCSP 300
C           ZL=-B/A AND ZR=(1-B)/A.                                     TCSP 310
C           FOR GIVEN ZL, ZR WE HAVE A=1/(ZR-ZL) AND B=-ZL/(ZR-ZL).     TCSP 320
C                                                                       TCSP 330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TCSP 340
C           NONE                                                        TCSP 350
C                                                                       TCSP 360
C        METHOD                                                         TCSP 370
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION FOR  TCSP 380
C           SHIFTED CHEBYSHEV POLYNOMIALS TS(N,X)                       TCSP 390
C           TS(N+1,X)=(4*X-2)*TS(N,X)-TS(N-1,X),                        TCSP 400
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              TCSP 410
C           THE SECOND IS THE ARGUMENT.                                 TCSP 420
```

```
      STARTING VALUES ARE TS(0,X)=1, TS(1,X)=2*X-1.                     TCSP 430
      THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF              TCSP 440
      X=A*Z+B TOGETHER WITH                                             TCSP 450
      SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)                   TCSP 460
      =SUM(C(I)*TS(I-1,X), SUMMED OVER I FROM 1 TO N).                  TCSP 470
                                                                        TCSP 480
 ......................................................................TCSP 490
                                                                        TCSP 500
 SUBROUTINE TCSP(A,B,POL,N,C,WORK)                                      TCSP 510
                                                                        TCSP 520
 DIMENSION POL(1),C(1),WORK(1)                                          TCSP 530
                                                                        TCSP 540
    TEST OF DIMENSION                                                   TCSP 550
 IF(N-1)2,1,3                                                           TCSP 560
                                                                        TCSP 570
    DIMENSION LESS THAN 2                                               TCSP 580
1 POL(1)=C(1)                                                           TCSP 590
2 RETURN                                                                TCSP 600
                                                                        TCSP 610
3 XD=A+A                                                                TCSP 620
  X0=B+B-1.                                                             TCSP 630
  POL(1)=C(1)+C(2)*X0                                                   TCSP 640
  POL(2)=C(2)*XD                                                        TCSP 650
  IF(N-2)2,2,4                                                          TCSP 660
                                                                        TCSP 670
    INITIALIZATION                                                      TCSP 680
4 WORK(1)=1.                                                            TCSP 690
  WORK(2)=X0                                                            TCSP 700
  WORK(3)=0.                                                            TCSP 710
  WORK(4)=XD                                                            TCSP 720
  XD=XD+XD                                                              TCSP 730
  X0=X0+X0                                                              TCSP 740
                                                                        TCSP 750
    CALCULATE COEFFICIENT VECTOR OF NEXT SHIFTED CHEBYSHEV              TCSP 760
    POLYNOMIAL AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL        TCSP 770
  DO 6 J=3,N                                                            TCSP 780
  P=0.                                                                  TCSP 790
                                                                        TCSP 800
  DO 5 K=2,J                                                            TCSP 810
  H=P-WORK(2*K-3)+X0*WORK(2*K-2)                                        TCSP 820
  P=WORK(2*K-2)                                                         TCSP 830
  WORK(2*K-2)=H                                                         TCSP 840
  WORK(2*K-3)=P                                                         TCSP 850
  POL(K-1)=POL(K-1)+H*C(J)                                              TCSP 860
5 P=XD*P                                                                TCSP 870
  WORK(2*J-1)=0.                                                        TCSP 880
  WORK(2*J)=P                                                           TCSP 890
6 POL(J)=C(J)*P                                                         TCSP 900
  RETURN                                                                TCSP 910
  END                                                                   TCSP 920
```

```
C                                                                       DTCS  10
C     ..................................................................DTCS  20
C                                                                       DTCS  30
C        SUBROUTINE DTCSP                                               DTCS  40
C                                                                       DTCS  50
C        PURPOSE                                                        DTCS  60
C           A SERIES EXPANSION IN SHIFTED CHEBYSHEV POLYNOMIALS WITH    DTCS  70
C           INDEPENDENT VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH  DTCS  80
C           INDEPENDENT VARIABLE Z, WHERE X=A*Z+B.                      DTCS  90
C                                                                       DTCS 100
C        USAGE                                                          DTCS 110
C           CALL DTCSP(A,B,POL,N,C,WORK)                                DTCS 120
C                                                                       DTCS 130
C        DESCRIPTION OF PARAMETERS                                      DTCS 140
C           A     - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONDTCS 150
C                   DOUBLE PRECISION VARIABLE                           DTCS 160
C           B     - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION        DTCS 170
C                   DOUBLE PRECISION VARIABLE                           DTCS 180
C           POL   - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE)  DTCS 190
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           DTCS 200
C                   DOUBLE PRECISION VECTOR                             DTCS 210
C           N     - DIMENSION OF COEFFICIENT VECTORS POL AND C          DTCS 220
C           C     - GIVEN COEFFICIENT VECTOR OF EXPANSION               DTCS 230
C                   POL AND C MAY BE IDENTICALLY LOCATED                DTCS 240
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           DTCS 250
C                   DOUBLE PRECISION VECTOR                             DTCS 260
C           WORK  - WORKING STORAGE OF DIMENSION 2*N                    DTCS 270
C                   DOUBLE PRECISION ARRAY                              DTCS 280
C                                                                       DTCS 290
C        REMARKS                                                        DTCS 300
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    DTCS 310
C           WITH COEFFICIENT VECTOR POL.                                DTCS 320
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                DTCS 330
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSDTCS 340
C           THE RANGE (0,1) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE       DTCS 350
C           ZL=-B/A AND ZR=(1-B)/A.                                     DTCS 360
C           FOR GIVEN ZL, ZR WE HAVE A=1/(ZR-ZL) AND B=-ZL/(ZR-ZL).     DTCS 370
C                                                                       DTCS 380
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTCS 390
C           NONE                                                        DTCS 400
C                                                                       DTCS 410
C        METHOD                                                         DTCS 420
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION FOR  DTCS 430
C           SHIFTED CHEBYSHEV POLYNOMIALS TS(N,X)                       DTCS 440
C           TS(N+1,X)=(4*X-2)*TS(N,X)-TS(N-1,X),                        DTCS 450
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              DTCS 460
C           THE SECOND IS THE ARGUMENT.                                 DTCS 470
C           STARTING VALUES ARE TS(0,X)=1, TS(1,X)=2*X-1.               DTCS 480
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        DTCS 490
C           X=A*Z+B TOGETHER WITH                                       DTCS 500
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             DTCS 510
C           =SUM(C(I)*TS(I-1,X), SUMMED OVER I FROM 1 TO N).            DTCS 520
C                                                                       DTCS 530
C     ..................................................................DTCS 540
C                                                                       DTCS 550
      SUBROUTINE DTCSP(A,B,POL,N,C,WORK)                                DTCS 560
C                                                                       DTCS 570
      DIMENSION POL(1),C(1),WORK(1)                                     DTCS 580
      DOUBLE PRECISION A,B,POL,C,WORK,H,P,XD,X0                         DTCS 590
C                                                                       DTCS 600
C        TEST OF DIMENSION                                              DTCS 610
      IF(N-1)2,1,3                                                      DTCS 620
C                                                                       DTCS 630
C        DIMENSION LESS THAN 2                                          DTCS 640
    1 POL(1)=C(1)                                                       DTCS 650
    2 RETURN                                                            DTCS 660
C                                                                       DTCS 670
    3 XD=A+A                                                            DTCS 680
      X0=B+B-1.D0                                                       DTCS 690
      POL(1)=C(1)+C(2)*X0                                               DTCS 700
      POL(2)=C(2)*XD                                                    DTCS 710
      IF(N-2)2,2,4                                                      DTCS 720
C                                                                       DTCS 730
C        INITIALIZATION                                                 DTCS 740
    4 WORK(1)=1.D0                                                      DTCS 750
      WORK(2)=X0                                                        DTCS 760
      WORK(3)=0.D0                                                      DTCS 770
      WORK(4)=XD                                                        DTCS 780
      XD=XD+XD                                                          DTCS 790
      X0=X0+X0                                                          DTCS 800
C                                                                       DTCS 810
C        CALCULATE COEFFICIENT VECTOR OF NEXT SHIFTED CHEBYSHEV         DTCS 820
C        POLYNOMIAL AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL   DTCS 830
      DO 6 J=3,N                                                        DTCS 840
      P=0.D0                                                            DTCS 850
C                                                                       DTCS 860
      DO 5 K=2,J                                                        DTCS 870
      H=P-WORK(2*K-3)+X0*WORK(2*K-2)                                    DTCS 880
      P=WORK(2*K-2)                                                     DTCS 890
      WORK(2*K-2)=H                                                     DTCS 900
      WORK(2*K-3)=P                                                     DTCS 910
      POL(K-1)=POL(K-1)+H*C(J)                                          DTCS 920
    5 P=XD*P                                                            DTCS 930
      WORK(2*J-1)=0.D0                                                  DTCS 940
      WORK(2*J)=P                                                       DTCS 950
    6 POL(J)=C(J)*P                                                     DTCS 960
      RETURN                                                            DTCS 970
      END                                                               DTCS 980
```

Subroutines HEP and DHEP

These subroutines compute the values of the Hermite polynomials for a given argument x and orders zero up to N. The Hermite polynomial $H_n(x)$ satisfies the recurrence equation

$$H_{n+1}(x) = 2\,(x \cdot H_n(x) - n \cdot H_{n-1}(x))$$

with starting values $H_0(x) = 1$, $H_1(x) = 2\,x$.

The order is assumed to be zero for negative values of N.

```
C                                                                       HEP   10
C     ...................................................................HEP   20
C                                                                       HEP   30
C        SUBROUTINE HEP                                                 HEP   40
C                                                                       HEP   50
C        PURPOSE                                                        HEP   60
C           COMPUTE THE VALUES OF THE HERMITE POLYNOMIALS H(N,X)        HEP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  HEP   80
C                                                                       HEP   90
C        USAGE                                                          HEP  100
C           CALL HEP(Y,X,N)                                             HEP  110
C                                                                       HEP  120
C        DESCRIPTION OF PARAMETERS                                      HEP  130
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESHEP  140
C                    OF HERMITE POLYNOMIALS OF ORDER 0 UP TO N          HEP  150
C                    FOR GIVEN ARGUMENT X.                              HEP  160
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER          HEP  170
C           X      - ARGUMENT OF HERMITE POLYNOMIAL                    HEP  180
C           N      - ORDER OF HERMITE POLYNOMIAL                       HEP  190
C                                                                       HEP  200
C        REMARKS                                                        HEP  210
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     HEP  220
C                                                                       HEP  230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  HEP  240
C           NONE                                                        HEP  250
C                                                                       HEP  260
C        METHOD                                                         HEP  270
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          HEP  280
C           HERMITE POLYNOMIALS H(N,X)                                  HEP  290
C           H(N+1,X)=2*(X*H(N,X)-N*H(N-1,X))                            HEP  300
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              HEP  310
C           THE SECOND IS THE ARGUMENT.                                 HEP  320
C           STARTING VALUES ARE H(0,X)=1, H(1,X)=2*X.                   HEP  330
C                                                                       HEP  340
C     ...................................................................HEP  350
C                                                                       HEP  360
      SUBROUTINE HEP(Y,X,N)                                             HEP  370
C                                                                       HEP  380
      DIMENSION Y(1)                                                    HEP  390
C                                                                       HEP  400
C        TEST OF ORDER                                                  HEP  410
      Y(1)=1.                                                           HEP  420
      IF(N)1,1,2                                                        HEP  430
    1 RETURN                                                            HEP  440
C                                                                       HEP  450
    2 Y(2)=X+X                                                          HEP  460
      IF(N-1)1,1,3                                                      HEP  470
C                                                                       HEP  480
    3 DO 4 I=2,N                                                        HEP  490
      F=X*Y(I)-FLOAT(I-1)*Y(I-1)                                        HEP  500
    4 Y(I+1)=F+F                                                        HEP  510
      RETURN                                                            HEP  520
      END                                                               HEP  530
```

```
C                                                                       DHP   10
C     ...................................................................DHP   20
C                                                                       DHP   30
C        SUBROUTINE DHEP                                                DHP   40
C                                                                       DHP   50
C        PURPOSE                                                        DHP   60
C           COMPUTE THE VALUES OF THE HERMITE POLYNOMIALS H(N,X)        DHP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  DHP   80
C                                                                       DHP   90
C        USAGE                                                          DHP  100
C           CALL DHEP(Y,X,N)                                            DHP  110
C                                                                       DHP  120
C        DESCRIPTION OF PARAMETERS                                      DHP  130
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESDHP  140
C                    OF HERMITE POLYNOMIALS OF ORDER 0 UP TO N          DHP  150
C                    FOR GIVEN ARGUMENT X.                              DHP  160
C                    DOUBLE PRECISION VECTOR.                           DHP  170
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER          DHP  180
C           X      - ARGUMENT OF HERMITE POLYNOMIAL                    DHP  190
C                    DOUBLE PRECISION VARIABLE.                         DHP  200
C           N      - ORDER OF HERMITE POLYNOMIAL                       DHP  210
C                                                                       DHP  220
C        REMARKS                                                        DHP  230
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     DHP  240
C                                                                       DHP  250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DHP  260
C           NONE                                                        DHP  270
C                                                                       DHP  280
C        METHOD                                                         DHP  290
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          DHP  300
C           HERMITE POLYNOMIALS H(N,X)                                  DHP  310
C           H(N+1,X)=2*(X*H(N,X)-N*H(N-1,X))                            DHP  320
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              DHP  330
C           THE SECOND IS THE ARGUMENT.                                 DHP  340
C           STARTING VALUES ARE H(0,X)=1, H(1,X)=2*X.                   DHP  350
C                                                                       DHP  360
C     ...................................................................DHP  370
C                                                                       DHP  380
      SUBROUTINE DHEP(Y,X,N)                                            DHP  390
C                                                                       DHP  400
      DIMENSION Y(1)                                                    DHP  410
      DOUBLE PRECISION Y,X,F                                            DHP  420
C                                                                       DHP  430
C        TEST OF ORDER                                                  DHP  440
      Y(1)=1.D0                                                         DHP  450
      IF(N)1,1,2                                                        DHP  460
    1 RETURN                                                            DHP  470
C                                                                       DHP  480
    2 Y(2)=X+X                                                          DHP  490
      IF(N-1)1,1,3                                                      DHP  500
C                                                                       DHP  510
    3 DO 4 I=2,N                                                        DHP  520
      F=X*Y(I)-DFLOAT(I-1)*Y(I-1)                                       DHP  530
    4 Y(I+1)=F+F                                                        DHP  540
      RETURN                                                            DHP  550
      END                                                               DHP  560
```

Subroutines HEPS and DHEPS

These subroutines compute the value of a series expansion in Hermite polynomials. The Hermite polynomial $H_n(x)$ satisfies the recurrence equation

$$H_{n+1}(x) = 2\,(x \cdot H_n(x) - n \cdot H_{n-1}(x))$$

with starting values $H_0(x) = 1$, $H_1(x) = 2\,x$.

An n-term expansion in Hermite polynomials with coefficient vector $C = (c_1, \ldots, c_n)$ is evaluated by means of a forward iteration scheme:

$$Y = c_1,\ H_0 = 1,\ H_1 = 2 \cdot x$$

$$\left.\begin{aligned} H_2 &= x \cdot H_1 - (i-1)\,H_0 \\ H_0 &= H_1 \\ H_1 &= H_2 + H_2 \\ Y &= Y + c_i \cdot H_0 \end{aligned}\right\} \quad \text{for } i = 2, \ldots, n$$

This gives the result $Y(x) = \sum_{i=1}^{n} c_i \cdot H_{i-1}(x)$.

Calculation is bypassed in case of a nonpositive value of the dimension n.

```
C                                                                       HEPS  10
C     ...................................................................HEPS  20
C                                                                       HEPS  30
C        SUBROUTINE HEPS                                                HEPS  40
C                                                                       HEPS  50
C        PURPOSE                                                        HEPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN HERMITE        HEPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. HEPS  80
C                                                                       HEPS  90
C        USAGE                                                          HEPS 100
C           CALL HEPS(Y,X,C,N)                                          HEPS 110
C                                                                       HEPS 120
C        DESCRIPTION OF PARAMETERS                                      HEPS 130
C           Y     - RESULT VALUE                                        HEPS 140
C           X     - ARGUMENT VALUE                                      HEPS 150
C           C     - COEFFICIENT VECTOR OF GIVEN EXPANSION               HEPS 160
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           HEPS 170
C           N     - DIMENSION OF COEFFICIENT VECTOR C                   HEPS 180
C                                                                       HEPS 190
C        REMARKS                                                        HEPS 200
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 HEPS 210
C                                                                       HEPS 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  HEPS 230
C           NONE                                                        HEPS 240
C                                                                       HEPS 250
C        METHOD                                                         HEPS 260
C           DEFINITION                                                  HEPS 270
C           Y=SUM(C(I)*H(I-1,X), SUMMED OVER I FROM 1 TO N).            HEPS 280
C           EVALUATION IS DONE BY MEANS OF UPWARD RECURSION             HEPS 290
C           USING THE RECURRENCE EQUATION FOR HERMITE POLYNOMIALS       HEPS 300
C           H(N+1,X)=2*(X*H(N,X)-N*H(N-1,X)).                           HEPS 310
C                                                                       HEPS 320
C     ...................................................................HEPS 330
C                                                                       HEPS 340
      SUBROUTINE HEPS(Y,X,C,N)                                          HEPS 350
C                                                                       HEPS 360
      DIMENSION C(1)                                                    HEPS 370
C                                                                       HEPS 380
C        TEST OF DIMENSION                                              HEPS 390
      IF(N)1,1,2                                                        HEPS 400
    1 RETURN                                                            HEPS 410
C                                                                       HEPS 420
    2 Y=C(1)                                                            HEPS 430
      IF(N-2)1,3,3                                                      HEPS 440
C                                                                       HEPS 450
C        INITIALIZATION                                                 HEPS 460
    3 H0=1.                                                             HEPS 470
      H1=X+X                                                            HEPS 480
C                                                                       HEPS 490
      DO 4 I=2,N                                                        HEPS 500
      H2=X*H1-FLOAT(I-1)*H0                                             HEPS 510
      H0=H1                                                             HEPS 520
      H1=H2+H2                                                          HEPS 530
    4 Y=Y+C(I)*H0                                                       HEPS 540
      RETURN                                                            HEPS 550
      END                                                               HEPS 560
```

```
C                                                                       DHPS  10
C     ...................................................................DHPS  20
C                                                                       DHPS  30
C        SUBROUTINE DHEPS                                               DHPS  40
C                                                                       DHPS  50
C        PURPOSE                                                        DHPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN HERMITE        DHPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. DHPS  80
C                                                                       DHPS  90
C        USAGE                                                          DHPS 100
C           CALL DHEPS(Y,X,C,N)                                         DHPS 110
C                                                                       DHPS 120
C        DESCRIPTION OF PARAMETERS                                      DHPS 130
C           Y     - RESULT VALUE                                        DHPS 140
C                   DOUBLE PRECISION VARIABLE                           DHPS 150
C           X     - ARGUMENT VALUE                                      DHPS 160
C                   DOUBLE PRECISION VARIABLE                           DHPS 170
C           C     - COEFFICIENT VECTOR OF GIVEN EXPANSION               DHPS 180
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           DHPS 190
C                   DOUBLE PRECISION VECTOR                             DHPS 200
C           N     - DIMENSION OF COEFFICIENT VECTOR C                   DHPS 210
C                                                                       DHPS 220
C        REMARKS                                                        DHPS 230
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 DHPS 240
C                                                                       DHPS 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DHPS 260
C           NONE                                                        DHPS 270
C                                                                       DHPS 280
C        METHOD                                                         DHPS 290
C           DEFINITION                                                  DHPS 300
C           Y=SUM(C(I)*H(I-1,X), SUMMED OVER I FROM 1 TO N).            DHPS 310
C           EVALUATION IS DONE BY MEANS OF UPWARD RECURSION             DHPS 320
C           USING THE RECURRENCE EQUATION FOR HERMITE POLYNOMIALS       DHPS 330
C           H(N+1,X)=2*(X*H(N,X)-N*H(N-1,X)).                           DHPS 340
C                                                                       DHPS 350
C     ...................................................................DHPS 360
C                                                                       DHPS 370
      SUBROUTINE DHEPS(Y,X,C,N)                                         DHPS 380
C                                                                       DHPS 390
      DIMENSION C(1)                                                    DHPS 400
      DOUBLE PRECISION C,Y,X,H0,H1,H2                                   DHPS 410
C                                                                       DHPS 420
C        TEST OF DIMENSION                                              DHPS 430
      IF(N)1,1,2                                                        DHPS 440
    1 RETURN                                                            DHPS 450
C                                                                       DHPS 460
    2 Y=C(1)                                                            DHPS 470
      IF(N-2)1,3,3                                                      DHPS 480
C                                                                       DHPS 490
C        INITIALIZATION                                                 DHPS 500
    3 H0=1.D0                                                           DHPS 510
      H1=X+X                                                            DHPS 520
C                                                                       DHPS 530
      DO 4 I=2,N                                                        DHPS 540
      H2=X*H1-DFLOAT(I-1)*H0                                            DHPS 550
      H0=H1                                                             DHPS 560
      H1=H2+H2                                                          DHPS 570
    4 Y=Y+C(I)*H0                                                       DHPS 580
      RETURN                                                            DHPS 590
      END                                                               DHPS 600
```

Subroutines THEP and DTHEP

These subroutines transform a given series expansion in Hermite polynomials to a polynomial. The independent variable x of the given expansion is thereby substituted by the independent variable z of the resulting polynomial using the linear transformation $x = A * z + B$ or $z = (x-B)/A$.

This means:
$$\sum_{i=1}^{N} C_i * H_{i-1} (A * z + B)$$
$$= \sum_{i=1}^{N} POL_i * z^{i-1}$$

Calculation is bypassed in case of a nonpositive value of the dimension N.

The range $(-c, +c)$ in x is transformed to the range (z_l, z_r) in z with $z_l = -(c + B)/A$ and $z_r = (c-B)/A$, or vice versa, $A = 2c/(z_r - z_l)$ and $B = -c(z_r + z_l)/(z_r - z_l)$.

The Hermite polynomial $H_n(x)$ satisfies the recurrence equation

$$H_{n+1}(x) = 2(x \cdot H_n(x) - n \cdot H_{n-1}(x))$$

with starting values $H_0(x) = 1$, $H_1(x) = 2x$.

The transformation is performed by means of a forward iteration scheme:

1. The coefficient vector of the Hermite polynomial $H_i(x)$ is calculated from the coefficient vectors of $H_{i-1}(x)$, $H_{i-2}(x)$ using the recurrence equation, for $i = 3, \ldots, N$.
2. The resulting polynomial coefficient vector is obtained by summation of c_i times coefficient vector of $H_{i-1}(x)$ over i, for $i = 1, \ldots, N$.

```
C                                                                       THEP  10
C     ...................................................................THEP  20
C                                                                       THEP  30
C        SUBROUTINE THEP                                                THEP  40
C                                                                       THEP  50
C        PURPOSE                                                        THEP  60
C           A SERIES EXPANSION IN HERMITE POLYNOMIALS WITH INDEPENDENT  THEP  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  THEP  80
C           VARIABLE Z, WHERE X=A*Z+B                                   THEP  90
C                                                                       THEP 100
C        USAGE                                                          THEP 110
C           CALL THEP(A,B,POL,N,C,WORK)                                 THEP 120
C                                                                       THEP 130
C        DESCRIPTION OF PARAMETERS                                      THEP 140
C           A     - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONTHEP 150
C           B     - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION        THEP 160
C           POL   - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE)  THEP 170
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           THEP 180
C           N     - DIMENSION OF COEFFICIENT VECTOR POL AND C           THEP 190
C           C     - COEFFICIENT VECTOR OF GIVEN EXPANSION               THEP 200
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           THEP 210
C                   POL AND C MAY BE IDENTICALLY LOCATED                THEP 220
C           WORK  - WORKING STORAGE OF DIMENSION 2*N                    THEP 230
C                                                                       THEP 240
C        REMARKS                                                        THEP 250
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    THEP 260
C           WITH COEFFICIENT VECTOR POL.                                THEP 270
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                THEP 280
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSTHEP 290
C           THE RANGE (-C,C) IN X TO THE RANGE (ZL,ZR) IN Z WHERE       THEP 300
C           ZL=-(C+B)/A AND ZR=(C-B)/A.                                 THEP 310
C           FOR GIVEN ZL, ZR AND C WE HAVE A=2C/(ZR-ZL) AND             THEP 320
C           B=-C(ZR+ZL)/(ZR-ZL)                                         THEP 330
C                                                                       THEP 340
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  THEP 350
C           NONE                                                        THEP 360
C                                                                       THEP 370
C        METHOD                                                         THEP 380
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      THEP 390
C           FOR HERMITE POLYNOMIALS H(N,X)                              THEP 400
C           H(N+1,X)=2*(X*H(N,X)-N*H(N-1,X)),                           THEP 410
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX               THEP 420
C           THE SECOND IS THE ARGUMENT.                                 THEP 430
C           STARTING VALUES ARE H(0,X)=1,H(1,X)=2*X.                    THEP 440
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        THEP 450
C           X=A*Z+B TOGETHER WITH                                       THEP 460
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             THEP 470
C           =SUM(C(I)*H(I-1,X), SUMMED OVER I FROM 1 TO N).             THEP 480
C                                                                       THEP 490
C     ...................................................................THEP 500
C                                                                       THEP 510
      SUBROUTINE THEP(A,B,POL,N,C,WORK)                                 THEP 520
C                                                                       THEP 530
      DIMENSION POL(1),C(1),WORK(1)                                     THEP 540
C                                                                       THEP 550
C        TEST OF DIMENSION                                              THEP 560
      IF(N-1)2,1,3                                                      THEP 570
C                                                                       THEP 580
C        DIMENSION LESS THAN 2                                          THEP 590
    1 POL(1)=C(1)                                                       THEP 600
    2 RETURN                                                            THEP 610
C                                                                       THEP 620
    3 XO=A+A                                                            THEP 630
      XO=B+B                                                            THEP 640
      POL(1)=C(1)+C(2)*XO                                               THEP 650
      POL(2)=C(2)*XO                                                    THEP 660
      IF(N-2)2,2,4                                                      THEP 670
C                                                                       THEP 680
C        INITIALIZATION                                                 THEP 690
    4 WORK(1)=1.                                                        THEP 700
      WORK(2)=XO                                                        THEP 710
      WORK(3)=0.                                                        THEP 720
      WORK(4)=XO                                                        THEP 730
      FI=2.                                                             THEP 740
C                                                                       THEP 750
C        CALCULATE COEFFICIENT VECTOR OF NEXT HERMITE POLYNOMIAL        THEP 760
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              THEP 770
      DO 6 J=3,N                                                        THEP 780
      P=0.                                                              THEP 790
C                                                                       THEP 800
      DO 5 K=2,J                                                        THEP 810
      H=P*XO+WORK(2*K-2)*XO-FI*WORK(2*K-3)                              THEP 820
      P=WORK(2*K-2)                                                     THEP 830
      WORK(2*K-2)=H                                                     THEP 840
      WORK(2*K-3)=P                                                     THEP 850
    5 POL(K-1)=POL(K-1)+H*C(J)                                          THEP 860
      WORK(2*J-1)=0.                                                    THEP 870
      WORK(2*J)=P*XO                                                    THEP 880
      FI=FI+2.                                                          THEP 890
    6 POL(J)=C(J)*WORK(2*J)                                             THEP 900
      RETURN                                                            THEP 910
      END                                                               THEP 920
```

```
C                                                                       DTHE  10
C     ..................................................................DTHE  20
C                                                                       DTHE  30
C        SUBROUTINE DTHEP                                               DTHE  40
C                                                                       DTHE  50
C        PURPOSE                                                        DTHE  60
C           A SERIES EXPANSION IN HERMITE POLYNOMIALS WITH INDEPENDENT  DTHE  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  DTHE  80
C           VARIABLE Z, WHERE X=A*Z+B                                   DTHE  90
C                                                                       DTHE 100
C        USAGE                                                          DTHE 110
C           CALL DTHEP(A,B,POL,N,C,WORK)                                DTHE 120
C                                                                       DTHE 130
C        DESCRIPTION OF PARAMETERS                                      DTHE 140
C           A     - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONDTHE 150
C                   DOUBLE PRECISION VARIABLE                           DTHE 160
C           B     - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION        DTHE 170
C                   DOUBLE PRECISION VARIABLE                           DTHE 180
C           POL   - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE)  DTHE 190
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           DTHE 200
C                   DOUBLE PRECISION VECTOR                             DTHE 210
C           N     - DIMENSION OF COEFFICIENT VECTOR POL AND C           DTHE 220
C           C     - COEFFICIENT VECTOR OF GIVEN EXPANSION               DTHE 230
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           DTHE 240
C                   POL AND C MAY BE IDENTICALLY LOCATED                DTHE 250
C                   DOUBLE PRECISION VECTOR                             DTHE 260
C           WORK  - WORKING STORAGE OF DIMENSION 2*N                    DTHE 270
C                   DOUBLE PRECISION ARRAY                              DTHE 280
C                                                                       DTHE 290
C        REMARKS                                                        DTHE 300
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    DTHE 310
C           WITH COEFFICIENT VECTOR POL.                                DTHE 320
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                DTHE 330
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSDTHE 340
C           THE RANGE (-C,C) IN X TO THE RANGE (ZL,ZR) IN Z WHERE       DTHE 350
C           ZL=-(C+B)/A AND ZR=(C-B)/A.                                 DTHE 360
C           FOR GIVEN ZL, ZR AND C WE HAVE A=2C/(ZR-ZL) AND             DTHE 370
C           B=-C(ZR+ZL)/(ZR-ZL)                                         DTHE 380
C                                                                       DTHE 390
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTHE 400
C           NONE                                                        DTHE 410
C                                                                       DTHE 420
C        METHOD                                                         DTHE 430
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      DTHE 440
C           FOR HERMITE POLYNOMIALS H(N,X)                              DTHE 450
C           H(N+1,X)=2*(X*H(N,X)-NM1*H(N-1,X)),                         DTHE 460
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX               DTHE 470
C           THE SECOND IS THE ARGUMENT.                                 DTHE 480
C           STARTING VALUES ARE H(0,X)=1,H(1,X)=2*X.                    DTHE 490
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        DTHE 500
C           X=A*Z+B TOGETHER WITH                                       DTHE 510
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             DTHE 520
C           =SUM(C(I)*H(I-1,X), SUMMED OVER I FROM 1 TO N).             DTHE 530
C                                                                       DTHE 540
C     ..................................................................DTHE 550
C                                                                       DTHE 560
      SUBROUTINE DTHEP(A,B,POL,N,C,WORK)                                DTHE 570
C                                                                       DTHE 580
      DIMENSION POL(1),C(1),WORK(1)                                     DTHE 590
      DOUBLE PRECISION A,B,POL,C,WORK,H,P,FI,XD,XO                      DTHE 600
C                                                                       DTHE 610
C        TEST OF DIMENSION                                              DTHE 620
      IF(N-1)2,1,3                                                      DTHE 630
C                                                                       DTHE 640
C        DIMENSION LESS THAN 2                                          DTHE 650
    1 POL(1)=C(1)                                                       DTHE 660
    2 RETURN                                                            DTHE 670
C                                                                       DTHE 680
    3 XD=A+A                                                            DTHE 690
      XO=B+B                                                            DTHE 700
      POL(1)=C(1)+C(2)*XO                                               DTHE 710
      POL(2)=C(2)*XD                                                    DTHE 720
      IF(N-2)2,2,4                                                      DTHE 730
C                                                                       DTHE 740
C        INITIALIZATION                                                 DTHE 750
    4 WORK(1)=1.D0                                                      DTHE 760
      WORK(2)=XO                                                        DTHE 770
      WORK(3)=0.D0                                                      DTHE 780
      WORK(4)=XD                                                        DTHE 790
      FI=2.D0                                                           DTHE 800
C                                                                       DTHE 810
C        CALCULATE COEFFICIENT VECTOR OF NEXT HERMITE POLYNOMIAL        DTHE 820
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              DTHE 830
      DO 6 J=3,N                                                        DTHE 840
      P=0.D0                                                            DTHE 850
C                                                                       DTHE 860
      DO 5 K=2,J                                                        DTHE 870
      H=P*XO+WORK(2*K-2)*XO-FI*WORK(2*K-3)                              DTHE 880
      P=WORK(2*K-2)                                                     DTHE 890
      WORK(2*K-2)=H                                                     DTHE 900
      WORK(2*K-3)=P                                                     DTHE 910
    5 POL(K-1)=POL(K-1)+H*C(J)                                          DTHE 920
      WORK(2*J-1)=0.D0                                                  DTHE 930
      WORK(2*J)=P*XD                                                    DTHE 940
      FI=FI+2.D0                                                        DTHE 950
    6 POL(J)=C(J)*WORK(2*J)                                             DTHE 960
      RETURN                                                            DTHE 970
      END                                                               DTHE 980
```

Subroutines LAP and DLAP

These subroutines compute the values of the normalized Laguerre polynomials for a given argument x and orders zero up to N. The polynomial $L_n(x)/n!$, termed $L_n(x)$ below, satisfies the recurrence equation

$$L_{n+1}(x) = \left[(2n+1-x)\,L_n(x) - n\,L_{n-1}(x)\right]/(n+1)$$

with starting values $L_0(x) = 1$, $L_1(x) = 1 - x$.

For reasons of economy and numerical stability the recurrence equation is used in the form

$$L_{n+1}(x) = L_n(x) - L_{n-1}(x) + L_n(x)$$

$$-\left[(1+x)\,L_n(x) - L_{n-1}(x)\right]/(n+1).$$

For large values of n the last term is negligible, giving the approximation

$$L_{n+1}(x) = 2\,L_n(x) - L_{n-1}(x).$$

This form shows that roundoff errors grow at worst linearly as long as x is small compared to n.

If e_{n+r} is the error in $L_{n+r}(x)$ due to a single rounding error e in $L_n(x)$, the approximation is:

$$e_{n+r+1} = 2 \cdot e_{n+r} - e_{n+r-1}$$

with initial conditions $e_n = e$, $e_{n-1} = 0$. This implies:

$$e_{n+1} = 2e,\ e_{n+2} = 3 \cdot e,\ \ldots,\ e_{n+r} = (r+1)e$$

The order is assumed to be zero for negative values of N.

```
C                                                                       LAP   10
C     ..................................................................LAP   20
C                                                                       LAP   30
C        SUBROUTINE LAP                                                 LAP   40
C                                                                       LAP   50
C        PURPOSE                                                        LAP   60
C           COMPUTE THE VALUES OF THE LAGUERRE POLYNOMIALS L(N,X)       LAP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  LAP   80
C                                                                       LAP   90
C        USAGE                                                          LAP  100
C           CALL LAP(Y,X,N)                                             LAP  110
C                                                                       LAP  120
C        DESCRIPTION OF PARAMETERS                                      LAP  130
C           Y     - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESLAP  140
C                   OF LAGUERRE POLYNOMIALS OF ORDER 0 UP TO N          LAP  150
C                   FOR GIVEN ARGUMENT X.                               LAP  160
C                   VALUES ARE ORDERED FROM LOW TO HIGH ORDER           LAP  170
C           X     - ARGUMENT OF LAGUERRE POLYNOMIAL                     LAP  180
C           N     - ORDER OF LAGUERRE POLYNOMIAL                        LAP  190
C                                                                       LAP  200
C        REMARKS                                                        LAP  210
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     LAP  220
C                                                                       LAP  230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  LAP  240
C           NONE                                                        LAP  250
C                                                                       LAP  260
C        METHOD                                                         LAP  270
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          LAP  280
C           LAGUERRE POLYNOMIALS L(N,X)                                 LAP  290
C           L(N+1,X)=2*L(N,X)-L(N-1,X)-((1+X)*L(N,X)-L(N-1,X))/(N+1),   LAP  300
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              LAP  310
C           THE SECOND IS THE ARGUMENT.                                 LAP  320
C           STARTING VALUES ARE L(0,X)=1, L(1,X)=1.-X.                  LAP  330
C                                                                       LAP  340
C     ..................................................................LAP  350
C                                                                       LAP  360
```

```
      SUBROUTINE LAP(Y,X,N)
C
      DIMENSION Y(1)
C
C        TEST OF ORDER
      Y(1)=1.
      IF(N)1,1,2
    1 RETURN
C
    2 Y(2)=1.-X
      IF(N-1)1,1,3
C
C        INITIALIZATION
    3 T=1.+X
C
      DO 4 I=2,N
    4 Y(I+1)=Y(I)-Y(I-1)+Y(I)-(T*Y(I)-Y(I-1))/FLOAT(I)
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DLAP
C
C        PURPOSE
C           COMPUTE THE VALUES OF THE LAGUERRE POLYNOMIALS L(N,X)
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.
C
C        USAGE
C           CALL DLAP(Y,X,N)
C
C        DESCRIPTION OF PARAMETERS
C           Y      - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUES
C                    OF LAGUERRE POLYNOMIALS OF ORDER 0 UP TO N
C                    FOR GIVEN ARGUMENT X.
C                    DOUBLE PRECISION VECTOR.
C                    VALUES ARE ORDERED FROM LOW TO HIGH ORDER
C           X      - ARGUMENT OF LAGUERRE POLYNOMIAL
C                    DOUBLE PRECISION VARIABLE.
C           N      - ORDER OF LAGUERRE POLYNOMIAL
C
C        REMARKS
C           N LESS THAN 0 IS TREATED AS IF N WERE 0
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR
C           LAGUERRE POLYNOMIALS L(N,X)
C           L(N+1,X)=2*L(N,X)-L(N-1,X)-((1+X)*L(N,X)-L(N-1,X))/(N+1),
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,
C           THE SECOND IS THE ARGUMENT.
C           STARTING VALUES ARE L(0,X)=1, L(1,X)=1.-X.
C
C     ..................................................................
C
      SUBROUTINE DLAP(Y,X,N)
C
      DIMENSION Y(1)
      DOUBLE PRECISION Y,X,T
C
C        TEST OF ORDER
      Y(1)=1.D0
      IF(N)1,1,2
    1 RETURN
C
    2 Y(2)=1.D0-X
      IF(N-1)1,1,3
C
C        INITIALIZATION
    3 T=1.D0+X
C
      DO 4 I=2,N
    4 Y(I+1)=Y(I)-Y(I-1)+Y(I)-(T*Y(I)-Y(I-1))/DFLOAT(I)
      RETURN
      END
```

Subroutines LAPS and DLAPS

These subroutines compute the value of a series expansion in normalized Laguerre polynomials. The polynomial $L_n(x)/n!$, termed $L_n(x)$ below, satisfies the recurrence equation

$$L_{n+1}(x) = 2 \cdot L_n(x) - L_{n-1}(x) - \left[(1 + x) \cdot L_n(x) - L_{n-1}(x)\right] /(n+1)$$

with starting values $L_0(x) = 1$, $L_1(x) = 1 - x$.

An n-term expansion in Laguerre polynomials with coefficient vector $C = (c_1, \ldots, c_n)$ is evaluated by means of a forward iteration scheme:

$$Y = c_1,\ H_0 = 1,\ H_1 = 1 - x,\ T = 1 - x$$

$$\left.\begin{aligned} H_2 &= H_1 - H_0 + H_1 - (T \cdot H_1 - H_0)/i \\ H_0 &= H_1 \\ H_1 &= H_2 \\ Y &= Y + c_i \cdot H_0 \end{aligned}\right\} \text{ for } i = 2, \ldots, n$$

This gives the result $Y(x) = \sum_{i=1}^{n} c_i \cdot L_{i-1}(x)$.

Calculation is bypassed in case of a nonpositive value of the dimension n.

```
C
C     ..................................................................
C
C        SUBROUTINE LAPS
C
C        PURPOSE
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN LAGUERRE
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X.
C
C        USAGE
C           CALL LAPS(Y,X,C,N)
C
C        DESCRIPTION OF PARAMETERS
C           Y      - RESULT VALUE
C           X      - ARGUMENT VALUE
C           C      - COEFFICIENT VECTOR OF GIVEN EXPANSION
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH
C           N      - DIMENSION OF COEFFICIENT VECTOR C
C
C        REMARKS
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DEFINITION
C           Y=SUM(C(I)*L(I-1,X), SUMMED OVER I FROM 1 TO N).
C           EVALUATION IS DONE BY MEANS OF UPWARD RECURSION
C           USING THE RECURRENCE EQUATION FOR LAGUERRE POLYNOMIALS
C           L(N+1,X)=2*L(N,X)-L(N-1,X)-((1+X)*L(N,X)-L(N-1,X))/(N+1).
C
C     ..................................................................
C
      SUBROUTINE LAPS(Y,X,C,N)
C
      DIMENSION C(1)
C
C        TEST OF DIMENSION
      IF(N)1,1,2
    1 RETURN
C
    2 Y=C(1)
      IF(N-2)1,3,3
C
C        INITIALIZATION
    3 H0=1.
      H1=1.-X
      T=1.+X
C
      DO 4 I=2,N
      H2=H1-H0+H1-(T*H1-H0)/FLOAT(I)
      H0=H1
      H1=H2
    4 Y=Y+C(I)*H0
      RETURN
      END
```

```
C                                                                       DAPS  10
C     ..................................................................DAPS  20
C                                                                       DAPS  30
C        SUBROUTINE DLAPS                                               DAPS  40
C                                                                       DAPS  50
C        PURPOSE                                                        DAPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN LAGUERRE       DAPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. DAPS  80
C                                                                       DAPS  90
C        USAGE                                                          DAPS 100
C           CALL DLAPS(Y,X,C,N)                                         DAPS 110
C                                                                       DAPS 120
C        DESCRIPTION OF PARAMETERS                                      DAPS 130
C           Y      - RESULT VALUE                                       DAPS 140
C                    DOUBLE PRECISION VARIABLE                          DAPS 150
C           X      - ARGUMENT VALUE                                     DAPS 160
C                    DOUBLE PRECISION VARIABLE                          DAPS 170
C           C      - COEFFICIENT VECTOR OF GIVEN EXPANSION              DAPS 180
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DAPS 190
C                    DOUBLE PRECISION VECTOR                            DAPS 200
C           N      - DIMENSION OF COEFFICIENT VECTOR C                  DAPS 210
C                                                                       DAPS 220
C        REMARKS                                                        DAPS 230
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 DAPS 240
C                                                                       DAPS 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DAPS 260
C           NONE                                                        DAPS 270
C                                                                       DAPS 280
C        METHOD                                                         DAPS 290
C           DEFINITION                                                  DAPS 300
C           Y=SUM(C(I)*L(I-1,X), SUMMED OVER I FROM 1 TO N).            DAPS 310
C           EVALUATION IS DONE BY MEANS OF UPWARD RECURSION             DAPS 320
C           USING THE RECURRENCE EQUATION FOR LAGUERRE POLYNOMIALS      DAPS 330
C           L(N+1,X)=2*L(N,X)-L(N-1,X)-((1+X)*L(N,X)-L(N-1,X))/(N+1).   DAPS 340
C                                                                       DAPS 350
C     ..................................................................DAPS 360
C                                                                       DAPS 370
      SUBROUTINE DLAPS(Y,X,C,N)                                         DAPS 380
C                                                                       DAPS 390
      DIMENSION C(1)                                                    DAPS 400
      DOUBLE PRECISION C,Y,X,H0,H1,H2,T                                 DAPS 410
C                                                                       DAPS 420
C        TEST OF DIMENSION                                              DAPS 430
      IF(N)1,1,2                                                        DAPS 440
    1 RETURN                                                            DAPS 450
C                                                                       DAPS 460
    2 Y=C(1)                                                            DAPS 470
      IF(N-2)1,3,3                                                      DAPS 480
C                                                                       DAPS 490
C        INITIALIZATION                                                 DAPS 500
    3 H0=1.D0                                                           DAPS 510
      H1=1.D0-X                                                         DAPS 520
      T=1.D0+X                                                          DAPS 530
      DO 4 I=2,N                                                        DAPS 540
      H2=H1-H0+H1-(T*H1-H0)/DFLOAT(I)                                   DAPS 550
      H0=H1                                                             DAPS 560
      H1=H2                                                             DAPS 570
    4 Y=Y+C(I)*H0                                                       DAPS 580
      RETURN                                                            DAPS 590
      END                                                               DAPS 600
```

Subroutines TLAP and DTLAP

These subroutines transform a given series expansion in Laguerre polynomials to a polynomial. The independent variable x of the given expansion is thereby substituted by the independent variable z of the resulting polynomial using the linear transformation $x = A * z + B$ or $z = (x-B)/A$.

This means:

$$\sum_{i=1}^{N} C_i * L_{i-1}(A * z + B)$$

$$= \sum_{i=1}^{N} POL_i * z^{i-1}$$

Calculation is bypassed in case of a nonpositive value of the dimension N.

The range (0, c) in x is transformed to the range (z_l, z_r) in z with $z_l = -B/A$ and $z_r = (c - B)/A$, or vice versa, $A = c/(z_r - z_l)$ and $B = -c * z_l/(z_r - z_l)$.

The Laguerre polynomial $L_n(x)$ satisfies the recurrence equation

$$L_{n+1}(x) = x/(n+1) \cdot L_n(x) + \left[2 - 1/(n+1)\right] L_n(x) - \left[1 - 1/(n+1)\right] L_{n-1}(x)$$

with starting values $L_0(x) = 1$, $L_1(x) = 1 - x$.

The transformation is performed by means of a forward iteration scheme:

1. The coefficient vector of the Laguerre polynomial $L_i(x)$ is calculated from the coefficient vectors of $L_{i-1}(x)$, $L_{i-2}(x)$ using the recurrence equation, for $i = 3, \ldots, N$.
2. The resulting polynomial coefficient vector is obtained by summation of c_i times $L_{i-1}(x)$ over i, for $i = 1, \ldots, N$.

```
C                                                                       TLAP  10
C     ..................................................................TLAP  20
C                                                                       TLAP  30
C        SUBROUTINE TLAP                                                TLAP  40
C                                                                       TLAP  50
C        PURPOSE                                                        TLAP  60
C           A SERIES EXPANSION IN LAGUERRE POLYNOMIALS WITH INDEPENDENT TLAP  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  TLAP  80
C           VARIABLE Z, WHERE X=A*Z+B                                   TLAP  90
C                                                                       TLAP 100
C        USAGE                                                          TLAP 110
C           CALL TLAP(A,B,POL,N,C,WORK)                                 TLAP 120
C                                                                       TLAP 130
C        DESCRIPTION OF PARAMETERS                                      TLAP 140
C           A      - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONTLAP 150
C           B      - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION      TLAP 160
C           POL    - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE) TLAP 170
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH         TLAP 180
C           N      - DIMENSION OF COEFFICIENT VECTORS POL AND C        TLAP 190
C           C      - GIVEN COEFFICIENT VECTOR OF EXPANSION             TLAP 200
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH         TLAP 210
C                    POL AND C MAY BE IDENTICALLY LOCATED              TLAP 220
C           WORK   - WORKING STORAGE OF DIMENSION 2*N                  TLAP 230
C                                                                       TLAP 240
C        REMARKS                                                        TLAP 250
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    TLAP 260
C           WITH COEFFICIENT VECTOR POL.                                TLAP 270
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                TLAP 280
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSTLAP 290
C           THE RANGE (0,C) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE       TLAP 300
C           ZL=-B/A AND ZR=(C-B)/A.                                     TLAP 310
C           FOR GIVEN ZL, ZR AND C WE HAVE A=C/(ZR-ZL) AND              TLAP 320
C           B=-C*ZL/(ZR-ZL)                                             TLAP 330
C                                                                       TLAP 340
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TLAP 350
C           NONE                                                        TLAP 360
C                                                                       TLAP 370
C        METHOD                                                         TLAP 380
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      TLAP 390
C           FOR LAGUERRE POLYNOMIALS L(N,X)                             TLAP 400
C           L(N+1,X)=2*L(N,X)-L(N-1,X)-((1+X)*L(N,X)-L(N-1,X))/(N+1),   TLAP 410
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              TLAP 420
C           THE SECOND IS THE ARGUMENT.                                 TLAP 430
C           STARTING VALUES ARE L(0,X)=1, L(1,X)=1-X.                   TLAP 440
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        TLAP 450
C           X=A*Z+B TOGETHER WITH                                       TLAP 460
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             TLAP 470
C           =SUM(C(I)*L(I-1,X), SUMMED OVER I FROM 1 TO N).             TLAP 480
C                                                                       TLAP 490
C     ..................................................................TLAP 500
C                                                                       TLAP 510
      SUBROUTINE TLAP(A,B,POL,N,C,WORK)                                 TLAP 520
C                                                                       TLAP 530
      DIMENSION POL(1),C(1),WORK(1)                                     TLAP 540
C                                                                       TLAP 550
C        TEST OF DIMENSION                                              TLAP 560
      IF(N-1)2,1,3                                                      TLAP 570
C                                                                       TLAP 580
C        DIMENSION LESS THAN 2                                          TLAP 590
    1 POL(1)=C(1)                                                       TLAP 600
    2 RETURN                                                            TLAP 610
C                                                                       TLAP 620
    3 POL(1)=C(1)+C(2)-B*C(2)                                           TLAP 630
      POL(2)=-C(2)*A                                                    TLAP 640
      IF(N-2)2,2,4                                                      TLAP 650
C                                                                       TLAP 660
C        INITIALIZATION                                                 TLAP 670
    4 WORK(1)=1.                                                        TLAP 680
      WORK(2)=1.D0-B                                                    TLAP 690
      WORK(3)=0.                                                        TLAP 700
      WORK(4)=-A                                                        TLAP 710
      FI=1.                                                             TLAP 720
C                                                                       TLAP 730
C        CALCULATE COEFFICIENT VECTOR OF NEXT LAGUERRE POLYNOMIAL       TLAP 740
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              TLAP 750
      DO 6 J=3,N                                                        TLAP 760
      FI=FI+1.                                                          TLAP 770
      Q=1./FI                                                           TLAP 780
      Q1=Q-1.                                                           TLAP 790
      Q2=1.-Q1-B*Q                                                      TLAP 800
      Q=Q*A                                                             TLAP 810
      P=0.                                                              TLAP 820
C                                                                       TLAP 830
      DO 5 K=2,J                                                        TLAP 840
      H=-P*Q+WORK(2*K-2)*Q2+WORK(2*K-3)*Q1                              TLAP 850
      P=WORK(2*K-2)                                                     TLAP 860
      WORK(2*K-2)=H                                                     TLAP 870
      WORK(2*K-3)=P                                                     TLAP 880
    5 POL(K-1)=POL(K-1)+H*C(J)                                          TLAP 890
      WORK(2*J-1)=0.                                                    TLAP 900
      WORK(2*J)=-Q*P                                                    TLAP 910
    6 POL(J)=C(J)*WORK(2*J)                                             TLAP 920
      RETURN                                                            TLAP 930
      END                                                               TLAP 940
```

```
C                                                                       DTLA  10
C     ..................................................................DTLA  20
C                                                                       DTLA  30
C        SUBROUTINE DTLAP                                               DTLA  40
C                                                                       DTLA  50
C        PURPOSE                                                        DTLA  60
C           A SERIES EXPANSION IN LAGUERRE POLYNOMIALS WITH INDEPENDENT DTLA  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  DTLA  80
C           VARIABLE Z, WHERE X=A*Z+B                                   DTLA  90
C                                                                       DTLA 100
C        USAGE                                                          DTLA 110
C           CALL DTLAP(A,B,POL,N,C,WORK)                                DTLA 120
C                                                                       DTLA 130
C        DESCRIPTION OF PARAMETERS                                      DTLA 140
C           A      - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONDTLA 150
C                    DOUBLE PRECISION VARIABLE                          DTLA 160
C           B      - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION      DTLA 170
C                    DOUBLE PRECISION VARIABLE                          DTLA 180
C           POL    - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE) DTLA 190
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DTLA 200
C                    DOUBLE PRECISION VECTOR                            DTLA 210
C           N      - DIMENSION OF COEFFICIENT VECTORS POL AND C        DTLA 220
C           C      - GIVEN COEFFICIENT VECTOR OF EXPANSION             DTLA 230
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DTLA 240
C                    POL AND C MAY BE IDENTICALLY LOCATED               DTLA 250
C                    DOUBLE PRECISION VECTOR                            DTLA 260
C           WORK   - WORKING STORAGE OF DIMENSION 2*N                  DTLA 270
C                    DOUBLE PRECISION ARRAY                             DTLA 280
C                                                                       DTLA 290
C        REMARKS                                                        DTLA 300
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    DTLA 310
C           WITH COEFFICIENT VECTOR POL.                                DTLA 320
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                DTLA 330
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSDTLA 340
C           THE RANGE (0,C) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE       DTLA 350
C           ZL=-B/A AND ZR=(C-B)/A.                                     DTLA 360
C           FOR GIVEN ZL, ZR AND C WE HAVE A=C/(ZR-ZL) AND              DTLA 370
C           B=-C*ZL/(ZR-ZL)                                             DTLA 380
C                                                                       DTLA 390
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTLA 400
C           NONE                                                        DTLA 410
C                                                                       DTLA 420
C        METHOD                                                         DTLA 430
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      DTLA 440
C           FOR LAGUERRE POLYNOMIALS L(N,X)                             DTLA 450
C           L(N+1,X)=2*L(N,X)-L(N-1,X)-((1+X)*L(N,X)-L(N-1,X))/(N+1),   DTLA 460
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              DTLA 470
C           THE SECOND IS THE ARGUMENT.                                 DTLA 480
C           STARTING VALUES ARE L(0,X)=1, L(1,X)=1-X.                   DTLA 490
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        DTLA 500
C           X=A*Z+B TOGETHER WITH                                       DTLA 510
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             DTLA 520
C           =SUM(C(I)*L(I-1,X), SUMMED OVER I FROM 1 TO N).             DTLA 530
C                                                                       DTLA 540
C     ..................................................................DTLA 550
C                                                                       DTLA 560
      SUBROUTINE DTLAP(A,B,POL,N,C,WORK)                                DTLA 570
C                                                                       DTLA 580
      DIMENSION POL(1),C(1),WORK(1)                                     DTLA 590
      DOUBLE PRECISION A,B,POL,C,WORK,H,P,Q,Q1,Q2,FI                    DTLA 600
C                                                                       DTLA 610
C        TEST OF DIMENSION                                              DTLA 620
      IF(N-1)2,1,3                                                      DTLA 630
C                                                                       DTLA 640
C        DIMENSION LESS THAN 2                                          DTLA 650
    1 POL(1)=C(1)                                                       DTLA 660
    2 RETURN                                                            DTLA 670
C                                                                       DTLA 680
    3 POL(1)=C(1)+C(2)-B*C(2)                                           DTLA 690
      POL(2)=-C(2)*A                                                    DTLA 700
      IF(N-2)2,2,4                                                      DTLA 710
C                                                                       DTLA 720
C        INITIALIZATION                                                 DTLA 730
    4 WORK(1)=1.D0                                                      DTLA 740
      WORK(2)=1.D0-B                                                    DTLA 750
      WORK(3)=0.D0                                                      DTLA 760
      WORK(4)=-A                                                        DTLA 770
      FI=1.D0                                                           DTLA 780
C                                                                       DTLA 790
C        CALCULATE COEFFICIENT VECTOR OF NEXT LAGUERRE POLYNOMIAL       DTLA 800
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              DTLA 810
      DO 6 J=3,N                                                        DTLA 820
      FI=FI+1.D0                                                        DTLA 830
      Q=1.D0/FI                                                         DTLA 840
      Q1=Q-1.D0                                                         DTLA 850
      Q2=1.D0-Q1-B*Q                                                    DTLA 860
      Q=Q*A                                                             DTLA 870
      P=0.D0                                                            DTLA 880
C                                                                       DTLA 890
      DO 5 K=2,J                                                        DTLA 900
      H=-P*Q+WORK(2*K-2)*Q2+WORK(2*K-3)*Q1                              DTLA 910
      P=WORK(2*K-2)                                                     DTLA 920
      WORK(2*K-2)=H                                                     DTLA 930
      WORK(2*K-3)=P                                                     DTLA 940
    5 POL(K-1)=POL(K-1)+H*C(J)                                          DTLA 950
      WORK(2*J-1)=0.D0                                                  DTLA 960
      WORK(2*J)=-Q*P                                                    DTLA 970
    6 POL(J)=C(J)*WORK(2*J)                                             DTLA 980
      RETURN                                                            DTLA 990
      END                                                               DTLA1000
```

Subroutines LEP and DLEP

These subroutines compute the values of the Legendre polynomials for a given argument x and orders zero up to N. The Legendre polynomial $P_n(x)$ satisfies the recurrence equation

$$P_{n+1}(x) = ((2n+1) \cdot x \cdot P_n(x) - n \cdot P_{n-1}(x))/(n+1)$$

with starting values $P_0(x) = 1$, $P_1(x) = x$.

For reasons of economy and numerical stability the recurrence equation is used in the form:

$$P_{n+1}(x) = x \cdot P_n(x) - P_{n-1}(x) + x \cdot P_n(x) - (x \cdot P_n(x) - P_{n-1}(x))/(n+1)$$

For large values of n the last term is negligible, giving the approximation:

$$P_{n+1}(x) = 2 \cdot x \cdot P_n(x) - P_{n-1}(x)$$

This form shows that roundoff errors grow at worst linearly, assuming that the argument x is absolutely less than one.

If e_{n+r} is the error in $P_{n+r}(x)$ due to a single rounding error e in $P_n(x)$, the approximation is

$$e_{n+r+1} = 2x \cdot e_{n+r} - e_{n+r-1}$$

with initial conditions $e_n = e$, $e_{n-1} = 0$. The solution of this difference equation has its maximum for $|x| = 1$:

$$e_{n-1} = 0,\ e_n = e,\ |e_{n+1}| = 2e,\ \ldots,\ |e_{n+r}| = (r+1)e$$

The order is assumed to be zero for negative values of N.

```
C                                                                       LEP   10
C     ..................................................................LEP   20
C                                                                       LEP   30
C        SUBROUTINE LEP                                                 LEP   40
C                                                                       LEP   50
C        PURPOSE                                                        LEP   60
C           COMPUTE THE VALUES OF THE LEGENDRE POLYNOMIALS P(N,X)       LEP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  LEP   80
C                                                                       LEP   90
C        USAGE                                                          LEP  100
C           CALL LEP(Y,X,N)                                             LEP  110
C                                                                       LEP  120
C        DESCRIPTION OF PARAMETERS                                      LEP  130
C           Y       - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESLEP  140
C                     OF LEGENDRE POLYNOMIALS OF ORDER 0 UP TO N        LEP  150
C                     FOR GIVEN ARGUMENT X.                             LEP  160
C                     VALUES ARE ORDERED FROM LOW TO HIGH ORDER         LEP  170
C           X       - ARGUMENT OF LEGENDRE POLYNOMIAL                   LEP  180
C           N       - ORDER OF LEGENDRE POLYNOMIAL                      LEP  190
C                                                                       LEP  200
C        REMARKS                                                        LEP  210
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     LEP  220
C                                                                       LEP  230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  LEP  240
C           NONE                                                        LEP  250
C                                                                       LEP  260
C        METHOD                                                         LEP  270
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          LEP  280
C           LEGENDRE POLYNOMIALS P(N,X)                                 LEP  290
C           P(N+1,X)=2*X*P(N,X)-P(N-1,X)-(X*P(N,X)-P(N-1,X))/(N+1),     LEP  300
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              LEP  310
C           THE SECOND IS THE ARGUMENT.                                 LEP  320
C           STARTING VALUES ARE P(0,X)=1, P(1,X)=X.                     LEP  330
C                                                                       LEP  340
C     ..................................................................LEP  350
C                                                                       LEP  360
      SUBROUTINE LEP(Y,X,N)                                             LEP  370
C                                                                       LEP  380
      DIMENSION Y(1)                                                    LEP  390
C                                                                       LEP  400
C        TEST OF ORDER                                                  LEP  410
      Y(1)=1.                                                           LEP  420
      IF(N)1,1,2                                                        LEP  430
    1 RETURN                                                            LEP  440
C                                                                       LEP  450
    2 Y(2)=X                                                            LEP  460
      IF(N-1)1,1,3                                                      LEP  470
C                                                                       LEP  480
    3 DO 4 I=2,N                                                        LEP  490
      G=X*Y(I)                                                          LEP  500
    4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/FLOAT(I)                             LEP  510
      RETURN                                                            LEP  520
      END                                                               LEP  530
```

```
C                                                                       DEP   10
C     ..................................................................DEP   20
C                                                                       DEP   30
C        SUBROUTINE DLEP                                                DEP   40
C                                                                       DEP   50
C        PURPOSE                                                        DEP   60
C           COMPUTE THE VALUES OF THE LEGENDRE POLYNOMIALS P(N,X)       DEP   70
C           FOR ARGUMENT VALUE X AND ORDERS 0 UP TO N.                  DEP   80
C                                                                       DEP   90
C        USAGE                                                          DEP  100
C           CALL DLEP(Y,X,N)                                            DEP  110
C                                                                       DEP  120
C        DESCRIPTION OF PARAMETERS                                      DEP  130
C           Y       - RESULT VECTOR OF DIMENSION N+1 CONTAINING THE VALUESDEP  140
C                     OF LEGENDRE POLYNOMIALS OF ORDER 0 UP TO N        DEP  150
C                     FOR GIVEN ARGUMENT X.                             DEP  160
C                     DOUBLE PRECISION VECTOR.                          DEP  170
C                     VALUES ARE ORDERED FROM LOW TO HIGH ORDER         DEP  180
C           X       - ARGUMENT OF LEGENDRE POLYNOMIAL                   DEP  190
C                     DOUBLE PRECISION VARIABLE.                        DEP  200
C           N       - ORDER OF LEGENDRE POLYNOMIAL                      DEP  210
C                                                                       DEP  220
C        REMARKS                                                        DEP  230
C           N LESS THAN 0 IS TREATED AS IF N WERE 0                     DEP  240
C                                                                       DEP  250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DEP  260
C           NONE                                                        DEP  270
C                                                                       DEP  280
C        METHOD                                                         DEP  290
C           EVALUATION IS BASED ON THE RECURRENCE EQUATION FOR          DEP  300
C           LEGENDRE POLYNOMIALS P(N,X)                                 DEP  310
C           P(N+1,X)=2*X*P(N,X)-P(N-1,X)-(X*P(N,X)-P(N-1,X))/(N+1),     DEP  320
C           WHERE THE FIRST TERM IN BRACKETS IS THE ORDER,              DEP  330
C           THE SECOND IS THE ARGUMENT.                                 DEP  340
C           STARTING VALUES ARE P(0,X)=1, P(1,X)=X.                     DEP  350
C                                                                       DEP  360
C     ..................................................................DEP  370
C                                                                       DEP  380
      SUBROUTINE DLEP(Y,X,N)                                            DEP  390
C                                                                       DEP  400
      DIMENSION Y(1)                                                    DEP  410
      DOUBLE PRECISION Y,X,G                                            DEP  420
C                                                                       DEP  430
C        TEST OF ORDER                                                  DEP  440
      Y(1)=1.D0                                                         DEP  450
      IF(N)1,1,2                                                        DEP  460
    1 RETURN                                                            DEP  470
C                                                                       DEP  480
    2 Y(2)=X                                                            DEP  490
      IF(N-1)1,1,3                                                      DEP  500
C                                                                       DEP  510
    3 DO 4 I=2,N                                                        DEP  520
      G=X*Y(I)                                                          DEP  530
    4 Y(I+1)=G-Y(I-1)+G-(G-Y(I-1))/DFLOAT(I)                            DEP  540
      RETURN                                                            DEP  550
      END                                                               DEP  560
```

Subroutines LEPS and DLEPS

These subroutines compute the value of a series expansion in Legendre polynomials. The Legendre polynomial $P_n(x)$ satisfies the recurrence equation

$$P_{n+1}(x) = 2x \cdot P_n(x) - P_{n-1}(x) - (x \cdot P_n(x) - P_{n-1}(x))/(n+1)$$

with starting values $P_0(x) = 1$, $P_1(x) = x$.

An n-term expansion in Legendre polynomials with coefficient vector $C = (c_1, \ldots, c_n)$ is evaluated by means of a forward iteration scheme:

$$Y = c_1, \quad H_0 = 1, \quad H_1 = x$$

$$\left.\begin{aligned} H_2 &= x \cdot H_1 \\ H_2 &= H_2 - H_0 + H_2 - (H_2 - H_0)/i \\ H_0 &= H_1 \\ H_1 &= H_2 \\ Y &= Y + c_i \cdot H_0 \end{aligned}\right\} \quad \text{for } i = 2, \ldots, n$$

This gives the result $Y(x) = \sum_{i=1}^{n} c_i \cdot P_{i-1}(x)$.

Calculation is bypassed in case of a nonpositive value of the dimension N.

```
C                                                                       LEPS  10
C     ..................................................................LEPS  20
C                                                                       LEPS  30
C        SUBROUTINE LEPS                                                LEPS  40
C                                                                       LEPS  50
C        PURPOSE                                                        LEPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN LEGENDRE       LEPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. LEPS  80
C                                                                       LEPS  90
C        USAGE                                                          LEPS 100
C           CALL LEPS(Y,X,C,N)                                          LEPS 110
C                                                                       LEPS 120
C        DESCRIPTION OF PARAMETERS                                      LEPS 130
C           Y     - RESULT VALUE                                        LEPS 140
C           X     - ARGUMENT VALUE                                      LEPS 150
C           C     - COEFFICIENT VECTOR OF GIVEN EXPANSION               LEPS 160
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           LEPS 170
C           N     - DIMENSION OF COEFFICIENT VECTOR C                   LEPS 180
C                                                                       LEPS 190
C        REMARKS                                                        LEPS 200
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 LEPS 210
C                                                                       LEPS 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  LEPS 230
C           NONE                                                        LEPS 240
C                                                                       LEPS 250
C        METHOD                                                         LEPS 260
C           DEFINITION                                                  LEPS 270
C           Y=SUM(C(I)*P(I-1,X), SUMMED OVER I FROM 1 TO N).            LEPS 280
C           EVALUATION IS DONE BY MEANS OF UPWARD RECURSION             LEPS 290
C           USING THE RECURRENCE EQUATION FOR LEGENDRE POLYNOMIALS      LEPS 300
C           P(N+1,X)=2*X*P(N,X)-P(N-1,X)-(X*P(N,X)-P(N-1,X))/(N+1).     LEPS 310
C                                                                       LEPS 320
C     ..................................................................LEPS 330
C                                                                       LEPS 340
      SUBROUTINE LEPS(Y,X,C,N)                                          LEPS 350
C                                                                       LEPS 360
      DIMENSION C(1)                                                    LEPS 370
C                                                                       LEPS 380
C        TEST OF DIMENSION                                              LEPS 390
      IF(N)1,1,2                                                        LEPS 400
    1 RETURN                                                            LEPS 410
C                                                                       LEPS 420
    2 Y=C(1)                                                            LEPS 430
      IF(N-2)1,3,3                                                      LEPS 440
C                                                                       LEPS 450
C        INITIALIZATION                                                 LEPS 460
    3 H0=1.                                                             LEPS 470
      H1=X                                                              LEPS 480
C                                                                       LEPS 490
      DO 4 I=2,N                                                        LEPS 500
      H2=X*H1                                                           LEPS 510
      H2=H2-H0+H2-(H2-H0)/FLOAT(I)                                      LEPS 520
      H0=H1                                                             LEPS 530
      H1=H2                                                             LEPS 540
    4 Y=Y+C(I)*H0                                                       LEPS 550
      RETURN                                                            LEPS 560
      END                                                               LEPS 570
```

```
C                                                                       DEPS  10
C     ..................................................................DEPS  20
C                                                                       DEPS  30
C        SUBROUTINE DLEPS                                               DEPS  40
C                                                                       DEPS  50
C        PURPOSE                                                        DEPS  60
C           COMPUTES THE VALUE OF AN N-TERM EXPANSION IN LEGENDRE       DEPS  70
C           POLYNOMIALS WITH COEFFICIENT VECTOR C FOR ARGUMENT VALUE X. DEPS  80
C                                                                       DEPS  90
C        USAGE                                                          DEPS 100
C           CALL DLEPS(Y,X,C,N)                                         DEPS 110
C                                                                       DEPS 120
C        DESCRIPTION OF PARAMETERS                                      DEPS 130
C           Y     - RESULT VALUE                                        DEPS 140
C                   DOUBLE PRECISION VARIABLE                           DEPS 150
C           X     - ARGUMENT VALUE                                      DEPS 160
C                   DOUBLE PRECISION VARIABLE                           DEPS 170
C           C     - COEFFICIENT VECTOR OF GIVEN EXPANSION               DEPS 180
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH           DEPS 190
C                   DOUBLE PRECISION VECTOR                             DEPS 200
C           N     - DIMENSION OF COEFFICIENT VECTOR C                   DEPS 210
C                                                                       DEPS 220
C        REMARKS                                                        DEPS 230
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1                 DEPS 240
C                                                                       DEPS 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DEPS 260
C           NONE                                                        DEPS 270
C                                                                       DEPS 280
C        METHOD                                                         DEPS 290
C           DEFINITION                                                  DEPS 300
C           Y=SUM(C(I)*P(I-1,X), SUMMED OVER I FROM 1 TO N).            DEPS 310
C           EVALUATION IS DONE BY MEANS OF UPWARD RECURSION             DEPS 320
C           USING THE RECURRENCE EQUATION FOR LEGENDRE POLYNOMIALS      DEPS 330
C           P(N+1,X)=2*X*P(N,X)-P(N-1,X)-(X*P(N,X)-P(N-1,X))/(N+1).     DEPS 340
C                                                                       DEPS 350
C     ..................................................................DEPS 360
C                                                                       DEPS 370
      SUBROUTINE DLEPS(Y,X,C,N)                                         DEPS 380
C                                                                       DEPS 390
      DIMENSION C(1)                                                    DEPS 400
      DOUBLE PRECISION C,Y,X,H0,H1,H2                                   DEPS 410
C                                                                       DEPS 420
C        TEST OF DIMENSION                                              DEPS 430
      IF(N)1,1,2                                                        DEPS 440
    1 RETURN                                                            DEPS 450
C                                                                       DEPS 460
    2 Y=C(1)                                                            DEPS 470
      IF(N-2)1,3,3                                                      DEPS 480
C                                                                       DEPS 490
C        INITIALIZATION                                                 DEPS 500
    3 H0=1.D0                                                           DEPS 510
      H1=X                                                              DEPS 520
C                                                                       DEPS 530
      DO 4 I=2,N                                                        DEPS 540
      H2=X*H1                                                           DEPS 550
      H2=H2-H0+H2-(H2-H0)/DFLOAT(I)                                     DEPS 560
      H0=H1                                                             DEPS 570
      H1=H2                                                             DEPS 580
    4 Y=Y+C(I)*H0                                                       DEPS 590
      RETURN                                                            DEPS 600
      END                                                               DEPS 610
```

Subroutines TLEP and DTLEP

These subroutines transform a given series expansion in Legendre polynomials to a polynomial. The independent variable x of the given expansion is thereby substituted by the independent variable z of the resulting polynomial using the linear transformation $x = A * z + B$ or $z = (x - B)/A$.

This means:

$$\sum_{i=1}^{N} C_i * P_{i-1}(A * z + B)$$

$$= \sum_{i=1}^{N} POL_i * z^{i-1}$$

Calculation is bypassed in case of a nonpositive value of the dimension N.

The range (-1, +1) in x is transformed to the range (z_1, z_r) in z with $z_1 = -(1 + B)/A$ and $z_r = (1 - B)/A$, or vice versa, $A = 2/(z_r - z_1)$ and $B = -(z_r + z_1)/(z_r - z_1)$.

The Legendre polynomial $P_n(x)$ satisfies the recurrence equation

$$P_{n+1}(x) = x\left[2-1/(n+1)\right] P_n(x) - \left[1-1/(n+1)\right] P_{n-1}(x)$$

with starting values $P_0(x) = 1$, $P_1(x) = x$.

The transformation is performed by means of a forward iteration scheme:

1. The coefficient vector of the Legendre polynomial $P_i(x)$ is calculated from the coefficient vectors of $P_{i-1}(x)$, $P_{i-2}(x)$ using the recurrence equation, for $i = 3, \ldots, N$.
2. The resulting polynomial coefficient vector is obtained by summation of c_i times coefficient vector of $P_{i-1}(x)$ over i, for $i = 1, \ldots, N$.

```
C                                                                       TLEP  10
C     ...................................................................TLEP  20
C                                                                       TLEP  30
C        SUBROUTINE TLEP                                                TLEP  40
C                                                                       TLEP  50
C        PURPOSE                                                        TLEP  60
C           A SERIES EXPANSION IN LEGENDRE POLYNOMIALS WITH INDEPENDENT TLEP  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  TLEP  80
C           VARIABLE Z, WHERE X=A*Z+B                                   TLEP  90
C                                                                       TLEP 100
C        USAGE                                                          TLEP 110
C           CALL TLEP(A,B,POL,N,C,WORK)                                 TLEP 120
C                                                                       TLEP 130
C        DESCRIPTION OF PARAMETERS                                      TLEP 140
C           A      - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONTLEP 150
C           B      - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION       TLEP 160
C           POL    - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE) TLEP 170
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          TLEP 180
C           N      - DIMENSION OF COEFFICIENT VECTORS POL AND C         TLEP 190
C           C      - GIVEN COEFFICIENT VECTOR OF EXPANSION              TLEP 200
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          TLEP 210
C                    POL AND C MAY BE IDENTICALLY LOCATED               TLEP 220
C           WORK   - WORKING STORAGE OF DIMENSION 2*N                   TLEP 230
C                                                                       TLEP 240
C        REMARKS                                                        TLEP 250
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    TLEP 260
C           WITH COEFFICIENT VECTOR POL.                                TLEP 270
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                TLEP 280
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSTLEP 290
C           THE RANGE (-1,+1) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE     TLEP 300
C           ZL=-(1+B)/A AND ZR=(1-B)/A.                                 TLEP 310
C           FOR GIVEN ZL, ZR WE HAVE A=2/(ZR-ZL) AND B=-(ZR+ZL)/(ZR-ZL) TLEP 320
C                                                                       TLEP 330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TLEP 340
C           NONE                                                        TLEP 350
C                                                                       TLEP 360
C        METHOD                                                         TLEP 370
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      TLEP 380
C           FOR LEGENDRE POLYNOMIALS P(N,X)                             TLEP 390
C           P(N+1,X)=2*X*P(N,X)-P(N-1,X)-(X*P(N,X)-P(N-1,X))/(N+1),     TLEP 400
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              TLEP 410
C           THE SECOND IS THE ARGUMENT.                                 TLEP 420
C           STARTING VALUES ARE P(0,X)=1, P(1,X)=X.                     TLEP 430
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        TLEP 440
C           X=A*Z+B TOGETHER WITH                                       TLEP 450
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             TLEP 460
C           =SUM(C(I)*P(I-1,X), SUMMED OVER I FROM 1 TO N).             TLEP 470
C                                                                       TLEP 480
C     ...................................................................TLEP 490
C                                                                       TLEP 500
      SUBROUTINE TLEP(A,B,POL,N,C,WORK)                                 TLEP 510
C                                                                       TLEP 520
      DIMENSION POL(1),C(1),WORK(1)                                     TLEP 530
C                                                                       TLEP 540
C        TEST OF DIMENSION                                              TLEP 550
      IF(N-1)2,1,3                                                      TLEP 560
C                                                                       TLEP 570
C        DIMENSION LESS THAN 2                                          TLEP 580
    1 POL(1)=C(1)                                                       TLEP 590
    2 RETURN                                                            TLEP 600
C                                                                       TLEP 610
    3 POL(1)=C(1)+B*C(2)                                                TLEP 620
      POL(2)=A*C(2)                                                     TLEP 630
      IF(N-2)2,2,4                                                      TLEP 640
C                                                                       TLEP 650
C        INITIALIZATION                                                 TLEP 660
    4 WORK(1)=1.                                                        TLEP 670
      WORK(2)=B                                                         TLEP 680
      WORK(3)=0.                                                        TLEP 690
      WORK(4)=A                                                         TLEP 700
      FI=1.                                                             TLEP 710
C                                                                       TLEP 720
C        CALCULATE COEFFICIENT VECTOR OF NEXT LEGENDRE POLYNOMIAL       TLEP 730
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              TLEP 740
      DO 6 J=3,N                                                        TLEP 750
      FI=FI+1.                                                          TLEP 760
      Q=1./FI-1.                                                        TLEP 770
      Q1=1.-Q                                                           TLEP 780
      P=0.                                                              TLEP 790
C                                                                       TLEP 800
      DO 5 K=2,J                                                        TLEP 810
      H=(A*P+B*WORK(2*K-2))*Q1+Q*WORK(2*K-3)                            TLEP 820
      P=WORK(2*K-2)                                                     TLEP 830
      WORK(2*K-2)=H                                                     TLEP 840
      WORK(2*K-3)=P                                                     TLEP 850
    5 POL(K-1)=POL(K-1)+H*C(J)                                          TLEP 860
      WORK(2*J-1)=0.                                                    TLEP 870
      WORK(2*J)=A*P*Q1                                                  TLEP 880
    6 POL(J)=C(J)*WORK(2*J)                                             TLEP 890
      RETURN                                                            TLEP 900
      END                                                               TLEP 910
```

```
C                                                                       DTLE  10
C     ...................................................................DTLE  20
C                                                                       DTLE  30
C        SUBROUTINE DTLEP                                               DTLE  40
C                                                                       DTLE  50
C        PURPOSE                                                        DTLE  60
C           A SERIES EXPANSION IN LEGENDRE POLYNOMIALS WITH INDEPENDENT DTLE  70
C           VARIABLE X IS TRANSFORMED TO A POLYNOMIAL WITH INDEPENDENT  DTLE  80
C           VARIABLE Z, WHERE X=A*Z+B                                   DTLE  90
C                                                                       DTLE 100
C        USAGE                                                          DTLE 110
C           CALL DTLEP(A,B,POL,N,C,WORK)                                DTLE 120
C                                                                       DTLE 130
C        DESCRIPTION OF PARAMETERS                                      DTLE 140
C           A      - FACTOR OF LINEAR TERM IN GIVEN LINEAR TRANSFORMATIONDTLE 150
C                    DOUBLE PRECISION VARIABLE                          DTLE 160
C           B      - CONSTANT TERM IN GIVEN LINEAR TRANSFORMATION       DTLE 170
C                    DOUBLE PRECISION VARIABLE                          DTLE 180
C           POL    - COEFFICIENT VECTOR OF POLYNOMIAL (RESULTANT VALUE) DTLE 190
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DTLE 200
C                    DOUBLE PRECISION VECTOR                            DTLE 210
C           N      - DIMENSION OF COEFFICIENT VECTORS POL AND C         DTLE 220
C           C      - GIVEN COEFFICIENT VECTOR OF EXPANSION              DTLE 230
C                    COEFFICIENTS ARE ORDERED FROM LOW TO HIGH          DTLE 240
C                    POL AND C MAY BE IDENTICALLY LOCATED               DTLE 250
C                    DOUBLE PRECISION VECTOR                            DTLE 260
C           WORK   - WORKING STORAGE OF DIMENSION 2*N                   DTLE 270
C                    DOUBLE PRECISION ARRAY                             DTLE 280
C                                                                       DTLE 290
C        REMARKS                                                        DTLE 300
C           COEFFICIENT VECTOR C REMAINS UNCHANGED IF NOT COINCIDING    DTLE 310
C           WITH COEFFICIENT VECTOR POL.                                DTLE 320
C           OPERATION IS BYPASSED IN CASE N LESS THAN 1.                DTLE 330
C           THE LINEAR TRANSFORMATION X=A*Z+B OR Z=(1/A)(X-B) TRANSFORMSDTLE 340
C           THE RANGE (-1,+1) IN X TO THE RANGE (ZL,ZR) IN Z, WHERE     DTLE 350
C           ZL=-(1+B)/A AND ZR=(1-B)/A.                                 DTLE 360
C           FOR GIVEN ZL, ZR WE HAVE A=2/(ZR-ZL) AND B=-(ZR+ZL)/(ZR-ZL) DTLE 370
C                                                                       DTLE 380
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTLE 390
C           NONE                                                        DTLE 400
C                                                                       DTLE 410
C        METHOD                                                         DTLE 420
C           THE TRANSFORMATION IS BASED ON THE RECURRENCE EQUATION      DTLE 430
C           FOR LEGENDRE POLYNOMIALS P(N,X)                             DTLE 440
C           P(N+1,X)=2*X*P(N,X)-P(N-1,X)-(X*P(N,X)-P(N-1,X))/(N+1),     DTLE 450
C           WHERE THE FIRST TERM IN BRACKETS IS THE INDEX,              DTLE 460
C           THE SECOND IS THE ARGUMENT.                                 DTLE 470
C           STARTING VALUES ARE P(0,X)=1, P(1,X)=X.                     DTLE 480
C           THE TRANSFORMATION IS IMPLICITLY DEFINED BY MEANS OF        DTLE 490
C           X=A*Z+B TOGETHER WITH                                       DTLE 500
C           SUM(POL(I)*Z**(I-1), SUMMED OVER I FROM 1 TO N)             DTLE 510
C           =SUM(C(I)*P(I-1,X), SUMMED OVER I FROM 1 TO N).             DTLE 520
C                                                                       DTLE 530
C     ...................................................................DTLE 540
C                                                                       DTLE 550
      SUBROUTINE DTLEP(A,B,POL,N,C,WORK)                                DTLE 560
C                                                                       DTLE 570
      DIMENSION POL(1),C(1),WORK(1)                                     DTLE 580
      DOUBLE PRECISION A,B,POL,C,WORK,H,P,Q,Q1,FI                       DTLE 590
C                                                                       DTLE 600
C        TEST OF DIMENSION                                              DTLE 610
      IF(N-1)2,1,3                                                      DTLE 620
C                                                                       DTLE 630
C        DIMENSION LESS THAN 2                                          DTLE 640
    1 POL(1)=C(1)                                                       DTLE 650
    2 RETURN                                                            DTLE 660
C                                                                       DTLE 670
    3 POL(1)=C(1)+B*C(2)                                                DTLE 680
      POL(2)=A*C(2)                                                     DTLE 690
      IF(N-2)2,2,4                                                      DTLE 700
C                                                                       DTLE 710
C        INITIALIZATION                                                 DTLE 720
    4 WORK(1)=1.D0                                                      DTLE 730
      WORK(2)=B                                                         DTLE 740
      WORK(3)=0.D0                                                      DTLE 750
      WORK(4)=A                                                         DTLE 760
      FI=1.D0                                                           DTLE 770
C                                                                       DTLE 780
C        CALCULATE COEFFICIENT VECTOR OF NEXT LEGENDRE POLYNOMIAL       DTLE 790
C        AND ADD MULTIPLE OF THIS VECTOR TO POLYNOMIAL POL              DTLE 800
      DO 6 J=3,N                                                        DTLE 810
      FI=FI+1.D0                                                        DTLE 820
      Q=1.D0/FI-1.D0                                                    DTLE 830
      Q1=1.D0-Q                                                         DTLE 840
      P=0.D0                                                            DTLE 850
C                                                                       DTLE 860
      DO 5 K=2,J                                                        DTLE 870
      H=(A*P+B*WORK(2*K-2))*Q1+Q*WORK(2*K-3)                            DTLE 880
      P=WORK(2*K-2)                                                     DTLE 890
      WORK(2*K-2)=H                                                     DTLE 900
      WORK(2*K-3)=P                                                     DTLE 910
    5 POL(K-1)=POL(K-1)+H*C(J)                                          DTLE 920
      WORK(2*J-1)=0.D0                                                  DTLE 930
      WORK(2*J)=A*P*Q1                                                  DTLE 940
    6 POL(J)=C(J)*WORK(2*J)                                             DTLE 950
      RETURN                                                            DTLE 960
      END                                                               DTLE 970
```

Roots of Nonlinear Equations

Subroutines RTWI and DRTWI

These subroutines refine the initial guess x_0 of a root of the general nonlinear equation $x = f(x)$. Wegstein's iteration scheme is used in order to get accelerated convergence in case of a function f (x), which has at least continuous first derivative in the range in which iteration moves.

Following Figure 14, set $x_1 = y_0 = f(x_0)$ and $y_1 = f(x_1)$.

Refinement of x_1 is done by determination of the intersection of the linear function $y = x$ and the secant through the points (x_0, y_0) and (x_1, y_1), thus getting:

$$x_2 = x_1 + \frac{x_1 - x_0}{\dfrac{x_0 - y_0}{x_1 - y_1} - 1}$$

and $y_2 = f(x_2)$

The next step is done by starting at (x_2, y_2) and setting:

$$x_3 = x_2 + \frac{x_2 - x_1}{\dfrac{x_1 - y_1}{x_2 - y_2} - 1}$$

$$y_3 = f(x_3)$$

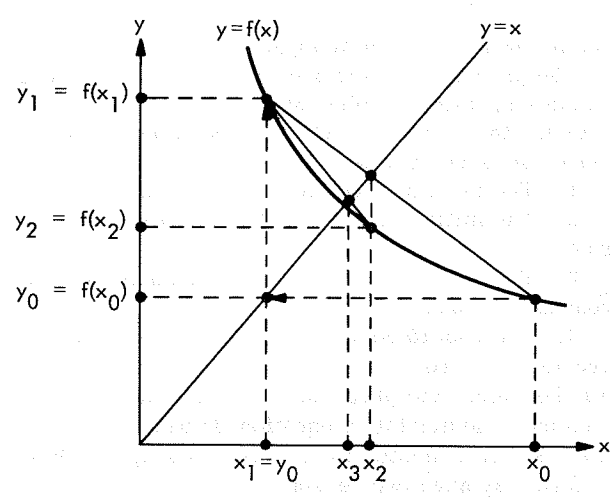


Figure 14. Wegstein's iterative method

It is easily seen that this determines the intersection between $y = x$ and the secant through the points (x_1, y_1) and (x_2, y_2). Therefore Wegstein's iteration scheme is often called the secant modification of the normal iteration scheme $x_{i+1} = f(x_i)$.

Repeating these steps, the result is the iteration scheme:

$$\left.\begin{aligned} x_{i+1} &= x_i + \frac{x_i - x_{i-1}}{\dfrac{x_{i-1} - y_{i-1}}{x_i - y_i} - 1} \\ y_{i+1} &= f(x_{i+1}) \end{aligned}\right\} (i = 1, 2, \ldots) \quad (1)$$

Each step requires one evaluation of $f(x)$.

This iterative procedure is terminated if the following two conditions are satisfied:

$$\left.\begin{aligned} &\delta_1 \leqq \quad \text{and} \quad \delta_2 \leqq 10 \cdot \varepsilon \\ &\text{with} \quad \delta_1 = \begin{cases} \left|\dfrac{x_{i+1} - x_i}{x_{i+1}}\right| & \text{in case of } |x_{i+1}| > 1 \\ \left|x_{i+1} - x_i\right| & \text{in case of } |x_{i+1}| \leqq 1, \end{cases} \\ &\qquad\qquad \delta_2 = \begin{cases} \left|\dfrac{x_{i+1} - y_{i+1}}{x_{i+1}}\right| & \text{in case of } |x_{i+1}| > 1 \\ \left|x_{i+1} - y_{i+1}\right| & \text{in case of } |x_{i+1}| \leqq 1 \end{cases} \end{aligned}\right\} (2)$$

and tolerance ε given by input.

The procedure described above may not converge within a specified number of iteration steps. Reasons for this behavior, which is indicated by an error message may be:

1. Too few iteration steps are specified.
2. The initial guess x_0 is too far away from any root.
3. The tolerance ε is too small with respect to roundoff errors.
4. The root to be determined is of multiplicity greater than one.

Furthermore, the procedure fails if at any iteration step the denominator of equation (1) becomes zero. This is also indicated by an error message. This failure may have two reasons:

1. The secant has the slope 1, either exactly or due to roundoff errors. In both cases it is probable that there is at least one point ξ in the range in which iteration moves with $f'(\xi) = 1$.
2. $x_i = x_{i-1}$ and $x_i \neq y_i = f(x_i)$. This case is possible due to roundoff errors or to a very steep slope of the secant.

For reference see G. N. Lance, Numerical Methods for High Speed Computers, Iliffe, London, 1960, pp. 134 - 138.

```
C                                                                       RTWI  10
C     ...................................................................RTWI  20
C                                                                       RTWI  30
C        SUBROUTINE RTWI                                                RTWI  40
C                                                                       RTWI  50
C        PURPOSE                                                        RTWI  60
C           TO SOLVE GENERAL NONLINEAR EQUATIONS OF THE FORM X=FCT(X)   RTWI  70
C           BY MEANS OF WEGSTEIN-S ITERATION METHOD.                    RTWI  80
C                                                                       RTWI  90
C        USAGE                                                          RTWI 100
C           CALL RTWI (X,VAL,FCT,XST,EPS,IEND,IER)                      RTWI 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.               RTWI 120
C                                                                       RTWI 130
C        DESCRIPTION OF PARAMETERS                                      RTWI 140
C           X     - RESULTANT ROOT OF EQUATION X=FCT(X).                RTWI 150
C           VAL   - RESULTANT VALUE OF X-FCT(X) AT ROOT X.              RTWI 160
C           FCT   - NAME OF THE EXTERNAL FUNCTION SUBPROGRAM USED.      RTWI 170
C           XST   - INPUT VALUE WHICH SPECIFIES THE INITIAL GUESS OF    RTWI 180
C                   THE ROOT X.                                         RTWI 190
C           EPS   - INPUT VALUE WHICH SPECIFIES THE UPPER BOUND OF THE  RTWI 200
C                   ERROR OF RESULT X.                                  RTWI 210
C           IEND  - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.        RTWI 220
C           IER   - RESULTANT ERROR PARAMETER CODED AS FOLLOWS          RTWI 230
C                     IER=0 - NO ERROR,                                 RTWI 240
C                     IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS,RTWI 250
C                     IER=2 - AT ANY ITERATION STEP THE DENOMINATOR OF  RTWI 260
C                             ITERATION FORMULA WAS EQUAL TO ZERO.      RTWI 270
C                                                                       RTWI 280
C        REMARKS                                                        RTWI 290
C           THE PROCEDURE IS BYPASSED AND GIVES THE ERROR MESSAGE IER=2 RTWI 300
C           IF AT ANY ITERATION STEP THE DENOMINATOR OF ITERATION       RTWI 310
C           FORMULA WAS EQUAL TO ZERO. THAT MEANS THAT THERE IS AT      RTWI 320
C           LEAST ONE POINT IN THE RANGE IN WHICH ITERATION MOVES WITH  RTWI 330
C           DERIVATIVE OF FCT(X) EQUAL TO 1.                            RTWI 340
C                                                                       RTWI 350
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RTWI 360
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   RTWI 370
C           BY THE USER.                                                RTWI 380
C                                                                       RTWI 390
C        METHOD                                                         RTWI 400
C           SOLUTION OF EQUATION X=FCT(X) IS DONE BY MEANS OF           RTWI 410
C           WEGSTEIN-S ITERATION METHOD, WHICH STARTS AT THE INITIAL    RTWI 420
C           GUESS XST OF A ROOT X. ONE ITERATION STEP REQUIRES ONE      RTWI 430
C           EVALUATION OF FCT(X). FOR TEST ON SATISFACTORY ACCURACY SEE RTWI 440
C           FORMULAE (2) OF MATHEMATICAL DESCRIPTION.                   RTWI 450
C           FOR REFERENCE, SEE                                          RTWI 460
C           (1) G. N. LANCE, NUMERICAL METHODS FOR HIGH SPEED COMPUTERS,RTWI 470
C               ILIFFE, LONDON, 1960, PP.134-138,                       RTWI 480
C           (2) J. WEGSTEIN, ALGORITHM 2, CACM, VOL.3, ISS.2 (1960),    RTWI 490
C               PP.74,                                                  RTWI 500
C           (3) H.C. THACHER, ALGORITHM 15, CACM, VOL.3, ISS.8 (1960),  RTWI 510
C               PP.475,                                                 RTWI 520
C           (4) J.G. HERRIOT, ALGORITHM 26, CACM, VOL.3, ISS.11 (1960), RTWI 530
C               PP.603.                                                 RTWI 540
C                                                                       RTWI 550
C     ...................................................................RTWI 560
C                                                                       RTWI 570
      SUBROUTINE RTWI(X,VAL,FCT,XST,EPS,IEND,IER)                       RTWI 580
C                                                                       RTWI 590
C                                                                       RTWI 600
C     PREPARE ITERATION                                                 RTWI 610
      IER=0                                                             RTWI 620
      TOL=XST                                                           RTWI 630
      X=FCT(TOL)                                                        RTWI 640
      A=X-XST                                                           RTWI 650
      B=-A                                                              RTWI 660
      TOL=X                                                             RTWI 670
      VAL=X-FCT(TOL)                                                    RTWI 680
C                                                                       RTWI 690
C                                                                       RTWI 700
C     START ITERATION LOOP                                              RTWI 710
      DO 6 I=1,IEND                                                     RTWI 720
      IF(VAL)1,7,1                                                      RTWI 730
C                                                                       RTWI 740
C     EQUATION IS NOT SATISFIED BY X                                    RTWI 750
    1 B=B/VAL-1.                                                        RTWI 760
      IF(B)2,8,2                                                        RTWI 770
C                                                                       RTWI 780
C     ITERATION IS POSSIBLE                                             RTWI 790
    2 A=A/B                                                             RTWI 800
      X=X+A                                                             RTWI 810
      B=VAL                                                             RTWI 820
      TOL=X                                                             RTWI 830
      VAL=X-FCT(TOL)                                                    RTWI 840
C                                                                       RTWI 850
C     TEST ON SATISFACTORY ACCURACY                                     RTWI 860
      TOL=EPS                                                           RTWI 870
      D=ABS(X)                                                          RTWI 880
      IF(D-1.)4,4,3                                                     RTWI 890
    3 TOL=TOL*D                                                         RTWI 900
    4 IF(ABS(A)-TOL)5,5,6                                               RTWI 910
    5 IF(ABS(VAL)-10.*TOL)7,7,6                                         RTWI 920
    6 CONTINUE                                                          RTWI 930
C     END OF ITERATION LOOP                                             RTWI 940
C                                                                       RTWI 950
C                                                                       RTWI 960
C     NO CONVERGENCE AFTER IEND ITERATION STEPS. ERROR RETURN.          RTWI 970
      IER=1                                                             RTWI 980
    7 RETURN                                                            RTWI 990
C                                                                       RTWI1000
C     ERROR RETURN IN CASE OF ZERO DIVISOR                              RTWI1010
    8 IER=2                                                             RTWI1020
      RETURN                                                            RTWI1030
      END                                                               RTWI1040
```

```
C                                                                       DRTW  10
C     ..................................................................DRTW  20
C                                                                       DRTW  30
C        SUBROUTINE DRTWI                                               DRTW  40
C                                                                       DRTW  50
C        PURPOSE                                                        DRTW  60
C           TO SOLVE GENERAL NONLINEAR EQUATIONS OF THE FORM X=FCT(X)   DRTW  70
C           BY MEANS OF WEGSTEIN-S ITERATION METHOD.                    DRTW  80
C                                                                       DRTW  90
C        USAGE                                                          DRTW 100
C           CALL DRTWI (X,VAL,FCT,XST,EPS,IEND,IER)                     DRTW 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.               DRTW 120
C                                                                       DRTW 130
C        DESCRIPTION OF PARAMETERS                                      DRTW 140
C           X      - DOUBLE PRECISION RESULTANT ROOT OF EQUATION        DRTW 150
C                    X=FCT(X).                                          DRTW 160
C           VAL    - DOUBLE PRECISION RESULTANT VALUE OF X-FCT(X)       DRTW 170
C                    AT ROOT X.                                         DRTW 180
C           FCT    - NAME OF THE EXTERNAL DOUBLE PRECISION FUNCTION     DRTW 190
C                    SUBPROGRAM USED.                                   DRTW 200
C           XST    - DOUBLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTW 210
C                    INITIAL GUESS OF THE ROOT X.                       DRTW 220
C           EPS    - SINGLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTW 230
C                    UPPER BOUND OF THE ERROR OF RESULT X.              DRTW 240
C           IEND   - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.       DRTW 250
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS         DRTW 260
C                      IER=0 - NO ERROR,                                DRTW 270
C                      IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS,DRTW 280
C                      IER=2 - AT ANY ITERATION STEP THE DENOMINATOR OF DRTW 290
C                              ITERATION FORMULA WAS EQUAL TO ZERO.     DRTW 300
C                                                                       DRTW 310
C        REMARKS                                                        DRTW 320
C           THE PROCEDURE IS BYPASSED AND GIVES THE ERROR MESSAGE IER=2 DRTW 330
C           IF AT ANY ITERATION STEP THE DENOMINATOR OF ITERATION       DRTW 340
C           FORMULA WAS EQUAL TO ZERO. THAT MEANS THAT THERE IS AT      DRTW 350
C           LEAST ONE POINT IN THE RANGE IN WHICH ITERATION MOVES WITH  DRTW 360
C           DERIVATIVE OF FCT(X) EQUAL TO 1.                            DRTW 370
C                                                                       DRTW 380
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DRTW 390
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DRTW 400
C           MUST BE FURNISHED BY THE USER.                              DRTW 410
C                                                                       DRTW 420
C        METHOD                                                         DRTW 430
C           SOLUTION OF EQUATION X=FCT(X) IS DONE BY MEANS OF           DRTW 440
C           WEGSTEIN-S ITERATION METHOD, WHICH STARTS AT THE INITIAL    DRTW 450
C           GUESS XST OF A ROOT X. ONE ITERATION STEP REQUIRES ONE      DRTW 460
C           EVALUATION OF FCT(X). FOR TEST ON SATISFACTORY ACCURACY SEE DRTW 470
C           FORMULAE (2) OF MATHEMATICAL DESCRIPTION.                   DRTW 480
C           FOR REFERENCE, SEE                                          DRTW 490
C           (1) G. N. LANCE, NUMERICAL METHODS FOR HIGH SPEED COMPUTERS,DRTW 500
C               ILIFFE, LONDON, 1960, PP.134-138,                       DRTW 510
C           (2) J. WEGSTEIN, ALGORITHM 2, CACM, VOL.3, ISS.2 (1960),    DRTW 520
C               PP.74,                                                  DRTW 530
C           (3) H.C. THACHER, ALGORITHM 15, CACM, VOL.3, ISS.8 (1960),  DRTW 540
C               PP.475,                                                 DRTW 550
C           (4) J.G. HERRIOT, ALGORITHM 26, CACM, VOL.3, ISS.11 (1960), DRTW 560
C               PP.603.                                                 DRTW 570
C                                                                       DRTW 580
C     ..................................................................DRTW 590
C                                                                       DRTW 600
      SUBROUTINE DRTWI(X,VAL,FCT,XST,EPS,IEND,IER)                      DRTW 610
C                                                                       DRTW 620
C                                                                       DRTW 630
      DOUBLE PRECISION X,VAL,FCT,XST,A,B,D,TOL                          DRTW 640
C                                                                       DRTW 650
C     PREPARE ITERATION                                                 DRTW 660
      IER=0                                                             DRTW 670
      TOL=XST                                                           DRTW 680
      X=FCT(TOL)                                                        DRTW 690
      A=X-XST                                                           DRTW 700
      B=-A                                                              DRTW 710
      TOL=X                                                             DRTW 720
      VAL=X-FCT(TOL)                                                    DRTW 730
C                                                                       DRTW 740
C                                                                       DRTW 750
C     START ITERATION LOOP                                              DRTW 760
      DO 6 I=1,IEND                                                     DRTW 770
      IF(VAL)1,7,1                                                      DRTW 780
C                                                                       DRTW 790
C     EQUATION IS NOT SATISFIED BY X                                    DRTW 800
    1 B=B/VAL-1.D0                                                      DRTW 810
      IF(B)2,8,2                                                        DRTW 820
C                                                                       DRTW 830
C     ITERATION IS POSSIBLE                                             DRTW 840
    2 A=A/B                                                             DRTW 850
      X=X+A                                                             DRTW 860
      B=VAL                                                             DRTW 870
      TOL=X                                                             DRTW 880
      VAL=X-FCT(TOL)                                                    DRTW 890
C                                                                       DRTW 900
C     TEST ON SATISFACTORY ACCURACY                                     DRTW 910
      TOL=EPS                                                           DRTW 920
      D=DABS(X)                                                         DRTW 930
      IF(D-1.D0)4,4,3                                                   DRTW 940
    3 TOL=TOL*D                                                         DRTW 950
    4 IF(DABS(A)-TOL)5,5,6                                              DRTW 960
    5 IF(DABS(VAL)-1.D1*TOL)7,7,6                                       DRTW 970
    6 CONTINUE                                                          DRTW 980
C     END OF ITERATION LOOP                                             DRTW 990
C                                                                       DRTW1000
C                                                                       DRTW1010
C     NO CONVERGENCE AFTER IEND ITERATION STEPS. ERROR RETURN.          DRTW1020
      IER=1                                                             DRTW1030
    7 RETURN                                                            DRTW1040
C                                                                       DRTW1050
C     ERROR RETURN IN CASE OF ZERO DIVISOR                              DRTW1060
    8 IER=2                                                             DRTW1070
      RETURN                                                            DRTW1080
      END                                                               DRTW1090
```

Subroutines RTMI and DRTMI

These subroutines determine a root of the general nonlinear equation $f(x) = 0$ in the range of x from x_{li} up to x_{ri} (x_{li}, x_{ri} given by input) by means of Mueller's iteration scheme of successive bisection and inverse parabolic interpolation. The procedure assumes $f(x_{li}) \cdot f(x_{ri}) \leq 0$.

Starting with $x_l = x_{li}$ and $x_r = x_{ri}$ and following Fig. 15, one iteration step is described.

First, the middle of the interval $x_l \ldots x_r$ is computed:

$$x_m = \frac{1}{2}(x_l + x_r)$$

In case $f(x_m) \cdot f(x_r) < 0$, x_l and x_r are interchanged to ensure that $f(x_m) \cdot f(x_r) > 0$.

In case

$$2\, f(x_m)\,[f(x_m) - f(x_l)] - f(x_r)\,[f(x_r) - f(x_l)] \geq 0 \qquad (1)$$

x_r is replaced by x_m and the bisection step is repeated. If, after a specified number of successive bisections, inequality (1) is still satisfied, the procedure is bypassed and an error message is given.

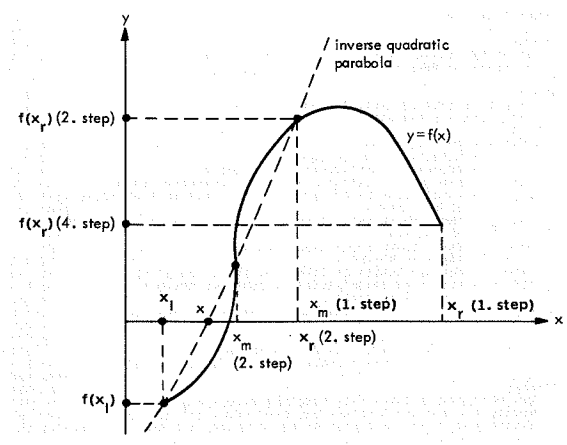
In Fig. 15, the second bisection step leads to a configuration which does not satisfy inequality (1). Thus by inverse parabolic interpolation:

$$\Delta x = f(x_l)\,\frac{x_m - x_l}{f(x_m) - f(x_l)}$$

$$\left\{1 + f(x_m)\,\frac{f(x_r) - 2\,f(x_m) + f(x_l)}{[f(x_r) - f(x_m)]\,[f(x_r) - f(x_l)]}\right\} \qquad (2)$$

$$\text{and } x = x_l - \Delta x$$

and x is sure to be situated between x_l and x_m.



• Figure 15. Mueller's iterative method

Now, for the next iteration step, x becomes x_l and x_m becomes x_r if $f(x) \cdot f(x_l) > 0$, or x becomes x_r if $f(x) \cdot f(x_l) < 0$.

Convergence is either quadratic or linear if the multiplicity of the root to be determined is equal to one or greater than one respectively, and if f(x) can be differentiated continuously at least twice in the range $x_{li} \cdots x_{ri}$. Each iteration step requires two evaluations of f(x).

This iterative procedure is terminated if either the two conditions (checked in bisection loop)

$$\left.\begin{aligned} |x_r - x_l| &\leq \varepsilon \cdot \max(1, |x_r|) \\ \text{and} \quad |f(x_r) - f(x_l)| &\leq 100 \cdot \varepsilon \end{aligned}\right\} \qquad (3)$$

or the two conditions (checked after inverse parabolic interpolation)

$$\left.\begin{aligned} |\Delta x| &\leq \varepsilon \cdot \max(1, |x|) \\ \text{and} \quad |f(x)| &\leq 100 \cdot \varepsilon \end{aligned}\right\} \qquad (4)$$

are satisfied, where tolerance ε is given by input.

The procedure described above may not converge within a specified number of iteration steps followed by the same number of successive bisections. Reasons for this behaviour, which is indicated by an error message, may be:

1. Too few iteration steps are specified.
2. The initial interval $x_{li} \cdots x_{ri}$ is too long.
3. The tolerance ε is too small with respect to roundoff errors.

Furthermore, the procedure is bypassed, also giving an error message, if the basic assumption $f(x_{li}) \cdot f(x_{ri}) \leq 0$ is not satisfied.

For reference see G. K. Kristiansen, "Zero of Arbitrary Function", BIT, vol. 3 (1963), pp. 205-206.

```
C
C     ..................................................................
C
C        SUBROUTINE RTMI
C
C        PURPOSE
C           TO SOLVE GENERAL NONLINEAR EQUATIONS OF THE FORM FCT(X)=0
C           BY MEANS OF MUELLER-S ITERATION METHOD.
C
C        USAGE
C           CALL RTMI (X,F,FCT,XLI,XRI,EPS,IEND,IER)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.
C
C        DESCRIPTION OF PARAMETERS
C           X      - RESULTANT ROOT OF EQUATION FCT(X)=0.
C           F      - RESULTANT FUNCTION VALUE AT ROOT X.
C           FCT    - NAME OF THE EXTERNAL FUNCTION SUBPROGRAM USED.
C           XLI    - INPUT VALUE WHICH SPECIFIES THE INITIAL LEFT BOUND
C                    OF THE ROOT X.
C           XRI    - INPUT VALUE WHICH SPECIFIES THE INITIAL RIGHT BOUND
C                    OF THE ROOT X.
C           EPS    - INPUT VALUE WHICH SPECIFIES THE UPPER BOUND OF THE
C                    ERROR OF RESULT X.
C           IEND   - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS
C                      IER=0 - NO ERROR,
C                      IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS
C                              FOLLOWED BY IEND SUCCESSIVE STEPS OF
C                              BISECTION,
C                      IER=2 - BASIC ASSUMPTION FCT(XLI)*FCT(XRI) LESS
C                              THAN OR EQUAL TO ZERO IS NOT SATISFIED.
C
C        REMARKS
C           THE PROCEDURE ASSUMES THAT FUNCTION VALUES AT INITIAL
C           BOUNDS XLI AND XRI HAVE NOT THE SAME SIGN. IF THIS BASIC
C           ASSUMPTION IS NOT SATISFIED BY INPUT VALUES XLI AND XRI, THER
C           PROCEDURE IS BYPASSED AND GIVES THE ERROR MESSAGE IER=2.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           SOLUTION OF EQUATION FCT(X)=0 IS DONE BY MEANS OF MUELLER-S
C           ITERATION METHOD OF SUCCESSIVE BISECTIONS AND INVERSE
C           PARABOLIC INTERPOLATION, WHICH STARTS AT THE INITIAL BOUNDS
C           XLI AND XRI. CONVERGENCE IS QUADRATIC IF THE DERIVATIVE OF
C           FCT(X) AT ROOT X IS NOT EQUAL TO ZERO. ONE ITERATION STEP
C           REQUIRES TWO EVALUATIONS OF FCT(X). FOR TEST ON SATISFACTORY
C           ACCURACY SEE FORMULAE (3,4) OF MATHEMATICAL DESCRIPTION.
C           FOR REFERENCE, SEE G. K. KRISTIANSEN, ZERO OF ARBITRARY
C           FUNCTION, BIT, VOL. 3 (1963), PP.205-206.
C
C     ..................................................................
C
      SUBROUTINE RTMI(X,F,FCT,XLI,XRI,EPS,IEND,IER)
C
C
C     PREPARE ITERATION
      IER=0
      XL=XLI
      XR=XRI
      X=XL
      TOL=X
      F=FCT(TOL)
      IF(F)1,16,1
    1 FL=F
      X=XR
      TOL=X
      F=FCT(TOL)
      IF(F)2,16,2
    2 FR=F
      IF(SIGN(1.,FL)+SIGN(1.,FR))25,3,25
C
C     BASIC ASSUMPTION FL*FR LESS THAN 0 IS SATISFIED.
C     GENERATE TOLERANCE FOR FUNCTION VALUES.
    3 I=0
      TOLF=100.*EPS
C
C
C     START ITERATION LOOP
    4 I=I+1
C
C     START BISECTION LOOP
      DO 13 K=1,IEND
      X=.5*(XL+XR)
      TOL=X
      F=FCT(TOL)
      IF(F)5,16,5
    5 IF(SIGN(1.,F)+SIGN(1.,FR))7,6,7
C
C     INTERCHANGE XL AND XR IN ORDER TO GET THE SAME SIGN IN F AND FR
    6 TOL=XL
      XL=XR
      XR=TOL
      TOL=FL
      FL=FR
      FR=TOL
    7 TOL=F-FL
      A=F*TOL
      A=A+A
      IF(A-FR*(FR-FL))8,9,9
    8 IF(I-IEND)17,17,9
    9 XR=X
      FR=F
C
C     TEST ON SATISFACTORY ACCURACY IN BISECTION LOOP
      TOL=EPS
      A=ABS(XR)
      IF(A-1.)11,11,10
   10 TOL=TOL*A
   11 IF(ABS(XR-XL)-TOL)12,12,13
   12 IF(ABS(FR-FL)-TOLF)14,14,13
   13 CONTINUE
C     END OF BISECTION LOOP
C
C     NO CONVERGENCE AFTER IEND ITERATION STEPS FOLLOWED BY IEND
C     SUCCESSIVE STEPS OF BISECTION OR STEADILY INCREASING FUNCTION
C     VALUES AT RIGHT BOUNDS. ERROR RETURN.
      IER=1
   14 IF(ABS(FR)-ABS(FL))16,16,15
   15 X=XL
      F=FL
   16 RETURN
C
C     COMPUTATION OF ITERATED X-VALUE BY INVERSE PARABOLIC INTERPOLATION
   17 A=FR-F
      DX=(X-XL)*FL*(1.+F*(A-TOL)/(A*(FR-FL)))/TOL
      XM=X
      FM=F
      X=XL-DX
      TOL=X
      F=FCT(TOL)
      IF(F)18,16,18
C
C     TEST ON SATISFACTORY ACCURACY IN ITERATION LOOP
   18 TOL=EPS
      A=ABS(X)
      IF(A-1.)20,20,19
   19 TOL=TOL*A
   20 IF(ABS(DX)-TOL)21,21,22
   21 IF(ABS(F)-TOLF)16,16,22
C
C     PREPARATION OF NEXT BISECTION LOOP
   22 IF(SIGN(1.,F)+SIGN(1.,FL))24,23,24
   23 XR=X
      FR=F
      GO TO 4
   24 XL=X
      FL=F
      XR=XM
      FR=FM
      GO TO 4
C     END OF ITERATION LOOP
C
C
C     ERROR RETURN IN CASE OF WRONG INPUT DATA
   25 IER=2
      RETURN
      END
```

```
C                                                                       DRTM  10
C     ..................................................................DRTM  20
C                                                                       DRTM  30
C        SUBROUTINE DRTMI                                               DRTM  40
C                                                                       DRTM  50
C        PURPOSE                                                        DRTM  60
C           TO SOLVE GENERAL NONLINEAR EQUATIONS OF THE FORM FCT(X)=0   DRTM  70
C           BY MEANS OF MUELLER-S ITERATION METHOD.                     DRTM  80
C                                                                       DRTM  90
C        USAGE                                                          DRTM 100
C           CALL DRTMI (X,F,FCT,XLI,XRI,EPS,IEND,IER)                   DRTM 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.               DRTM 120
C                                                                       DRTM 130
C        DESCRIPTION OF PARAMETERS                                      DRTM 140
C           X      - DOUBLE PRECISION RESULTANT ROOT OF EQUATION        DRTM 150
C                    FCT(X)=0.                                          DRTM 160
C           F      - DOUBLE PRECISION RESULTANT FUNCTION VALUE          DRTM 170
C                    AT ROOT X.                                         DRTM 180
C           FCT    - NAME OF THE EXTERNAL DOUBLE PRECISION FUNCTION     DRTM 190
C                    SUBPROGRAM USED.                                   DRTM 200
C           XLI    - DOUBLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTM 210
C                    INITIAL LEFT BOUND OF THE ROOT X.                  DRTM 220
C           XRI    - DOUBLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTM 230
C                    INITIAL RIGHT BOUND OF THE ROOT X.                 DRTM 240
C           EPS    - SINGLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTM 250
C                    UPPER BOUND OF THE ERROR OF RESULT X.              DRTM 260
C           IEND   - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.       DRTM 270
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS         DRTM 280
C                      IER=0 - NO ERROR,                                DRTM 290
C                      IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS DRTM 300
C                              FOLLOWED BY IEND SUCCESSIVE STEPS OF     DRTM 310
C                              BISECTION,                               DRTM 320
C                      IER=2 - BASIC ASSUMPTION FCT(XLI)*FCT(XRI) LESS  DRTM 330
C                              THAN OR EQUAL TO ZERO IS NOT SATISFIED.  DRTM 340
C                                                                       DRTM 350
C        REMARKS                                                        DRTM 360
C           THE PROCEDURE ASSUMES THAT FUNCTION VALUES AT INITIAL       DRTM 370
C           BOUNDS XLI AND XRI HAVE NOT THE SAME SIGN. IF THIS BASIC    DRTM 380
C           ASSUMPTION IS NOT SATISFIED BY INPUT VALUES XLI AND XRI, THEDRTM 390
C           PROCEDURE IS BYPASSED AND GIVES THE ERROR MESSAGE IER=2.    DRTM 400
C                                                                       DRTM 410
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DRTM 420
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DRTM 430
C           MUST BE FURNISHED BY THE USER.                              DRTM 440
C                                                                       DRTM 450
C        METHOD                                                         DRTM 460
C           SOLUTION OF EQUATION FCT(X)=0 IS DONE BY MEANS OF MUELLER-S DRTM 470
C           ITERATION METHOD OF SUCCESSIVE BISECTIONS AND INVERSE       DRTM 480
C           PARABOLIC INTERPOLATION, WHICH STARTS AT THE INITIAL BOUNDS DRTM 490
C           XLI AND XRI. CONVERGENCE IS QUADRATIC IF THE DERIVATIVE OF  DRTM 500
C           FCT(X) AT ROOT X IS NOT EQUAL TO ZERO. ONE ITERATION STEP   DRTM 510
C           REQUIRES TWO EVALUATIONS OF FCT(X). FOR TEST ON SATISFACTORYDRTM 520
C           ACCURACY SEE FORMULAE (3,4) OF MATHEMATICAL DESCRIPTION.    DRTM 530
C           FOR REFERENCE, SEE G. K. KRISTIANSEN, ZERO OF ARBITRARY     DRTM 540
C           FUNCTION, BIT, VOL. 3 (1963), PP.205-206.                   DRTM 550
C                                                                       DRTM 560
C     ..................................................................DRTM 570
C                                                                       DRTM 580
      SUBROUTINE DRTMI(X,F,FCT,XLI,XRI,EPS,IEND,IER)                    DRTM 590
C                                                                       DRTM 600
C                                                                       DRTM 610
      DOUBLE PRECISION X,F,FCT,XLI,XRI,XL,XR,FL,FR,TOL,TOLF,A,DX,XM,FM  DRTM 620
C                                                                       DRTM 630
C     PREPARE ITERATION                                                 DRTM 640
      IER=0                                                             DRTM 650
      XL=XLI                                                            DRTM 660
      XR=XRI                                                            DRTM 670
      X=XL                                                              DRTM 680
      TOL=X                                                             DRTM 690
      F=FCT(TOL)                                                        DRTM 700
      IF(F)1,16,1                                                       DRTM 710
    1 FL=F                                                              DRTM 720
      X=XR                                                              DRTM 730
      TOL=X                                                             DRTM 740
      F=FCT(TOL)                                                        DRTM 750
      IF(F)2,16,2                                                       DRTM 760
    2 FR=F                                                              DRTM 770
      IF(DSIGN(1.D0,FL)+DSIGN(1.D0,FR))25,3,25                          DRTM 780
C                                                                       DRTM 790
C     BASIC ASSUMPTION FL*FR LESS THAN 0 IS SATISFIED.                  DRTM 800
C     GENERATE TOLERANCE FOR FUNCTION VALUES.                           DRTM 810
    3 I=0                                                               DRTM 820
      TOLF=100.*EPS                                                     DRTM 830
C                                                                       DRTM 840
C                                                                       DRTM 850
C     START ITERATION LOOP                                              DRTM 860
    4 I=I+1                                                             DRTM 870
C                                                                       DRTM 880
C     START BISECTION LOOP                                              DRTM 890
      DO 13 K=1,IEND                                                    DRTM 900
      X=.5D0*(XL+XR)                                                    DRTM 910
      TOL=X                                                             DRTM 920
      F=FCT(TOL)                                                        DRTM 930
      IF(F)5,16,5                                                       DRTM 940
    5 IF(DSIGN(1.D0,F)+DSIGN(1.D0,FR))7,6,7                             DRTM 950
C                                                                       DRTM 960
C     INTERCHANGE XL AND XR IN ORDER TO GET THE SAME SIGN IN F AND FR   DRTM 970
    6 TOL=XL                                                            DRTM 980
      XL=XR                                                             DRTM 990
      XR=TOL                                                            DRTM1000
      TOL=FL                                                            DRTM1010
      FL=FR                                                             DRTM1020
      FR=TOL                                                            DRTM1030
    7 TOL=F-FL                                                          DRTM1040
      A=F*TOL                                                           DRTM1050
      A=A+A                                                             DRTM1060
      IF(A-FR*(FR-FL))8,9,9                                             DRTM1070
    8 IF(I-IEND)17,17,9                                                 DRTM1080
    9 XR=X                                                              DRTM1090
      FR=F                                                              DRTM1100
C                                                                       DRTM1110
C     TEST ON SATISFACTORY ACCURACY IN BISECTION LOOP                   DRTM1120
      TOL=EPS                                                           DRTM1130
      A=DABS(XR)                                                        DRTM1140
      IF(A-1.D0)11,11,10                                                DRTM1150
   10 TOL=TOL*A                                                         DRTM1160
   11 IF(DABS(XR-XL)-TOL)12,12,13                                       DRTM1170
   12 IF(DABS(FR-FL)-TOLF)14,14,13                                      DRTM1180
   13 CONTINUE                                                          DRTM1190
C     END OF BISECTION LOOP                                             DRTM1200
C                                                                       DRTM1210
C     NO CONVERGENCE AFTER IEND ITERATION STEPS FOLLOWED BY IEND        DRTM1220
C     SUCCESSIVE STEPS OF BISECTION OR STEADILY INCREASING FUNCTION     DRTM1230
C     VALUES AT RIGHT BOUNDS. ERROR RETURN.                             DRTM1240
      IER=1                                                             DRTM1250
   14 IF(DABS(FR)-DABS(FL))16,16,15                                     DRTM1260
   15 X=XL                                                              DRTM1270
      F=FL                                                              DRTM1280
   16 RETURN                                                            DRTM1290
C                                                                       DRTM1300
C     COMPUTATION OF ITERATED X-VALUE BY INVERSE PARABOLIC INTERPOLATIONDRTM1310
   17 A=FR-F                                                            DRTM1320
      DX=(X-XL)*FL*(1.D0+F*(A-TOL)/(A*(FR-FL)))/TOL                     DRTM1330
      XM=X                                                              DRTM1340
      FM=F                                                              DRTM1350
      X=XL-DX                                                           DRTM1360
      TOL=X                                                             DRTM1370
      F=FCT(TOL)                                                        DRTM1380
      IF(F)18,16,18                                                     DRTM1390
C                                                                       DRTM1400
C     TEST ON SATISFACTORY ACCURACY IN ITERATION LOOP                   DRTM1410
   18 TOL=EPS                                                           DRTM1420
      A=DABS(X)                                                         DRTM1430
      IF(A-1.D0)20,20,19                                                DRTM1440
   19 TOL=TOL*A                                                         DRTM1450
   20 IF(DABS(DX)-TOL)21,21,22                                          DRTM1460
   21 IF(DABS(F)-TOLF)16,16,22                                          DRTM1470
C                                                                       DRTM1480
C     PREPARATION OF NEXT BISECTION LOOP                                DRTM1490
   22 IF(DSIGN(1.D0,F)+DSIGN(1.D0,FL))24,23,24                          DRTM1500
   23 XR=X                                                              DRTM1510
      FR=F                                                              DRTM1520
      GO TO 4                                                           DRTM1530
   24 XL=X                                                              DRTM1540
      FL=F                                                              DRTM1550
      XR=XM                                                             DRTM1560
      FR=FM                                                             DRTM1570
      GO TO 4                                                           DRTM1580
C     END OF ITERATION LOOP                                             DRTM1590
C                                                                       DRTM1600
C                                                                       DRTM1610
C     ERROR RETURN IN CASE OF WRONG INPUT DATA                          DRTM1620
   25 IER=2                                                             DRTM1630
      RETURN                                                            DRTM1640
      END                                                               DRTM1650
```

Subroutines RTNI and DRTNI

These subroutines refine the initial guess x_0 of a root of the general nonlinear equation $f(x) = 0$. Newton's iteration scheme is used in the following form:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \qquad (i = 0, 1, 2, \ldots) \tag{1}$$

Convergence is quadratic or linear if the multiplicity of the root to be determined is equal to one or greater than one respectively, and if f(x) can be differentiated continuously at least twice in the range in which iteration moves. Each iteration step requires one evaluation of f(x) and one evaluation of $f'(x)$.

This iterative procedure is terminated if the following two conditions are satisfied:

$$\delta \leqq \varepsilon \text{ and } |f(x_{i+1})| \leqq 100 \cdot \varepsilon$$

with

$$\delta = \begin{cases} \left|\dfrac{x_{i+1} - x_i}{x_{i+1}}\right| & \text{in case of } |x_{i+1}| > 1 \\ |x_{i+1} - x_i| & \text{in case of } |x_{i+1}| \leqq 1 \end{cases} \tag{2}$$

and tolerance ε given by input.

The procedure described above may not converge within a specified number of iteration steps. Reasons for this behaviour, which is indicated by an error message, may be:

1. Too few iteration steps are specified.
2. The initial guess x_0 is too far away from any root.
3. The tolerance ε is too small with respect to roundoff errors.
4. The root to be determined is of multiplicity greater than one.

Furthermore, the procedure fails and is bypassed if at any iteration step the derivative $f(x_i)$ becomes zero. This is also indicated by an error message.

For reference see:

(1) F. B. Hildebrand, Introduction to Numerical Analysis, McGraw-Hill, New York/Toronto/London, 1956, pp. 447 - 450.

(2) R. Zurmühl, Praktische Mathematik für Ingenieure und Physiker, Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 12 - 17.


```
C
C     ..................................................................
C
C        SUBROUTINE RTNI
C
C        PURPOSE
C           TO SOLVE GENERAL NONLINEAR EQUATIONS OF THE FORM F(X)=0
C           BY MEANS OF NEWTON-S ITERATION METHOD.
C
C        USAGE
C           CALL RTNI (X,F,DERF,FCT,XST,EPS,IEND,IER)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.
C
C        DESCRIPTION OF PARAMETERS
C           X      - RESULTANT ROOT OF EQUATION F(X)=0.
C           F      - RESULTANT FUNCTION VALUE AT ROOT X.
C           DERF   - RESULTANT VALUE OF DERIVATIVE AT ROOT X.
C           FCT    - NAME OF THE EXTERNAL SUBROUTINE USED. IT COMPUTES
C                    TO GIVEN ARGUMENT X FUNCTION VALUE F AND DERIVATIVE
C                    DERF. ITS PARAMETER LIST MUST BE X,F,DERF.
C           XST    - INPUT VALUE WHICH SPECIFIES THE INITIAL GUESS OF
C                    THE ROOT X.
C           EPS    - INPUT VALUE WHICH SPECIFIES THE UPPER BOUND OF THE
C                    ERROR OF RESULT X.
C           IEND   - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS
C                      IER=0 - NO ERROR,
C                      IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS,
C                      IER=2 - AT ANY ITERATION STEP DERIVATIVE DERF WAS
C                              EQUAL TO ZERO.
C
C        REMARKS
C           THE PROCEDURE IS BYPASSED AND GIVES THE ERROR MESSAGE IER=2
C           IF AT ANY ITERATION STEP DERIVATIVE OF F(X) IS EQUAL TO 0.
C           POSSIBLY THE PROCEDURE WOULD BE SUCCESSFUL IF IT IS STARTED
C           ONCE MORE WITH ANOTHER INITIAL GUESS XST.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL SUBROUTINE FCT(X,F,DERF) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           SOLUTION OF EQUATION F(X)=0 IS DONE BY MEANS OF NEWTON-S
C           ITERATION METHOD, WHICH STARTS AT THE INITIAL GUESS XST OF
C           A ROOT X. CONVERGENCE IS QUADRATIC IF THE DERIVATIVE OF
C           F(X) AT ROOT X IS NOT EQUAL TO ZERO. ONE ITERATION STEP
C           REQUIRES ONE EVALUATION OF F(X) AND ONE EVALUATION OF THE
C           DERIVATIVE OF F(X). FOR TEST ON SATISFACTORY ACCURACY SEE
C           FORMULAE (2) OF MATHEMATICAL DESCRIPTION.
C           FOR REFERENCE, SEE R. ZURMUEHL, PRAKTISCHE MATHEMATIK FUER
C           INGENIEURE UND PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/
C           HEIDELBERG, 1963, PP.12-17.
C
C     ..................................................................
C
      SUBROUTINE RTNI(X,F,DERF,FCT,XST,EPS,IEND,IER)
C
C
C     PREPARE ITERATION
      IER=0
      X=XST
      TOL=X
      CALL FCT(TOL,F,DERF)
      TOLF=100.*EPS
C
C
C     START ITERATION LOOP
      DO 6 I=1,IEND
      IF(F)1,7,1
C
C     EQUATION IS NOT SATISFIED BY X
    1 IF(DERF)2,8,2
C
C     ITERATION IS POSSIBLE
    2 DX=F/DERF
      X=X-DX
      TOL=X
      CALL FCT(TOL,F,DERF)
C
C     TEST ON SATISFACTORY ACCURACY
      TOL=EPS
      A=ABS(X)
      IF(A-1.)4,4,3
    3 TOL=TOL*A
    4 IF(ABS(DX)-TOL)5,5,6
    5 IF(ABS(F)-TOLF)7,7,6
    6 CONTINUE
C     END OF ITERATION LOOP
C
C
C
C     NO CONVERGENCE AFTER IEND ITERATION STEPS. ERROR RETURN.
      IER=1
    7 RETURN
C
C     ERROR RETURN IN CASE OF ZERO DIVISOR
    8 IER=2
      RETURN
      END
```

```
C                                                                       DRTN  10
C     ......................................................................DRTN  20
C                                                                       DRTN  30
C        SUBROUTINE DRTNI                                               DRTN  40
C                                                                       DRTN  50
C        PURPOSE                                                        DRTN  60
C           TO SOLVE GENERAL NONLINEAR EQUATIONS OF THE FORM F(X)=0     DRTN  70
C           BY MEANS OF NEWTON-S ITERATION METHOD.                      DRTN  80
C                                                                       DRTN  90
C        USAGE                                                          DRTN 100
C           CALL DRTNI (X,F,DERF,FCT,XST,EPS,IEND,IER)                  DRTN 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.               DRTN 120
C                                                                       DRTN 130
C        DESCRIPTION OF PARAMETERS                                      DRTN 140
C           X      - DOUBLE PRECISION RESULTANT ROOT OF EQUATION F(X)=0.DRTN 150
C           F      - DOUBLE PRECISION RESULTANT FUNCTION VALUE AT       DRTN 160
C                    ROOT X.                                            DRTN 170
C           DERF   - DOUBLE PRECISION RESULTANT VALUE OF DERIVATIVE     DRTN 180
C                    AT ROOT X.                                         DRTN 190
C           FCT    - NAME OF THE EXTERNAL SUBROUTINE USED. IT COMPUTES  DRTN 200
C                    TO GIVEN ARGUMENT X FUNCTION VALUE F AND DERIVATIVEDRTN 210
C                    DERF. ITS PARAMETER LIST MUST BE X,F,DERF, WHERE   DRTN 220
C                    ALL PARAMETERS ARE DOUBLE PRECISION.               DRTN 230
C           XST    - DOUBLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTN 240
C                    INITIAL GUESS OF THE ROOT X.                       DRTN 250
C           EPS    - SINGLE PRECISION INPUT VALUE WHICH SPECIFIES THE   DRTN 260
C                    UPPER BOUND OF THE ERROR OF RESULT X.              DRTN 270
C           IEND   - MAXIMUM NUMBER OF ITERATION STEPS SPECIFIED.       DRTN 280
C           IER    - RESULTANT ERROR PARAMETER CODED AS FOLLOWS         DRTN 290
C                      IER=0 - NO ERROR,                                DRTN 300
C                      IER=1 - NO CONVERGENCE AFTER IEND ITERATION STEPS,DRTN 310
C                      IER=2 - AT ANY ITERATION STEP DERIVATIVE DERF WAS DRTN 320
C                              EQUAL TO ZERO.                           DRTN 330
C                                                                       DRTN 340
C        REMARKS                                                        DRTN 350
C           THE PROCEDURE IS BYPASSED AND GIVES THE ERROR MESSAGE IER=2 DRTN 360
C           IF AT ANY ITERATION STEP DERIVATIVE OF F(X) IS EQUAL TO 0.  DRTN 370
C           POSSIBLY THE PROCEDURE WOULD BE SUCCESSFUL IF IT IS STARTED DRTN 380
C           ONCE MORE WITH ANOTHER INITIAL GUESS XST.                   DRTN 390
C                                                                       DRTN 400
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DRTN 410
C           THE EXTERNAL SUBROUTINE FCT(X,F,DERF) MUST BE FURNISHED     DRTN 420
C           BY THE USER.                                                DRTN 430
C                                                                       DRTN 440
C                                                                       DRTN 450
C        METHOD                                                         DRTN 460
C           SOLUTION OF EQUATION F(X)=0 IS DONE BY MEANS OF NEWTON-S    DRTN 470
C           ITERATION METHOD, WHICH STARTS AT THE INITIAL GUESS XST OF  DRTN 480
C           A ROOT X. CONVERGENCE IS QUADRATIC IF THE DERIVATIVE OF     DRTN 490
C           F(X) AT ROOT X IS NOT EQUAL TO ZERO. ONE ITERATION STEP     DRTN 500
C           REQUIRES ONE EVALUATION OF F(X) AND ONE EVALUATION OF THE   DRTN 510
C           DERIVATIVE OF F(X). FOR TEST ON SATISFACTORY ACCURACY SEE   DRTN 520
C           FORMULAE (2) OF MATHEMATICAL DESCRIPTION.                   DRTN 530
C           FOR REFERENCE, SEE R. ZURMUEHL, PRAKTISCHE MATHEMATIK FUER  DRTN 540
C           INGENIEURE UND PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/       DRTN 550
C           HEIDELBERG, 1963, PP.12-17.                                 DRTN 560
C                                                                       DRTN 570
C     ......................................................................DRTN 580
C                                                                       DRTN 590
      SUBROUTINE DRTNI(X,F,DERF,FCT,XST,EPS,IEND,IER)                   DRTN 600
C                                                                       DRTN 610
C                                                                       DRTN 620
      DOUBLE PRECISION X,F,DERF,XST,TOL,TOLF,DX,A                       DRTN 630
C                                                                       DRTN 640
C     PREPARE ITERATION                                                 DRTN 650
      IER=0                                                             DRTN 660
      X=XST                                                             DRTN 670
      TOL=X                                                             DRTN 680
      CALL FCT(TOL,F,DERF)                                              DRTN 690
      TOLF=100.*EPS                                                     DRTN 700
C                                                                       DRTN 710
C                                                                       DRTN 720
C     START ITERATION LOOP                                              DRTN 730
      DO 6 I=1,IEND                                                     DRTN 740
      IF(F)1,7,1                                                        DRTN 750
C                                                                       DRTN 760
C     EQUATION IS NOT SATISFIED BY X                                    DRTN 770
    1 IF(DERF)2,8,2                                                     DRTN 780
C                                                                       DRTN 790
C     ITERATION IS POSSIBLE                                             DRTN 800
    2 DX=F/DERF                                                         DRTN 810
      X=X-DX                                                            DRTN 820
      TOL=X                                                             DRTN 830
      CALL FCT(TOL,F,DERF)                                              DRTN 840
C                                                                       DRTN 850
C     TEST ON SATISFACTORY ACCURACY                                     DRTN 860
      TOL=EPS                                                           DRTN 870
      A=DABS(X)                                                         DRTN 880
      IF(A-1.D0)4,4,3                                                   DRTN 890
    3 TOL=TOL*A                                                         DRTN 900
    4 IF(DABS(DX)-TOL)5,5,6                                             DRTN 910
    5 IF(DABS(F)-TOLF)7,7,6                                             DRTN 920
    6 CONTINUE                                                          DRTN 930
C     END OF ITERATION LOOP                                             DRTN 940
C                                                                       DRTN 950
C                                                                       DRTN 960
C     NO CONVERGENCE AFTER IEND ITERATION STEPS. ERROR RETURN.          DRTN 970
      IER=1                                                             DRTN 980
    7 RETURN                                                            DRTN 990
C                                                                       DRTN1000
C     ERROR RETURN IN CASE OF ZERO DIVISOR                              DRTN1010
    8 IER=2                                                             DRTN1020
      RETURN                                                            DRTN1030
      END
```

Extremum of Functions

Subroutines FMFP and DFMFP

These subroutines perform the calculation of an unconstrained minimum of a function of several variables using a method proposed by Davidon. The underlying method is described in the article by R. Fletcher and M.J.D. Powell, "A Rapidly Convergent Descent Method for Minimization", Computer Journal, vol. 6, iss. 2, 1963, pp. 163 - 168.

It is assumed that the function f of the n variables $x_1, \ldots, x_n$ (abbreviated as argument vector x) may be computed together with its gradient vector g(x) for any point x. The generalized Taylor expansion for functions of several variables is

$$f(x+u) = f(x) + g(x) \cdot u + \frac{1}{2} u^T G(x) u + \text{higher terms}$$

where g is the gradient vector and G the matrix of second order partial derivatives. Vectors are assumed to be column vectors; u^T means transpose of vector u. It is assumed that in the neighborhood of the required minimum x_{min} the function is approximated closely by the first three terms of its Taylor expansion, giving

$$f(x) = f(x_{min}) + \frac{1}{2} (x - x_{min})^T G(x_{min}) (x - x_{min})$$

since $g(x_{min}) = 0$. Then the gradient is seen to be approximately $g(x) = G(x_{min}) (x - x_{min})$.

Assume now that the symmetric matrix G is positive definite. Then the following equation holds true:

$$x - x_{min} = G^{-1} (x_{min}) \cdot g(x)$$

which would allow x_{min} to be calculated in one step if $G^{-1} (x_{min})$ were available.

To approach $G^{-1} (x_{min})$, a method of successive linear searches in G-conjugate directions is used. Starting with the identity matrix $G^{(0)} = I$, a sequence of symmetric matrices $G^{(i)}$ is generated which tends to G^{-1}. At the $(i+1)^{st}$ iteration step a linear search is made in direction $h^{(i)} = -G^{(i)} g^{(i)}$, where $g^{(i)}$ is an abbreviation for $g(x^{(i)})$. By means of the linear search the minimum of $y(t) = f(x^{(i)} + t \cdot h^{(i)})$ is determined, giving argument $x^{(i+1)} = x^{(i)} + t_i \cdot h^{(i)}$.

The argument of the minimum $x^{(i+1)}$ on the line through $x^{(i)}$ in direction $h^{(i)}$ is determined by the relation that scalar product $(g^{(i+1)}, h^{(i)}) = 0$.

Now: $$x^{(n)} = x^{(j)} + \sum_{i=j}^{n-1} t_i h^{(i)}$$

and: $$g^{(n)} = g^{(j)} + \sum_{i=j}^{n-1} t_i G h^{(i)}$$

Therefore:

scalar product $$(g^{(n)}, h^{(j)}) = \sum_{i=j+1}^{n-1} t_i (G h^{(i)}, h^{(j)})$$

Suppose now that the vectors $h^{(0)}, h^{(1)}, \ldots, h^{(n-1)}$ are G-conjugate, satisfying $(G h^{(i)}, h^{(j)}) = 0$ for $i \neq j$. Then $(g^{(n)}, h^{(j)}) = 0$, and since $h^{(0)}, h^{(1)}, \ldots, h^{(n-1)}$ form a basis, $g^{(n)} = 0$ and $x^{(n)} = x_{min}$. This shows that the minimum is located at the nth iteration for a quadratic function when using successive linear searches for G-conjugate directions.

For the generation of G-conjugate directions, start with $h^{(0)} = -g^{(0)}$ and calculate successive directions $h^{(i)}$ by means of $h^{(i)} = -G^{(i)}g^{(i)}$, where $G^{(i)}$ is modified to $G^{(i+1)}$ so that $h^{(i)}$ is an eigenvector of the matrix $G^{(i+1)}$ G with eigenvalue 1. This ensures that $G^{(i)}$ approaches G^{-1} as $x^{(i)}$ approaches x_{min}. An easy calculation shows:

$$G^{(i+1)} = G^{(i)} + \frac{dx \cdot dx^T}{dx^T \cdot dg} - \frac{G^{(i)} dg \cdot dg^T G^{(i)}}{dg^T G^{(i)} dg}$$

with $dg = g^{(i+1)} - g^{(i)}$

$dx = x^{(i+1)} - x^{(i)}$

where all vectors are regarded as column vectors, and superscript T means transpose of column vector--that is, row vector.

The strategy adopted for termination of the successive linear searches is as follows:

1. If the function value has not decreased in the last iteration step, the search for the minimum is terminated provided the gradient is already sufficiently small; otherwise, the next step is in the direction of steepest descent.
2. If the argument vector and the direction vector change by very small amounts, and at least n iterations are performed, the minimization is terminated again.
3. If the number of iterations exceeds an upper bound furnished by the user, further calculation is bypassed, and an error code is set to 1 indicating poor convergence.
4. If one of the successive linear searches indicates that no constrained minimum exists, further calculation is bypassed again, and the error code is set to 2 indicating that it is likely that no minimum exists.

The ith term $G^{(i)}$ is reset to the identity matrix if there is indication that the current $G^{(i)}$ is not positive definite, or if the formula for $G^{(i+1)}$ breaks down due to zero divisors.

The linear search technique used in subroutines FMFP and DFMFP is as follows. For a given argument vector x and vector h, defining a direction through x, a local minimum of the function $y(t) = f(x+th)$ must be found. This means that a value t_m must be determined for which

$$y'(t_m) = \text{scalar product } (g(x+t_m h), h) = 0$$

From $y'(0) = (g(x), h) < 0$ it is evident that a minimum $y(t_m) < y(0)$ should be found for positive values of t.

The calculation of the minimum is in three stages. The first estimates the magnitude of t_m, the second determines an interval containing t_m, and the third interpolates the value of t_m.

An estimate of the step-size may be obtained, assuming that the true value of the constrained minimum is equal to the estimated value EST of the unconstrained minimum and that y (t) is closely represented by a quadratic polynomial passing through (x, y(0)) with derivative y'(0):
step = 2 (EST - y(0))/y'(0).

This equation tends to overestimate the step-size, since the unconstrained minimum will normally not lie on the line through x with direction h. Therefore, step is taken as step-size s only if it is positive and less than one. Otherwise, s = 1 is taken as step-size.

At the second stage y(t) and y'(t) are examined at the points t = s, 2s, 4s,..., s_1, s_2, where successive values are obtained by doubling the step-size.

This search is terminated at $t = s_2$ if:

$$y'(s_2) = 0, \text{ or } y'(s_2) > 0, \text{ or } y(s_2) \geqq y(s_1)$$

$$\text{or if } s_2 \cdot \left(\sum_{i=1}^{n} |h_i| \right) > 10^{10}.$$

The last case (search argument runs out of range) is interpreted as an indication that no local minimum exists on the given line. In this case, the error indicator is set to (2) and further calculation is bypassed.

In case $y'(s_2) = 0$, t_m is set to s_2 and $x_m = x = s_2 h$ is used as the argument of a constrained minimum on the line through x with direction h.

In the second and third case ($y'(s_2) > 0$ and/or $y(s_2) \geq y(s_1)$ a minimum lies necessarily between s_1 and s_2. Its argument value gets approximated using cubic interpolation.

The extrema of the cubic interpolation passing through (s_1, $y_1 = y(s_1)$, $y'_1 = y'(s_1)$) and (s_2, $y_2 = y(s_2)$, $y'_2 = y'(s_2)$) are given by solving a quadratic equation. The solution s_3 can be expressed as $s_3 = s_1 + (1-\alpha)(s_2 - s_1) = s_2 - \alpha(s_2 - s_1)$ with

$$\alpha = \frac{y'_2 - z + w}{y'_2 - y'_1 + 2W} = \frac{y'_2 + z - w}{y'_2 + y'_1 + 2z},$$

$$0 < \alpha < 1, \text{ and } z = y'_1 + y'_2 - 3\,\frac{y_2 - y_2}{s_2 - s_1},$$

$$w = \pm (z^2 - y'_1 y'_2)^{1/2}.$$

The conditions on the problem guarantee a real value of w. The sign of w is so chosen that s_3 is in the interval (s_1, s_2). If this condition is satisfied for the two possible values of s_3, the value closer to s_1 is retained.

As written above, two equivalent formulas can provide α. The choice is based on numerical stability considerations.

If $y(s_3) \leq y(s_1)$ and $y(s_3) \leq y(s_2)$, then t_m is set equal to s_3 and $x_m = x + t_m$. h is used as argument of the desired minimum along the given line. Otherwise, the interval (s_1, s_2) is reduced by replacing s_1 by s_3 if $y(s_3) \leq y(s_1)$ and $y'(s_3) < 0$ and by replacing s_2 by s_3 in all other cases. The interpolation process is repeated for this new reduced interval.

```
C                                                                       FMFP  10
C     ..................................................................FMFP  20
C                                                                       FMFP  30
C        SUBROUTINE FMFP                                                FMFP  40
C                                                                       FMFP  50
C        PURPOSE                                                        FMFP  60
C           TO FIND A LOCAL MINIMUM OF A FUNCTION OF SEVERAL VARIABLES  FMFP  70
C           BY THE METHOD OF FLETCHER AND POWELL                        FMFP  80
C                                                                       FMFP  90
C        USAGE                                                          FMFP 100
C           CALL FMFP(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)                FMFP 110
C                                                                       FMFP 120
C        DESCRIPTION OF PARAMETERS                                      FMFP 130
C           FUNCT  - USER-WRITTEN SUBROUTINE CONCERNING THE FUNCTION TO FMFP 140
C                    BE MINIMIZED. IT MUST BE OF THE FORM               FMFP 150
C                    SUBROUTINE FUNCT(N,ARG,VAL,GRAD)                   FMFP 160
C                    AND MUST SERVE THE FOLLOWING PURPOSE               FMFP 170
C                    FOR EACH N-DIMENSIONAL ARGUMENT VECTOR  ARG,       FMFP 180
C                    FUNCTION VALUE AND GRADIENT VECTOR MUST BE COMPUTEDFMFP 190
C                    AND, ON RETURN, STORED IN VAL AND GRAD RESPECTIVELYFMFP 200
C           N      - NUMBER OF VARIABLES                               FMFP 210
C           X      - VECTOR OF DIMENSION N CONTAINING THE INITIAL      FMFP 220
C                    ARGUMENT WHERE THE ITERATION STARTS. ON RETURN,   FMFP 230
C                    X HOLDS THE ARGUMENT CORRESPONDING TO THE         FMFP 240
C                    COMPUTED MINIMUM FUNCTION VALUE                   FMFP 250
C           F      - SINGLE VARIABLE CONTAINING THE MINIMUM FUNCTION   FMFP 260
C                    VALUE ON RETURN, I.E. F=F(X).                     FMFP 270
C           G      - VECTOR OF DIMENSION N CONTAINING THE GRADIENT     FMFP 280
C                    VECTOR CORRESPONDING TO THE MINIMUM ON RETURN,    FMFP 290
C                    I.E. G=G(X).                                      FMFP 300
C           EST    - IS AN ESTIMATE OF THE MINIMUM FUNCTION VALUE.     FMFP 310
C           EPS    - TESTVALUE REPRESENTING THE EXPECTED ABSOLUTE ERROR.FMFP 320
C                    A REASONABLE CHOICE IS 10**(-6), I.E.             FMFP 330
C                    SOMEWHAT GREATER THAN 10**(-D), WHERE D IS THE    FMFP 340
C                    NUMBER OF SIGNIFICANT DIGITS IN FLOATING POINT    FMFP 350
C                    REPRESENTATION.                                   FMFP 360
C           LIMIT  - MAXIMUM NUMBER OF ITERATIONS.                     FMFP 370
C           IER    - ERROR PARAMETER                                   FMFP 380
C                    IER = 0 MEANS CONVERGENCE WAS OBTAINED            FMFP 390
C                    IER = 1 MEANS NO CONVERGENCE IN LIMIT ITERATIONS  FMFP 400
C                    IER =-1 MEANS ERRORS IN GRADIENT CALCULATION      FMFP 410
C                    IER = 2 MEANS LINEAR SEARCH TECHNIQUE INDICATES   FMFP 420
C                    IT IS LIKELY THAT THERE EXISTS NO MINIMUM.        FMFP 430
C           H      - WORKING STORAGE OF DIMENSION N*(N+7)/2.           FMFP 440
C                                                                       FMFP 450
C        REMARKS                                                        FMFP 460
C           1) THE SUBROUTINE NAME REPLACING THE DUMMY ARGUMENT  FUNCT FMFP 470
C              MUST BE DECLARED AS EXTERNAL IN THE CALLING PROGRAM.     FMFP 480
C           11) IER IS SET TO 2 IF , STEPPING IN ONE OF THE COMPUTED    FMFP 490
C               DIRECTIONS, THE FUNCTION WILL NEVER INCREASE WITHIN     FMFP 500
C               A TOLERABLE RANGE OF ARGUMENT.                          FMFP 510
C               IER = 2 MAY OCCUR ALSO IF THE INTERVAL WHERE F          FMFP 520
C               INCREASES IS SMALL AND THE INITIAL ARGUMENT WAS         FMFP 530
C               RELATIVELY FAR AWAY FROM THE MINIMUM SUCH THAT THE      FMFP 540
C               MINIMUM WAS OVERLEAPED. THIS IS DUE TO THE SEARCH       FMFP 550
C               TECHNIQUE WHICH DOUBLES THE STEPSIZE UNTIL A POINT      FMFP 560
C               IS FOUND WHERE THE FUNCTION INCREASES.                  FMFP 570
C                                                                       FMFP 580
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  FMFP 590
C           FUNCT                                                       FMFP 600
C                                                                       FMFP 610
C        METHOD                                                         FMFP 620
C           THE METHOD IS DESCRIBED IN THE FOLLOWING ARTICLE            FMFP 630
C           R. FLETCHER AND M.J.D. POWELL, A RAPID DESCENT METHOD FOR   FMFP 640
C           MINIMIZATION,                                               FMFP 650
C           COMPUTER JOURNAL VOL.6, ISS. 2, 1963, PP.163-168.           FMFP 660
C                                                                       FMFP 670
C     ..................................................................FMFP 680
C                                                                       FMFP 690
      SUBROUTINE FMFP(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)                FMFP 700
C                                                                       FMFP 710
C        DIMENSIONED DUMMY VARIABLES                                    FMFP 720
      DIMENSION H(1),X(1),G(1)                                          FMFP 730
C                                                                       FMFP 740
C        COMPUTE FUNCTION VALUE AND GRADIENT VECTOR FOR INITIAL ARGUMENTFMFP 750
      CALL FUNCT(N,X,F,G)                                               FMFP 760
C                                                                       FMFP 770
C        RESET ITERATION COUNTER AND GENERATE IDENTITY MATRIX           FMFP 780
      IER=0                                                             FMFP 790
      KCLAT=0                                                           FMFP 800
      N2=N+N                                                            FMFP 810
      N3=N2+N                                                           FMFP 820
      N31=N3+1                                                          FMFP 830
    1 K=N31                                                             FMFP 840
      DO 4 J=1,N                                                        FMFP 850
      H(K)=1.                                                           FMFP 860
      NJ=N-J                                                            FMFP 870
      IF(NJ)5,5,2                                                       FMFP 880
    2 DO 3 L=1,NJ                                                       FMFP 890
      KL=K+L                                                            FMFP 900
    3 H(KL)=0.                                                          FMFP 910
    4 K=KL+1                                                            FMFP 920
C                                                                       FMFP 930
C        START ITERATION LOOP                                           FMFP 940
    5 KCLAT=KCLAT +1                                                    FMFP 950
C                                                                       FMFP 960
C        SAVE FUNCTION VALUE, ARGUMENT VECTOR AND GRADIENT VECTOR       FMFP 970
      OLDF=F                                                            FMFP 980
      DO 9 J=1,N                                                        FMFP 990
      K=N+J                                                             FMFP1000
      H(K)=G(J)                                                         FMFP1010
      K=K+N                                                             FMFP1020
      H(K)=X(J)                                                         FMFP1030
C                                                                       FMFP1040
C        DETERMINE DIRECTION VECTOR H                                   FMFP1050
      K=J+N3                                                            FMFP1060
      T=0.                                                              FMFP1070
      DO 8 L=1,N                                                        FMFP1080
      T=T-G(L)*H(K)                                                     FMFP1090
      IF(L-J)6,7,7                                                      FMFP1100
    6 K=K+N-L                                                           FMFP1110
      GO TO 8                                                           FMFP1120
    7 K=K+1                                                             FMFP1130
    8 CONTINUE                                                          FMFP1140
    9 H(J)=T                                                            FMFP1150
C                                                                       FMFP1160
C        CHECK WHETHER FUNCTION WILL DECREASE STEPPING ALONG H.         FMFP1170
      DY=0.                                                             FMFP1180
      HNRM=0.                                                           FMFP1190
      GNRM=0.                                                           FMFP1200
C                                                                       FMFP1210
C        CALCULATE DIRECTIONAL DERIVATIVE AND TESTVALUES FOR DIRECTION  FMFP1220
C        VECTOR H AND GRADIENT VECTOR G.                                FMFP1230
      DO 10 J=1,N                                                       FMFP1240
      HNRM=HNRM+ABS(H(J))                                               FMFP1250
      GNRM=GNRM+ABS(G(J))                                               FMFP1260
   10 DY=DY+H(J)*G(J)                                                   FMFP1270
C                                                                       FMFP1280
C        REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF DIRECTIONAL  FMFP1290
C        DERIVATIVE APPEARS TO BE POSITIVE OR ZERO.                     FMFP1300
      IF(DY)11,51,51                                                    FMFP1310
C                                                                       FMFP1320
C        REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF DIRECTION    FMFP1330
C        VECTOR H IS SMALL COMPARED TO GRADIENT VECTOR G.               FMFP1340
   11 IF(HNRM/GNRM-EPS)51,51,12                                         FMFP1350
C                                                                       FMFP1360
C        SEARCH MINIMUM ALONG DIRECTION H                               FMFP1370
C                                                                       FMFP1380
C        SEARCH ALONG H FOR POSITIVE DIRECTIONAL DERIVATIVE             FMFP1390
   12 FY=F                                                              FMFP1400
      ALFA=2.*(EST-F)/DY                                                FMFP1410
      AMEGA=1.                                                          FMFP1420
C                                                                       FMFP1430
C        USE ESTIMATE FOR STEPSIZE ONLY IF IT IS POSITIVE AND LESS THAN FMFP1440
C        1. OTHERWISE TAKE 1. AS STEPSIZE                               FMFP1450
      IF(ALFA)15,15,13                                                  FMFP1460
   13 IF(ALFA-AMEGA)14,15,15                                            FMFP1470
   14 AMEGA=ALFA                                                        FMFP1480
   15 ALFA=0.                                                           FMFP1490
C                                                                       FMFP1500
C        SAVE FUNCTION AND DERIVATIVE VALUES FOR OLD ARGUMENT           FMFP1510
   16 FX=FY                                                             FMFP1520
      DX=DY                                                             FMFP1530
C                                                                       FMFP1540
C        STEP ARGUMENT ALONG H                                          FMFP1550
      DO 17 I=1,N                                                       FMFP1560
   17 X(I)=X(I)+AMEGA*H(I)                                              FMFP1570
C                                                                       FMFP1580
C        COMPUTE FUNCTION VALUE AND GRADIENT FOR NEW ARGUMENT           FMFP1590
      CALL FUNCT(N,X,F,G)                                               FMFP1600
      FY=F                                                              FMFP1610
C                                                                       FMFP1620
C        COMPUTE DIRECTIONAL DERIVATIVE DY FOR NEW ARGUMENT.  TERMINATE FMFP1630
C        SEARCH, IF DY IS POSITIVE. IF DY IS ZERO THE MINIMUM IS FOUND  FMFP1640
      DY=0.                                                             FMFP1650
      DO 18 I=1,N                                                       FMFP1660
   18 DY=DY+G(I)*H(I)                                                   FMFP1670
      IF(DY)19,36,22                                                    FMFP1680
C                                                                       FMFP1690
C        TERMINATE SEARCH ALSO IF THE FUNCTION VALUE INDICATES THAT     FMFP1700
C        A MINIMUM HAS BEEN PASSED                                      FMFP1710
   19 IF(FY-FX)20,22,22                                                 FMFP1720
C                                                                       FMFP1730
C        REPEAT SEARCH AND DOUBLE STEPSIZE FOR FURTHER SEARCHES         FMFP1740
   20 AMEGA=AMEGA+ALFA                                                  FMFP1750
      ALFA=AMEGA                                                        FMFP1760
C        END OF SEARCH LOOP                                             FMFP1770
C                                                                       FMFP1780
C        TERMINATE IF THE CHANGE IN ARGUMENT GETS VERY LARGE            FMFP1790
      IF(HNRM*AMEGA-1.E10)16,16,21                                      FMFP1800
C                                                                       FMFP1810
C        LINEAR SEARCH TECHNIQUE INDICATES THAT NO MINIMUM EXISTS       FMFP1820
   21 IER=2                                                             FMFP1830
      RETURN                                                            FMFP1840
C                                                                       FMFP1850
C        INTERPOLATE QUADRATICALLY IN THE INTERVAL DEFINED BY THE SEARCH FMFP1860
C        ABOVE AND COMPUTE THE ARGUMENT X FOR WHICH THE INTERPOLATION   FMFP1870
C        POLYNOMIAL IS MINIMIZED                                        FMFP1880
   22 T=0.                                                              FMFP1890
   23 IF(AMEGA)24,36,24                                                 FMFP1900
   24 Z=3.*(FX-FY)/AMEGA+DX+DY                                          FMFP1910
      ALFA=AMAX1(ABS(Z),ABS(DX),ABS(DY))                                FMFP1920
      ALFA=Z/ALFA                                                       FMFP1930
      ALFA=ALFA*ALFA-DX/ALFA*DY/ALFA                                    FMFP1940
      IF(ALFA)51,25,25                                                  FMFP1950
   25 W=ALFA*SQRT(ALFA)                                                 FMFP1960
      ALFA=DY-DX+W+W                                                    FMFP1970
      IF(ALFA) 250,251,250                                              FMFP1971
  250 ALFA=(DY-Z+W)/ALFA                                                FMFP1972
      GO TO 252                                                         FMFP1973
  251 ALFA=(Z+DY-W)/(Z+DX+Z+DY)                                         FMFP1974
  252 ALFA=ALFA*AMEGA                                                   FMFP1975
      DO 26 I=1,N                                                       FMFP1980
   26 X(I)=X(I)+(T-ALFA)*H(I)                                           FMFP1990
C                                                                       FMFP2000
C        TERMINATE, IF THE VALUE OF THE ACTUAL FUNCTION AT X IS LESS    FMFP2010
C        THAN THE FUNCTION VALUES AT THE INTERVAL ENDS. OTHERWISE REDUCEFMFP2020
C        THE INTERVAL BY CHOOSING ONE END-POINT EQUAL TO X AND REPEAT   FMFP2030
C        THE INTERPOLATION. WHICH END-POINT IS CHOSEN DEPENDS ON THE    FMFP2040
C        VALUE OF THE FUNCTION AND ITS GRADIENT AT X                    FMFP2050
C                                                                       FMFP2060
      CALL FUNCT(N,X,F,G)                                               FMFP2070
      IF(F-FX)27,27,28                                                  FMFP2080
   27 IF(F-FY)36,36,28                                                  FMFP2090
   28 ALFA=0.                                                           FMFP2100
      DO 29 I=1,N                                                       FMFP2110
   29 ALFA=ALFA+G(I)*H(I)                                               FMFP2120
      IF(ALFA)30,33,33                                                  FMFP2130
   30 IF(F-FX)32,31,33                                                  FMFP2140
   31 IF(DX-ALFA)32,36,32                                               FMFP2150
   32 FX=F                                                              FMFP2160
      DX=ALFA                                                           FMFP2170
      T=ALFA                                                            FMFP2180
      AMEGA=ALFA                                                        FMFP2190
      GO TO 23                                                          FMFP2200
   33 IF(FY-F)35,34,35                                                  FMFP2210
   34 IF(DY-ALFA)35,36,35                                               FMFP2220
   35 FY=F                                                              FMFP2230
      DY=ALFA                                                           FMFP2240
      AMEGA=AMEGA-ALFA                                                  FMFP2250
      GO TO 22                                                          FMFP2260
C                                                                       FMFP2270
C        TERMINATE, IF FUNCTION HAS NOT DECREASED DURING LAST ITERATION FMFP2280
   36 IF(OLDF-F+EPS)51,28,38                                            FMFP2290
C                                                                       FMFP2300
C        COMPUTE DIFFERENCE VECTORS OF ARGUMENT AND GRADIENT FROM       FMFP2310
C        TWO CONSECUTIVE ITERATIONS                                     FMFP2320
   38 DO 37 J=1,N                                                       FMFP2330
      K=N+J                                                             FMFP2340
      H(K)=G(J)-H(K)                                                    FMFP2350
      K=N+K                                                             FMFP2360
```

```
   37 H(K)=X(J)-H(K)                                                    FMFP2370
C                                                                       FMFP2380
C        TEST LENGTH OF ARGUMENT DIFFERENCE VECTOR AND DIRECTION VECTOR FMFP2390
C        IF AT LEAST N ITERATIONS HAVE BEEN EXECUTED. TERMINATE, IF     FMFP2400
C        BOTH ARE LESS THAN  EPS                                        FMFP2410
      IER=0                                                             FMFP2420
      IF(KCLNT-N)42,39,39                                               FMFP2430
   39 T=0.                                                              FMFP2440
      Z=0.                                                              FMFP2450
      DO 40 J=1,N                                                       FMFP2460
      K=N+J                                                             FMFP2470
      W=H(K)                                                            FMFP2480
      K=K+N                                                             FMFP2490
      T=T+ZES(H(K))                                                     FMFP2500
   40 Z=Z+W*H(K)                                                        FMFP2510
      IF(T-ARP-EPS)41,41,42                                             FMFP2520
   41 IF(T-EPS)56,56,42                                                 FMFP2530
C                                                                       FMFP2540
C        TERMINATE, IF NUMBER OF ITERATIONS WOULD EXCEED  LIMIT         FMFP2550
   42 IF(KCLNT-LIMIT)43,50,50                                           FMFP2560
C                                                                       FMFP2570
C        PREPARE UPDATING OF MATRIX H                                   FMFP2580
   43 ALFA=0.                                                           FMFP2590
      DO 47 J=1,N                                                       FMFP2600
      M=J+N3                                                            FMFP2610
      W=0.                                                              FMFP2620
      DO 46 L=1,N                                                       FMFP2630
      KL=N+L                                                            FMFP2640
      W=W+H(KL)*H(K)                                                    FMFP2650
      IF(L-J)44,45,45                                                   FMFP2660
   44 K=K+N-L                                                           FMFP2670
      GO TO 46                                                          FMFP2680
   45 K=K+1                                                             FMFP2690
   46 CONTINUE                                                          FMFP2700
      K=N+J                                                             FMFP2710
      ALFA=ALFA+W*H(K)                                                  FMFP2720
   47 H(J)=W                                                            FMFP2730
C                                                                       FMFP2740
C        REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF RESULTS      FMFP2750
C        ARE NOT SATISFACTORY                                           FMFP2760
      IF(Z*ALFA)48,1,48                                                 FMFP2770
C                                                                       FMFP2780
C        UPDATE MATRIX H                                                FMFP2790
   48 K=N31                                                             FMFP2800
      DO 49 L=1,N                                                       FMFP2810
      KL=N2+L                                                           FMFP2820
      DO 49 J=L,N                                                       FMFP2830
      NJ=N2+J                                                           FMFP2840
      H(K)=H(K)+H(KL)*H(NJ)/Z-H(L)*H(J)/ALFA                            FMFP2850
   49 K=K+1                                                             FMFP2860
      GO TO 5                                                           FMFP2870
C        END OF ITERATION LOOP                                          FMFP2880
C                                                                       FMFP2890
C        NO CONVERGENCE AFTER  LIMIT  ITERATIONS                        FMFP2900
   50 IER=1                                                             FMFP2910
      RETURN                                                            FMFP2920
C                                                                       FMFP2930
C        RESTORE OLD VALUES OF FUNCTION AND ARGUMENTS                   FMFP2940
   51 DO 52 J=1,N                                                       FMFP2950
      K=N2+J                                                            FMFP2960
   52 X(J)=H(K)                                                         FMFP2970
      CALL FUNCTN,X,F,G)                                                FMFP2980
C                                                                       FMFP2990
C        REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF DERIVATIVE   FMFP3000
C        FAILS TO BE SUFFICIENTLY SMALL                                 FMFP3010
      IF(GARP-EPS)55,55,53                                              FMFP3020
C                                                                       FMFP3030
C        TEST FOR REPEATED FAILURE OF ITERATION                         FMFP3040
   53 IF(IER)56,54,54                                                   FMFP3050
   54 IER=-1                                                            FMFP3060
      GOTO 1                                                            FMFP3070
   55 IER=0                                                             FMFP3080
   56 RETURN                                                            FMFP3090
      END                                                               FMFP3100
```

```
C                                                                       DFMF  10
C     ..................................................................DFMF  20
C                                                                       DFMF  30
C        SUBROUTINE DFMFP                                               DFMF  40
C                                                                       DFMF  50
C        PURPOSE                                                        DFMF  60
C           TO FIND A LOCAL MINIMUM OF A FUNCTION OF SEVERAL VARIABLES  DFMF  70
C           BY THE METHOD OF FLETCHER AND POWELL                        DFMF  80
C                                                                       DFMF  90
C        USAGE                                                          DFMF 100
C           CALL DFMFP(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)               DFMF 110
C                                                                       DFMF 120
C        DESCRIPTION OF PARAMETERS                                      DFMF 130
C           FUNCT  - USER-WRITTEN SUBROUTINE CONCERNING THE FUNCTION TO DFMF 140
C                    BE MINIMIZED. IT MUST BE OF THE FORM               DFMF 150
C                    SUBROUTINE FUNCT(N,ARG,VAL,GRAD)                   DFMF 160
C                    AND MUST SERVE THE FOLLOWING PURPOSE               DFMF 170
C                    FOR EACH N-DIMENSIONAL ARGUMENT VECTOR  ARG,       DFMF 180
C                    FUNCTION VALUE AND GRADIENT VECTOR MUST BE COMPUTEDDFMF 190
C                    AND, ON RETURN, STORED IN VAL AND GRAD RESPECTIVELYDFMF 200
C                    ARG,VAL AND GRAD MUST BE OF DOUBLE PRECISION.      DFMF 210
C           N      - NUMBER OF VARIABLES                               DFMF 220
C           X      - VECTOR OF DIMENSION N CONTAINING THE INITIAL      DFMF 230
C                    ARGUMENT WHERE THE ITERATION STARTS. ON RETURN,    DFMF 240
C                    X HOLDS THE ARGUMENT CORRESPONDING TO THE          DFMF 250
C                    COMPUTED MINIMUM FUNCTION VALUE                    DFMF 260
C                    DOUBLE PRECISION VECTOR.                           DFMF 270
C           F      - SINGLE VARIABLE CONTAINING THE MINIMUM FUNCTION   DFMF 280
C                    VALUE ON RETURN, I.E. F=F(X).                      DFMF 290
C                    DOUBLE PRECISION VARIABLE.                         DFMF 300
C           G      - VECTOR OF DIMENSION N CONTAINING THE GRADIENT     DFMF 310
C                    VECTOR CORRESPONDING TO THE MINIMUM ON RETURN,     DFMF 320
C                    I.E. G=G(X).                                       DFMF 330
C                    DOUBLE PRECISION VECTOR.                           DFMF 340
C           EST    - IS AN ESTIMATE OF THE MINIMUM FUNCTION VALUE.     DFMF 350
C                    SINGLE PRECISION VARIABLE.                         DFMF 360
C           EPS    - TESTVALUE REPRESENTING THE EXPECTED ABSOLUTE ERROR.DFMF 370
C                    A REASONABLE CHOICE IS 10**(-16), I.E.             DFMF 380
C                    SOMEWHAT GREATER THAN 10**(-D), WHERE D IS THE     DFMF 390
C                    NUMBER OF SIGNIFICANT DIGITS IN FLOATING POINT     DFMF 400
C                    REPRESENTATION.                                    DFMF 410
C                    SINGLE PRECISION VARIABLE.                         DFMF 420
C           LIMIT  - MAXIMUM NUMBER OF ITERATIONS.                     DFMF 430
C           IER    - ERROR PARAMETER                                   DFMF 440
C                    IER = 0 MEANS CONVERGENCE WAS OBTAINED             DFMF 450
C                    IER = 1 MEANS NO CONVERGENCE IN LIMIT ITERATIONS   DFMF 460
C                    IER =-1 MEANS ERRORS IN GRADIENT CALCULATION       DFMF 470
C                    IER = 2 MEANS LINEAR SEARCH TECHNIQUE INDICATES    DFMF 480
C                    IT IS LIKELY THAT THERE EXISTS NO MINIMUM.         DFMF 490
C           H      - WORKING STORAGE OF DIMENSION N*(N+7)/2.           DFMF 500
C                    DOUBLE PRECISION ARRAY.                            DFMF 510
C                                                                       DFMF 520
C        REMARKS                                                        DFMF 530
C           I) THE SUBROUTINE NAME REPLACING THE DUMMY ARGUMENT  FUNCT DFMF 540
C              MUST BE DECLARED AS EXTERNAL IN THE CALLING PROGRAM.    DFMF 550
C           II) IER IS SET TO 2 IF , STEPPING IN ONE OF THE COMPUTED    DFMF 560
C              DIRECTIONS, THE FUNCTION WILL NEVER INCREASE WITHIN     DFMF 570
C              A TOLERABLE RANGE OF ARGUMENT.                          DFMF 580
C              IER = 2 MAY OCCUR ALSO IF THE INTERVAL WHERE F          DFMF 590
C              INCREASES IS SMALL AND THE INITIAL ARGUMENT WAS         DFMF 600
C              RELATIVELY FAR AWAY FROM THE MINIMUM SUCH THAT THE      DFMF 610
C              MINIMUM WAS OVERLEAPED. THIS IS DUE TO THE SEARCH       DFMF 620
C              TECHNIQUE WHICH DOUBLES THE STEPSIZE UNTIL A POINT      DFMF 630
C              IS FOUND WHERE THE FUNCTION INCREASES.                  DFMF 640
C                                                                       DFMF 650
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DFMF 660
C           FUNCT                                                       DFMF 670
C                                                                       DFMF 680
C        METHOD                                                         DFMF 690
C           THE METHOD IS DESCRIBED IN THE FOLLOWING ARTICLE            DFMF 700
C           R. FLETCHER AND M.J.D. POWELL, A RAPID DESCENT METHOD FOR   DFMF 710
C           MINIMIZATION,                                               DFMF 720
C           COMPUTER JOURNAL VOL.6, ISS. 2, 1963, PP.163-168.           DFMF 730
C                                                                       DFMF 740
C     ..................................................................DFMF 750
C                                                                       DFMF 760
      SUBROUTINE DFMFP(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)               DFMF 770
C                                                                       DFMF 780
C        DIMENSIONED DUMMY VARIABLES                                    DFMF 790
      DIMENSION H(1),X(1),G(1)                                          DFMF 800
      DOUBLE PRECISION X,F,FX,FY,OLDF,HARP,GARP,W,G,DX,DY,ALFA,DALFA,   DFMF 810
     1APOLD,T,Z,W                                                       DFMF 820
C                                                                       DFMF 830
C        COMPUTE FUNCTION VALUE AND GRADIENT VECTOR FOR INITIAL ARGUMENTDFMF 840
      CALL FUNCT(N,X,F,G)                                               DFMF 850
C                                                                       DFMF 860
C        RESET ITERATION COUNTER AND GENERATE IDENTITY MATRIX           DFMF 870
      IER=0                                                             DFMF 880
      KCLNT=0                                                           DFMF 890
      N2=N+N                                                            DFMF 900
      N3=N2+N                                                           DFMF 910
      N31=N3+1                                                          DFMF 920
    1 K=N31                                                             DFMF 930
      DO 4 J=1,N                                                        DFMF 940
      H(K)=1.D0                                                         DFMF 950
      NJ=N-J                                                            DFMF 960
      IF(NJ)5,5,2                                                       DFMF 970
    2 DO 3 L=1,NJ                                                       DFMF 980
      KL=K+L                                                            DFMF 990
    3 H(KL)=0.D0                                                        DFMF1000
    4 K=KL+1                                                            DFMF1010
C                                                                       DFMF1020
C        START ITERATION LOOP                                           DFMF1030
    5 KCLNT=KCLNT +1                                                    DFMF1040
C                                                                       DFMF1050
C        SAVE FUNCTION VALUE, ARGUMENT VECTOR AND GRADIENT VECTOR       DFMF1060
      OLDF=F                                                            DFMF1070
      DO 9 J=1,N                                                        DFMF1080
      K=N+J                                                             DFMF1090
      H(K)=G(J)                                                         DFMF1100
      K=K+N                                                             DFMF1110
      H(K)=X(J)                                                         DFMF1120
C                                                                       DFMF1130
C        DETERMINE DIRECTION VECTOR H                                   DFMF1140
      K=J+N3                                                            DFMF1150
      T=0.D0                                                            DFMF1160
      DO 8 L=1,N                                                        DFMF1170
      T=T-G(L)*H(K)                                                     DFMF1180
      IF(L-J)6,7,7                                                      DFMF1190
    6 K=K+N-L                                                           DFMF1200
      GO TO 8                                                           DFMF1210
    7 K=K+1                                                             DFMF1220
    8 CONTINUE                                                          DFMF1230
    9 H(J)=T                                                            DFMF1240
C                                                                       DFMF1250
C        CHECK WHETHER FUNCTION WILL DECREASE STEPPING ALONG H.         DFMF1260
      DY=0.D0                                                           DFMF1270
      HARP=0.D0                                                         DFMF1280
      GARP=0.D0                                                         DFMF1290
C                                                                       DFMF1300
C        CALCULATE DIRECTIONAL DERIVATIVE AND TESTVALUES FOR DIRECTION  DFMF1310
C        VECTOR H AND GRADIENT VECTOR G.                                DFMF1320
      DO 10 J=1,N                                                       DFMF1330
      HARP=HARP+DABS(H(J))                                              DFMF1340
      GARP=GARP+DABS(G(J))                                              DFMF1350
   10 DY=DY+H(J)*G(J)                                                   DFMF1360
C                                                                       DFMF1370
C        REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF DIRECTIONAL  DFMF1380
C        DERIVATIVE APPEARS TO BE POSITIVE OR ZERO.                     DFMF1390
      IF(DY)11,51,51                                                    DFMF1400
C                                                                       DFMF1410
C        REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF DIRECTION    DFMF1420
C        VECTOR H IS SMALL COMPARED TO GRADIENT VECTOR G.               DFMF1430
   11 IF(HARP/GARP-EPS)51,51,12                                         DFMF1440
C                                                                       DFMF1450
C        SEARCH MINIMUM ALONG DIRECTION H                               DFMF1460
C                                                                       DFMF1470
C        SEARCH ALONG H FOR POSITIVE DIRECTIONAL DERIVATIVE             DFMF1480
   12 FY=F                                                              DFMF1490
      ALFA=2.D0*(EST-F)/DY                                              DFMF1500
      APOLD=1.D0                                                        DFMF1510
C                                                                       DFMF1520
C        USE ESTIMATE FOR STEPSIZE ONLY IF IT IS POSITIVE AND LESS THAN DFMF1530
C        1. OTHERWISE TAKE 1. AS STEPSIZE                               DFMF1540
      IF(ALFA)15,15,13                                                  DFMF1550
   13 IF(ALFA-APOLD)14,15,15                                            DFMF1560
   14 APOLD=ALFA                                                        DFMF1570
   15 ALFA=0.D0                                                         DFMF1580
C                                                                       DFMF1590
C        SAVE FUNCTION AND DERIVATIVE VALUES FOR OLD ARGUMENT           DFMF1600
   16 FX=FY                                                             DFMF1610
      DX=DY                                                             DFMF1620
C                                                                       DFMF1630
C        STEP ARGUMENT ALONG H                                          DFMF1640
      DO 17 I=1,N                                                       DFMF1650
   17 X(I)=X(I)+APOLD*H(I)                                              DFMF1660
C                                                                       DFMF1670
C        COMPUTE FUNCTION VALUE AND GRADIENT FOR NEW ARGUMENT           DFMF1680
      CALL FUNCT(N,X,F,G)                                               DFMF1690
      FY=F                                                              DFMF1700
C                                                                       DFMF1710
C        COMPUTE DIRECTIONAL DERIVATIVE DY FOR NEW ARGUMENT.  TERMINATE DFMF1720
C        SEARCH, IF DY IS POSITIVE. IF DY IS ZERO THE MINIMUM IS FOUND  DFMF1730
      DY=0.D0                                                           DFMF1740
      DO 18 I=1,N                                                       DFMF1750
   18 DY=DY+G(I)*H(I)                                                   DFMF1760
      IF(DY)19,36,22                                                    DFMF1770
C                                                                       DFMF1780
C        TERMINATE SEARCH ALSO IF THE FUNCTION VALUE INDICATES THAT     DFMF1790
C        A MINIMUM HAS BEEN PASSED                                      DFMF1800
   19 IF(FY-F)20,22,22                                                  DFMF1810
C                                                                       DFMF1820
C        REPEAT SEARCH AND DOUBLE STEPSIZE FOR FURTHER SEARCHES         DFMF1830
   20 APOLD=APOLD+ALFA                                                  DFMF1840
      ALFA=APOLD                                                        DFMF1850
C        END OF SEARCH LOOP                                             DFMF1860
C                                                                       DFMF1870
```

```
C         TERMINATE IF THE CHANGE IN ARGUMENT GETS VERY LARGE                DFMF1880
      IF(HARP*AMBDA-1.E10)16,16,21                                           DFMF1890
C                                                                            DFMF1900
C         LINEAR SEARCH TECHNIQUE INDICATES THAT NO MINIMUM EXISTS           DFMF1910
   21 IER=2                                                                  DFMF1920
      RETURN                                                                 DFMF1930
C                                                                            DFMF1940
C         INTERPOLATE CUBICALLY IN THE INTERVAL DEFINED BY THE SEARCH        DFMF1950
C         ABOVE AND COMPUTE THE ARGUMENT X FOR WHICH THE INTERPOLATION       DFMF1960
C         POLYNOMIAL IS MINIMIZED                                            DFMF1970
   22 T=0.D0                                                                 DFMF1980
   23 IF(AMBDA)24,36,24                                                      DFMF1990
   24 Z=3.D0*(FX-FY)/AMBDA+DX+DY                                             DFMF2000
      ALFA=DMAX1(DABS(Z),DABS(DX),DABS(DY))                                  DFMF2010
      DALFA=Z/ALFA                                                           DFMF2020
      DALFA=DALFA*DALFA-DX/ALFA*DY/ALFA                                      DFMF2030
      IF(DALFA)51,25,25                                                      DFMF2040
   25 W=ALFA*DSQRT(DALFA)                                                    DFMF2050
      ALFA=DY-DX+W+W                                                         DFMF2060
      IF(ALFA) 250,251,250                                                   DFMF2061
  250 ALFA=(DY-Z+W)/ALFA                                                     DFMF2062
      GO TO 252                                                              DFMF2063
  251 ALFA=(Z+DY-W)/(Z+DX+Z+DY)                                              DFMF2064
  252 ALFA=ALFA*AMBDA                                                        DFMF2065
      DO 26 I=1,N                                                            DFMF2070
   26 X(I)=X(I)+(T-ALFA)*H(I)                                                DFMF2080
C                                                                            DFMF2090
C         TERMINATE, IF THE VALUE OF THE ACTUAL FUNCTION AT X IS LESS        DFMF2100
C         THAN THE FUNCTION VALUES AT THE INTERVAL ENDS. OTHERWISE REDUCEDFMF2110
C         THE INTERVAL BY CHOOSING ONE END-POINT EQUAL TO X AND REPEAT       DFMF2120
C         THE INTERPOLATION. WHICH END-POINT IS CHOOSEN DEPENDS ON THE       DFMF2130
C         VALUE OF THE FUNCTION AND ITS GRADIENT AT X                        DFMF2140
C                                                                            DFMF2150
      CALL FUNCT(N,X,F,G)                                                    DFMF2160
      IF(F-FX)27,27,28                                                       DFMF2170
   27 IF(F-FY)26,36,26                                                       DFMF2180
   28 DALFA=0.D0                                                             DFMF2190
      DO 29 I=1,N                                                            DFMF2200
   29 DALFA=DALFA+G(I)*H(I)                                                  DFMF2210
      IF(DALFA)30,33,33                                                      DFMF2220
   30 IF(F-FX)32,31,33                                                       DFMF2230
   31 IF(DX-DALFA)32,36,32                                                   DFMF2240
   32 FX=F                                                                   DFMF2250
      DX=DALFA                                                               DFMF2260
      T=ALFA                                                                 DFMF2270
      AMBDA=ALFA                                                             DFMF2280
      GO TO 23                                                               DFMF2290
   33 IF(FY-F)35,34,35                                                       DFMF2300
   34 IF(DY-DALFA)35,36,35                                                   DFMF2310
   35 FY=F                                                                   DFMF2320
      DY=DALFA                                                               DFMF2330
      AMBDA=AMBDA-ALFA                                                       DFMF2340
      GO TO 22                                                               DFMF2350
C                                                                            DFMF2360
C         TERMINATE, IF FUNCTION HAS NOT DECREASED DURING LAST ITERATION DFMF2370
   36 IF(DDOF-F+EPS)51,38,38                                                 DFMF2380
C                                                                            DFMF2390
C         COMPUTE DIFFERENCE VECTORS OF ARGUMENT AND GRADIENT FROM           DFMF2400
C         TWO CONSECUTIVE ITERATIONS                                         DFMF2410
   38 DO 37 J=1,N                                                            DFMF2420
      K=N+J                                                                  DFMF2430
      H(K)=G(J)-H(K)                                                         DFMF2440
      K=N+K                                                                  DFMF2450
   37 H(K)=X(J)-H(K)                                                         DFMF2460
C                                                                            DFMF2470
C         TEST LENGTH OF ARGUMENT DIFFERENCE VECTOR AND DIRECTION VECTOR DFMF2480
C         IF AT LEAST N ITERATIONS HAVE BEEN EXECUTED. TERMINATE, IF         DFMF2490
C         BOTH ARE LESS THAN  EPS                                            DFMF2500
      IER=0                                                                  DFMF2510
      IF(KCLNT-N)42,39,39                                                    DFMF2520
   39 T=0.D0                                                                 DFMF2530
      Z=0.D0                                                                 DFMF2540
      DO 40 J=1,N                                                            DFMF2550
      K=N+J                                                                  DFMF2560
      W=H(K)                                                                 DFMF2570
      K=K+N                                                                  DFMF2580
      T=T+DABS(H(K))                                                         DFMF2590
   40 Z=Z+W*H(K)                                                             DFMF2600
      IF(HARP-EPS)41,41,42                                                   DFMF2610
   41 IF(T-EPS)56,56,42                                                      DFMF2620
C                                                                            DFMF2630
C         TERMINATE, IF NUMBER OF ITERATIONS WOULD EXCEED  LIMIT             DFMF2640
   42 IF(KCLNT-LIMIT)43,50,50                                                DFMF2650
C                                                                            DFMF2660
C         PREPARE UPDATING OF MATRIX H                                       DFMF2670
   43 ALFA=0.D0                                                              DFMF2680
      DO 47 J=1,N                                                            DFMF2690
      K=J+N2                                                                 DFMF2700
      W=0.D0                                                                 DFMF2710
      DO 46 L=1,N                                                            DFMF2720
      KL=N+L                                                                 DFMF2730
      W=W+H(KL)*H(K)                                                         DFMF2740
      IF(L-J)44,45,45                                                        DFMF2750
   44 K=K+N-L                                                                DFMF2760
      GO TO 46                                                               DFMF2770
   45 K=K+1                                                                  DFMF2780
   46 CONTINUE                                                               DFMF2790
      K=N+J                                                                  DFMF2800
      ALFA=ALFA+W*H(K)                                                       DFMF2810
   47 H(J)=W                                                                 DFMF2820
C                                                                            DFMF2830
C         REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF RESULTS          DFMF2840
C         ARE NOT SATISFACTORY                                               DFMF2850
      IF(Z*ALFA)48,1,48                                                      DFMF2860
C                                                                            DFMF2870
C         UPDATE MATRIX H                                                    DFMF2880
   48 K=N31                                                                  DFMF2890
      DO 49 L=1,N                                                            DFMF2900
      KL=N2+L                                                                DFMF2910
      DO 49 J=L,N                                                            DFMF2920
      NJ=N2+J                                                                DFMF2930
      H(K)=H(K)+H(KL)*H(NJ)/Z-H(L)*H(J)/ALFA                                 DFMF2940
   49 K=K+1                                                                  DFMF2950
      GO TO 5                                                                DFMF2960
C         END OF ITERATION LOOP                                              DFMF2970
C                                                                            DFMF2980
C         NO CONVERGENCE AFTER  LIMIT  ITERATIONS                            DFMF2990
   50 IER=1                                                                  DFMF3000
      RETURN                                                                 DFMF3010
C                                                                            DFMF3020
C         RESTORE OLD VALUES OF FUNCTION AND ARGUMENTS                       DFMF3030
   51 DO 52 J=1,N                                                            DFMF3040
      K=N2+J                                                                 DFMF3050
   52 X(J)=H(K)                                                              DFMF3060
      CALL FUNCT(N,X,F,G)                                                    DFMF3070
C                                                                            DFMF3080
C         REPEAT SEARCH IN DIRECTION OF STEEPEST DESCENT IF DERIVATIVE       DFMF3090
C         FAILS TO BE SUFFICIENTLY SMALL                                     DFMF3100
      IF(GARP-EPS)55,55,53                                                   DFMF3110
C                                                                            DFMF3120
C         TEST FOR REPEATED FAILURE OF ITERATION                             DFMF3130
   53 IF(IER)56,54,54                                                        DFMF3140
   54 IER=-1                                                                 DFMF3150
      GOTO 1                                                                 DFMF3160
   55 IER=0                                                                  DFMF3170
   56 RETURN                                                                 DFMF3180
      END                                                                    DFMF3190
```

Subroutines FMCG and DFMCG

These subroutines perform the calculation of an unconstrained minimum of a function of several variables using conjugate gradients. The underlying method is described in the article by R. Fletcher and C. M. Reeves, "Function minimization by conjugate gradients", Computer Journal, vol. 7, no. 2, 1964, pp. 149-154.

It is assumed that the function f(x) of the n variables $x_1, \ldots, x_n$ may be computed, together with its gradient g(x) for any point $x = \begin{pmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix}$. In the neighborhood of the required minimum x_{min} the function f(x) is approximated closely by the first three terms of its Taylor expansion, giving

$$f(x) = f(x_{min}) + \frac{1}{2}(x - x_{min})^T G(x_{min})(x - x_{min})$$

since $g(x_{min}) = 0$. The quantity $G(x_{min})$ means the matrix of second-order partial derivatives taken at the point x_{min}.

From the Taylor expansion, the gradient is seen to be approximately:

$$g(x) = G(x_{min}) \cdot (x - x_{min})$$

The condition for the gradient to vanish is:

$$G(x_{min}) \cdot x = G(x_{min}) \cdot x_{min}$$

For solution of these equations, directions $h^{(0)}$, $h^{(1)}$, ... are generated such that $h^{(i+1)}$ is a linear combination of $-g^{(i+1)}(x^{(i+1)})$ and $h^{(0)}$, $h^{(1)}$, ..., $h^{(i)}$, so that G-orthogonality is satisfied, that is, $h^{(i)T} Gh^{(j)} = 0$ for $i \neq j$.

An easy calculation gives the following results. If a sequence of successive linear searches in directions $h^{(0)} = -g^{(0)}$, $h^{(i+1)} = -g^{(i+1)} + \beta_i \cdot h^{(i)}$ is performed with:

$$_i = \frac{(g^{(i+1)}, g^{(i+1)})}{(g^{(i)}, g^{(i)})}, \quad x^{(i+1)} = x^{(i)} + t_i h^{(i)} \quad \text{and}$$

$$_g{}^{(i+1)}, h^{(i)}) = 0$$

ıen this process is guaranteed to locate the mini-
ıum of a quadratic function in n iterations at most.
or general functions the process is iterative rather
ıan n-step. (This is true even for quadratic func-
ons due to roundoff errors.)

Strategy for termination of the successive linear
earches is as follows:

1. If the change of the argument is very small,
nd at least n + 1 iterations have been performed,
ıe search for the minimum is terminated.

2. If the function value has not decreased in the
ıst linear search, the minimization is terminated.

3. If the number of iterations exceeds an upper
ound furnished by the user, further calculation is
ypassed, and an error code is set to 1 indicating
oor convergence.

4. If one of the successive linear searches indi-
ates that no constrained minimum exists, further
alculation is bypassed, and the error code is set to
indicating that it is likely that no minimum exists.

5. After each cycle of n + 1 iterations, the next
teration is in the direction of steepest descent.

The linear search technique used in subroutines
'MCG and DFMCG is described in the writeup for
ubroutines FMFP and DFMFP.

```
C                                                                       FMCG  10
C     ..................................................................FMCG  20
C                                                                       FMCG  30
C        SUBROUTINE FMCG                                                FMCG  40
C                                                                       FMCG  50
C        PURPOSE                                                        FMCG  60
C           TO FIND A LOCAL MINIMUM OF A FUNCTION OF SEVERAL VARIABLES  FMCG  70
C           BY THE METHOD OF CONJUGATE GRADIENTS                        FMCG  80
C                                                                       FMCG  90
C        USAGE                                                          FMCG 100
C           CALL FMCG(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)                FMCG 110
C                                                                       FMCG 120
C        DESCRIPTION OF PARAMETERS                                      FMCG 130
C           FUNCT  - USER-WRITTEN SUBROUTINE CONCERNING THE FUNCTION TO FMCG 140
C                    BE MINIMIZED. IT MUST BE OF THE FORM               FMCG 150
C                    SUBROUTINE FUNCT(N,ARG,VAL,GRAD)                   FMCG 160
C                    AND MUST SERVE THE FOLLOWING PURPOSE               FMCG 170
C                    FOR EACH N-DIMENSIONAL ARGUMENT VECTOR  ARG,       FMCG 180
C                    FUNCTION VALUE AND GRADIENT VECTOR MUST BE COMPUTEDFMCG 190
C                    AND, ON RETURN, STORED IN VAL AND GRAD RESPECTIVELYFMCG 200
C           N      - NUMBER OF VARIABLES                               FMCG 210
C           X      - VECTOR OF DIMENSION N CONTAINING THE INITIAL      FMCG 220
C                    ARGUMENT WHERE THE ITERATION STARTS. ON RETURN,   FMCG 230
C                    X HOLDS THE ARGUMENT CORRESPONDING TO THE         FMCG 240
C                    COMPUTED MINIMUM FUNCTION VALUE                   FMCG 250
C           F      - SINGLE VARIABLE CONTAINING THE MINIMUM FUNCTION   FMCG 260
C                    VALUE ON RETURN, I.E. F=F(X).                     FMCG 270
C           G      - VECTOR OF DIMENSION N CONTAINING THE GRADIENT     FMCG 280
C                    VECTOR CORRESPONDING TO THE MINIMUM ON RETURN,    FMCG 290
C                    I.E. G=G(X).                                      FMCG 300
C           EST    - IS AN ESTIMATE OF THE MINIMUM FUNCTION VALUE.     FMCG 310
C           EPS    - TESTVALUE REPRESENTING THE EXPECTED ABSOLUTE ERROR.FMCG 320
C                    A REASONABLE CHOICE IS 10**(-E), I.E.             FMCG 330
C                    SOMEWHAT GREATER THAN 10**(-D), WHERE D IS THE    FMCG 340
C                    NUMBER OF SIGNIFICANT DIGITS IN FLOATING POINT    FMCG 350
C                    REPRESENTATION.                                   FMCG 360
C           LIMIT  - MAXIMUM NUMBER OF ITERATIONS.                     FMCG 370
C           IER    - ERROR PARAMETER                                   FMCG 380
C                    IER = 0 MEANS CONVERGENCE HAS OBTAINED            FMCG 390
C                    IER = 1 MEANS NO CONVERGENCE IN LIMIT ITERATIONS  FMCG 400
C                    IER =-1 MEANS ERRORS IN GRADIENT CALCULATION      FMCG 410
C                    IER = 2 MEANS LINEAR SEARCH TECHNIQUE INDICATES   FMCG 420
C                    IT IS LIKELY THAT THERE EXISTS NO MINIMUM.        FMCG 430
C           H      - WORKING STORAGE OF DIMENSION 2*N.                 FMCG 440
C                                                                       FMCG 450
C        REMARKS                                                        FMCG 460
C           I) THE SUBROUTINE NAME REPLACING THE DUMMY ARGUMENT  FUNCT  FMCG 470
C              MUST BE DECLARED AS EXTERNAL IN THE CALLING PROGRAM.     FMCG 480
C           II) IER IS SET TO 2 IF , STEPPING IN ONE OF THE COMPUTED    FMCG 490
C               DIRECTIONS, THE FUNCTION WILL NEVER INCREASE WITHIN     FMCG 500
C               A TOLERABLE RANGE OF ARGUMENT.                          FMCG 510
C               IER = 2 MAY OCCUR ALSO IF THE INTERVAL WHERE F          FMCG 520
C               INCREASES IS SMALL AND THE INITIAL ARGUMENT WAS         FMCG 530
C               RELATIVELY FAR AWAY FROM THE MINIMUM SUCH THAT THE      FMCG 540
C               MINIMUM WAS OVERLEAPED. THIS IS DUE TO THE SEARCH       FMCG 550
C               TECHNIQUE WHICH DOUBLES THE STEPSIZE UNTIL A POINT      FMCG 560
C               IS FOUND WHERE THE FUNCTION INCREASES.                  FMCG 570
C                                                                       FMCG 580
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  FMCG 590
C           FUNCT                                                       FMCG 600
C                                                                       FMCG 610
C        METHOD                                                         FMCG 620
C           THE METHOD IS DESCRIBED IN THE FOLLOWING ARTICLE            FMCG 630
C           R.FLETCHER AND C.M.REEVES, FUNCTION MINIMIZATION BY         FMCG 640
C           CONJUGATE GRADIENTS.                                        FMCG 650
C           COMPUTER JOURNAL VOL.7, ISS.2, 1964, PP.149-154.            FMCG 660
C                                                                       FMCG 670
C     ..................................................................FMCG 680
C                                                                       FMCG 690
      SUBROUTINE FMCG(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)                FMCG 700
C                                                                       FMCG 710
C        DIMENSIONED DUMMY VARIABLES                                    FMCG 720
      DIMENSION X(1),G(1),H(1)                                          FMCG 730
C                                                                       FMCG 740
C                                                                       FMCG 750
C        COMPUTE FUNCTION VALUE AND GRADIENT VECTOR FOR INITIAL ARGUMENTFMCG 760
      CALL FUNCT(N,X,F,G)                                               FMCG 770
C                                                                       FMCG 780
C        RESET ITERATION COUNTER                                        FMCG 790
      KCNT=0                                                            FMCG 800
      IER=0                                                             FMCG 810
      N1=N+1                                                            FMCG 820
C                                                                       FMCG 830
C        START ITERATION CYCLE FOR EVERY N+1 ITERATIONS                 FMCG 840
    1 DO 43 II=1,N1                                                     FMCG 850
C                                                                       FMCG 860
C        STEP ITERATION COUNTER AND SAVE FUNCTION VALUE                 FMCG 870
      KCNT=KCNT+1                                                       FMCG 880
      OLDF=F                                                            FMCG 890
C                                                                       FMCG 900
C        COMPUTE SQUARE OF GRADIENT AND TERMINATE IF ZERO               FMCG 910
      GNRM=0.                                                           FMCG 920
      DO 2 J=1,N                                                        FMCG 930
    2 GNRM=GNRM+G(J)*G(J)                                               FMCG 940
      IF(GNRM)46,46,3                                                   FMCG 950
C                                                                       FMCG 960
C        EACH TIME THE ITERATION LOOP IS EXECUTED , THE FIRST STEP WILL FMCG 970
C        BE IN DIRECTION OF STEEPEST DESCENT                            FMCG 980
    3 IF(II-1)4,4,6                                                     FMCG 990
    4 DO 5 J=1,N                                                        FMCG1000
    5 H(J)=-G(J)                                                        FMCG1010
      GO TO 8                                                           FMCG1020
C                                                                       FMCG1030
C        FURTHER DIRECTION VECTORS H WILL BE CHOSEN CORRESPONDING       FMCG1040
C        TO THE CONJUGATE GRADIENT METHOD                               FMCG1050
    6 BETA=GNRM/OLDG                                                    FMCG1060
      DO 7 J=1,N                                                        FMCG1070
    7 H(J)=BETA*H(J)-G(J)                                               FMCG1080
C                                                                       FMCG1090
C        COMPUTE TESTVALUE FOR DIRECTIONAL VECTOR AND DIRECTIONAL       FMCG1100
C        DERIVATIVE                                                     FMCG1110
    8 CY=0.                                                             FMCG1120
      HNRM=0.                                                           FMCG1130
      DO 9 J=1,N                                                        FMCG1140
      K=J+N                                                             FMCG1150
      HNRM=HNRM+ABS(H(J))                                               FMCG1160
C                                                                       FMCG1170
C        SAVE ARGUMENT VECTOR                                           FMCG1180
      H(K)=X(J)                                                         FMCG1190
      HNRM=HNRM+ABS(H(J))                                               FMCG1200
    9 CY=CY+H(J)*G(J)                                                   FMCG1210
C                                                                       FMCG1220
C        CHECK WHETHER FUNCTION WILL DECREASE STEPPING ALONG H AND      FMCG1230
C        SKIP LINEAR SEARCH ROUTINE IF NOT                              FMCG1240
      IF(CY)10,42,42                                                    FMCG1250
C                                                                       FMCG1260
C        COMPUTE SCALE FACTOR USED IN LINEAR SEARCH SUBROUTINE          FMCG1270
   10 SNRM=1./HNRM                                                      FMCG1280
C
C        SEARCH MINIMUM ALONG DIRECTION H                               FMCG1290
C                                                                       FMCG1300
C        SEARCH ALONG H FOR POSITIVE DIRECTIONAL DERIVATIVE             FMCG1310
      FY=F                                                              FMCG1320
      ALFA=2.*(EST-F)/CY                                                FMCG1330
      ABETA=SNRM                                                        FMCG1340
C                                                                       FMCG1350
C        USE ESTIMATE FOR STEPSIZE ONLY IF IT IS POSITIVE AND LESS THAN FMCG1360
C        SNRM. OTHERWISE TAKE SNRM AS STEPSIZE.                         FMCG1370
      IF(ALFA)12,13,11                                                  FMCG1380
   11 IF(ALFA-ABETA)12,13,13                                            FMCG1390
   12 ABETA=ALFA                                                        FMCG1400
   13 ALFA=0.                                                           FMCG1410
C                                                                       FMCG1420
C        SAVE FUNCTION AND DERIVATIVE VALUES FOR OLD ARGUMENT           FMCG1430
   14 FX=FY                                                             FMCG1440
      CX=CY                                                             FMCG1450
C                                                                       FMCG1460
C        STEP ARGUMENT ALONG H                                          FMCG1470
      DO 15 I=1,N                                                       FMCG1480
   15 X(I)=X(I)+ABETA*H(I)                                              FMCG1490
C                                                                       FMCG1500
C        COMPUTE FUNCTION VALUE AND GRADIENT FOR NEW ARGUMENT           FMCG1510
      CALL FUNCT(N,X,F,G)                                               FMCG1520
      FY=F                                                              FMCG1530
C                                                                       FMCG1540
C        COMPUTE DIRECTIONAL DERIVATIVE BY FOR NEW ARGUMENT.  TERMINATE FMCG1550
C        SEARCH, IF CY POSITIVE. IF CY IS ZERO THE MINIMUM IS FOUND     FMCG1560
      CY=0.                                                             FMCG1570
      DO 16 I=1,N                                                       FMCG1580
   16 CY=CY+G(I)*H(I)                                                   FMCG1590
      IF(CY)17,38,20                                                    FMCG1600
C                                                                       FMCG1610
C        TERMINATE SEARCH ALSO IF THE FUNCTION VALUE INDICATES THAT     FMCG1620
C        A MINIMUM HAS BEEN PASSED                                      FMCG1630
   17 IF(FY-FX)18,20,20                                                 FMCG1640
C                                                                       FMCG1650
C        REPEAT SEARCH AND DOUBLE STEPSIZE FOR FURTHER SEARCHES         FMCG1660
   18 ABETA=ABETA+ALFA                                                  FMCG1670
      ALFA=ABETA                                                        FMCG1680
C                                                                       FMCG1690
C        TERMINATE IF THE CHANGE IN ARGUMENT GETS VERY LARGE            FMCG1700
      IF(HNRM*ABETA-1.E10)14,14,19                                      FMCG1710
C                                                                       FMCG1720
C        LINEAR SEARCH TECHNIQUE INDICATES THAT NO MINIMUM EXISTS       FMCG1730
   19 IER=2                                                             FMCG1740
C                                                                       FMCG1741
C        RESTORE OLD VALUES OF FUNCTION AND ARGUMENTS                   FMCG1742
      F=OLDF                                                            FMCG1743
      DO 100 J=1,N                                                      FMCG1744
      G(J)=H(J)                                                         FMCG1745
      K=N+J                                                             FMCG1746
  100 X(J)=H(K)                                                         FMCG1747
      RETURN                                                            FMCG1750
C        END OF SEARCH LOOP                                             FMCG1760
C                                                                       FMCG1770
C        INTERPOLATE CUBICALLY IN THE INTERVAL DEFINED BY THE SEARCH    FMCG1780
C        ABOVE AND COMPUTE THE ARGUMENT X FOR WHICH THE INTERPOLATION   FMCG1790
C        POLYNOMIAL IS MINIMIZED                                        FMCG1800
C                                                                       FMCG1810
   20 Z=0.                                                              FMCG1820
   21 IF(ABETA)22,38,22                                                 FMCG1830
   22 Z=3.*(FX-FY)/ABETA+CX+CY                                          FMCG1840
      ALFA=AMAX1(ABS(Z),ABS(CX),ABS(CY))                                FMCG1850
      CALFA=Z/ALFA                                                      FMCG1860
      DALFA=CALFA*(Z-CX/ALFA*CY/ALFA                                    FMCG1870
      IF(DALFA)23,27,27                                                 FMCG1880
C                                                                       FMCG1890
C        RESTORE OLD VALUES OF FUNCTION AND ARGUMENTS                   FMCG1900
   23 DO 24 J=1,N                                                       FMCG1910
      K=N+J                                                             FMCG1920
   24 X(J)=H(K)                                                         FMCG1930
      CALL FUNCT(N,X,F,G)                                               FMCG1940
C                                                                       FMCG1950
C        TEST FOR REPEATED FAILURE OF ITERATION                         FMCG1960
   25 IF(IER)47,26,47                                                   FMCG1970
   26 IER=-1                                                            FMCG1980
      GOTO 1                                                            FMCG1990
   27 W=ALFA*SQRT(DALFA)                                                FMCG2000
      ALFA=CY-CX+W+W                                                    FMCG2010
      IF(ALFA)270,271,270                                               FMCG2011
  270 ALFA=(CY-Z+W)/ALFA                                                FMCG2012
      GO TO 272                                                         FMCG2013
  271 ALFA=(Z+CY-W)/(Z+CX+Z+CY)                                         FMCG2014
  272 ALFA=ALFA*ABETA                                                   FMCG2015
      DO 28 I=1,N                                                       FMCG2020
   28 X(I)=X(I)+(T-ALFA)*H(I)                                           FMCG2030
C                                                                       FMCG2040
C        TERMINATE, IF THE VALUE OF THE ACTUAL FUNCTION AT X IS LESS    FMCG2050
C        THAN THE FUNCTION VALUES AT THE INTERVAL ENDS. OTHERWISE REDUCEFMCG2060
C        THE INTERVAL BY CHOOSING ONE END-POINT EQUAL TO X AND REPEAT   FMCG2070
C        THE INTERPOLATION.  WHICH END-POINT IS CHOSEN DEPENDS ON THE   FMCG2080
C        VALUE OF THE FUNCTION AND ITS GRADIENT AT X                    FMCG2090
C                                                                       FMCG2100
      CALL FUNCT(N,X,F,G)                                               FMCG2110
      IF(F-FX)29,29,30                                                  FMCG2120
   29 IF(F-FY)30,38,30                                                  FMCG2130
C                                                                       FMCG2140
C        COMPUTE DIRECTIONAL DERIVATIVE                                 FMCG2150
   30 CALFA=0.                                                          FMCG2160
      DO 31 I=1,N                                                       FMCG2170
   31 CALFA=CALFA+G(I)*H(I)                                             FMCG2180
      IF(CALFA)32,35,35                                                 FMCG2190
   32 IF(F-FX)34,33,35                                                  FMCG2200
   33 IF(CX-CALFA)34,38,34                                              FMCG2210
   34 FX=F                                                              FMCG2220
      CX=CALFA                                                          FMCG2230
      T=ALFA                                                            FMCG2240
      ABETA=ALFA                                                        FMCG2250
      GO TO 21                                                          FMCG2260
   35 IF(FY-F)37,36,37                                                  FMCG2270
   36 IF(CY-CALFA)37,38,37                                              FMCG2280
   37 FY=F                                                              FMCG2290
      CY=CALFA                                                          FMCG2300
      ABETA=ABETA-ALFA                                                  FMCG2310
      GO TO 20                                                          FMCG2320
C                                                                       FMCG2330
C        TERMINATE, IF FUNCTION HAS NOT DECREASED DURING LAST ITERATION FMCG2340
C        OTHERWISE SAVE GRADIENT NORM                                   FMCG2350
   38 IF(OLDF-F+EPS)39,25,39                                            FMCG2360
   39 OLDG=GNRM                                                         FMCG2370
C                                                                       FMCG2380
C        COMPUTE DIFFERENCE OF NEW AND OLD ARGUMENT VECTOR              FMCG2390
      T=0.                                                              FMCG2400
      DO 40 J=1,N                                                       FMCG2410
      K=J+N                                                             FMCG2420
      H(K)=X(J)-H(K)                                                    FMCG2430
   40 T=T+ABS(H(K))                                                     FMCG2440
```

```
C                                                                       FMCG2450
C       TEST LENGTH OF DIFFERENCE VECTOR IF AT LEAST N+1 ITERATIONS     FMCG2460
C       HAVE BEEN EXECUTED. TERMINATE, IF LENGTH IS LESS THAN EPS       FMCG2470
      IF(KCUNT-N)42,41,41                                               FMCG2480
   41 IF(T-EPS)45,45,42                                                 FMCG2490
C                                                                       FMCG2500
C       TERMINATE, IF NUMBER OF ITERATIONS WOULD EXCEED  LIMIT          FMCG2510
   42 IF(KCUNT-LIMIT)43,44,44                                           FMCG2520
   43 IER=0                                                             FMCG2530
C       END OF ITERATION CYCLE                                          FMCG2540
C                                                                       FMCG2550
C       START NEXT ITERATION CYCLE                                      FMCG2560
      GO TO 1                                                           FMCG2570
C                                                                       FMCG2580
C       NO CONVERGENCE AFTER  LIMIT  ITERATIONS                         FMCG2590
   44 IER=1                                                             FMCG2600
      IF(GNRM-EPS)46,46,47                                              FMCG2610
C                                                                       FMCG2620
C       TEST FOR SUFFICIENTLY SMALL GRADIENT                            FMCG2630
   45 IF(GNRM-EPS)46,46,25                                              FMCG2640
   46 IER=0                                                             FMCG2650
   47 RETURN                                                            FMCG2660
      END                                                               FMCG2670
```

```
C                                                                       DFMC  10
C     ******************************************************************DFMC  20
C                                                                       DFMC  30
C       SUBROUTINE CFMCG                                                DFMC  40
C                                                                       DFMC  50
C       PURPOSE                                                         DFMC  60
C          TO FIND A LOCAL MINIMUM OF A FUNCTION OF SEVERAL VARIABLES   DFMC  70
C          BY THE METHOD OF CONJUGATE GRADIENTS                         DFMC  80
C                                                                       DFMC  90
C       USAGE                                                           DFMC 100
C          CALL CFMCG(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)                DFMC 110
C                                                                       DFMC 120
C       DESCRIPTION OF PARAMETERS                                       DFMC 130
C          FUNCT  - USER-WRITTEN SUBROUTINE CONCERNING THE FUNCTION TO  DFMC 140
C                   BE MINIMIZED. IT MUST BE OF THE FORM                DFMC 150
C                   SUBROUTINE FUNCTN(ARG,VAL,GRAD)                     DFMC 160
C                   AND MUST SERVE THE FOLLOWING PURPOSE                DFMC 170
C                   FOR EACH N-DIMENSIONAL ARGUMENT VECTOR  ARG,        DFMC 180
C                   FUNCTION VALUE AND GRADIENT VECTOR MUST BE COMPUTEDDFMC 190
C                   AND, ON RETURN, STORED IN VAL AND GRAD RESPECTIVELYDFMC 200
C                   ARG,VAL AND GRAD MUST BE OF DOUBLE PRECISION.       DFMC 210
C          N      - NUMBER OF VARIABLES                                DFMC 220
C          X      - VECTOR OF DIMENSION N CONTAINING THE INITIAL       DFMC 230
C                   ARGUMENT WHERE THE ITERATION STARTS. ON RETURN,     DFMC 240
C                   X HOLDS THE ARGUMENT CORRESPONDING TO THE           DFMC 250
C                   COMPUTED MINIMUM FUNCTION VALUE                     DFMC 260
C                   DOUBLE PRECISION VECTOR.                            DFMC 270
C          F      - SINGLE VARIABLE CONTAINING THE MINIMUM FUNCTION    DFMC 280
C                   VALUE ON RETURN, I.E. F=F(X).                       DFMC 290
C                   DOUBLE PRECISION VARIABLE.                          DFMC 300
C          G      - VECTOR OF DIMENSION N CONTAINING THE GRADIENT      DFMC 310
C                   VECTOR CORRESPONDING TO THE MINIMUM ON RETURN,      DFMC 320
C                   I.E. G=G(X).                                        DFMC 330
C                   DOUBLE PRECISION VECTOR.                            DFMC 340
C          EST    - IS AN ESTIMATE OF THE MINIMUM FUNCTION VALUE.      DFMC 350
C                   SINGLE PRECISION VARIABLE.                          DFMC 360
C          EPS    - TESTVALUE REPRESENTING THE EXPECTED ABSOLUTE ERROR.DFMC 370
C                   A REASONABLE CHOICE IS 10**(-16), I.E.              DFMC 380
C                   SOMEWHAT GREATER THAN 10**(-D), WHERE D IS THE      DFMC 390
C                   NUMBER OF SIGNIFICANT DIGITS IN FLOATING POINT      DFMC 400
C                   REPRESENTATION.                                     DFMC 410
C                   SINGLE PRECISION VARIABLE.                          DFMC 420
C          LIMIT  - MAXIMUM NUMBER OF ITERATIONS.                       DFMC 430
C          IER    - ERROR PARAMETER                                    DFMC 440
C                   IER = 0 MEANS CONVERGENCE WAS OBTAINED              DFMC 450
C                   IER = 1 MEANS NO CONVERGENCE IN LIMIT ITERATIONS    DFMC 460
C                   IER =-1 MEANS ERRORS IN GRADIENT CALCULATION        DFMC 470
C                   IER = 2 MEANS LINEAR SEARCH TECHNIQUE INDICATES     DFMC 480
C                   IT IS LIKELY THAT THERE EXISTS NO MINIMUM.          DFMC 490
C          H      - WORKING STORAGE OF DIMENSION 2*N.                  DFMC 500
C                   DOUBLE PRECISION ARRAY.                             DFMC 510
C                                                                       DFMC 520
C       REMARKS                                                         DFMC 530
C          I) THE SUBROUTINE NAME REPLACING THE DUMMY ARGUMENT  FUNCT   DFMC 540
C             MUST BE DECLARED AS EXTERNAL IN THE CALLING PROGRAM.      DFMC 550
C          II) IER IS SET TO 2 IF , STEPPING IN ONE OF THE COMPUTED     DFMC 560
C             DIRECTIONS, THE FUNCTION WILL NEVER INCREASE WITHIN       DFMC 570
C             A TOLERABLE RANGE OF ARGUMENT.                            DFMC 580
C             IER = 2 MAY OCCUR ALSO IF THE INTERVAL WHERE F            DFMC 590
C             INCREASES IS SMALL AND THE INITIAL ARGUMENT WAS           DFMC 600
C             RELATIVELY FAR AWAY FROM THE MINIMUM SUCH THAT THE        DFMC 610
C             MINIMUM WAS OVERLEAPED. THIS IS DUE TO THE SEARCH         DFMC 620
C             TECHNIQUE WHICH DOUBLES THE STEPSIZE UNTIL A POINT        DFMC 630
C             IS FOUND WHERE THE FUNCTION INCREASES.                    DFMC 640
C                                                                       DFMC 650
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DFMC 660
C          FUNCT                                                        DFMC 670
C                                                                       DFMC 680
C       METHOD                                                          DFMC 690
C          THE METHOD IS DESCRIBED IN THE FOLLOWING ARTICLE             DFMC 700
C          R.FLETCHER AND C.M.REEVES, FUNCTION MINIMIZATION BY          DFMC 710
C          CONJUGATE GRADIENTS.                                         DFMC 720
C          COMPUTER JOURNAL VOL.7, ISS.2, 1964, PP.149-154.             DFMC 730
C                                                                       DFMC 740
C     ******************************************************************DFMC 750
C                                                                       DFMC 760
      SUBROUTINE CFMCG(FUNCT,N,X,F,G,EST,EPS,LIMIT,IER,H)               DFMC 770
C                                                                       DFMC 780
C       DIMENSIONED DUMMY VARIABLES                                     DFMC 790
      DIMENSION X(1),G(1),H(1)                                          DFMC 800
      DOUBLE PRECISION X,G,GNRM,H,HNRM,F,FX,FY,OLDF,OLDG,SNRM,AMBDA,    DFMC 810
     1ALFA,CALFA,T,Z,W,CX,CY                                            DFMC 820
C                                                                       DFMC 830
C       COMPUTE FUNCTION VALUE AND GRADIENT VECTOR FOR INITIAL ARGUMENTDFMC 840
      CALL FUNCT(N,X,F,G)                                               DFMC 850
C                                                                       DFMC 860
C       RESET ITERATION COUNTER                                         DFMC 870
      KCUNT=0                                                           DFMC 880
      IER=0                                                             DFMC 890
      N1=N+1                                                            DFMC 900
C                                                                       DFMC 910
C       START ITERATION CYCLE FOR EVERY N+1 ITERATIONS                  DFMC 920
    1 DO 43 II=1,N1                                                     DFMC 930
C                                                                       DFMC 940
C       STEP ITERATION COUNTER AND SAVE FUNCTION VALUE                  DFMC 950
      KCUNT=KCUNT+1                                                     DFMC 960
      OLDF=F                                                            DFMC 970
C                                                                       DFMC 980
C       COMPUTE SQUARE OF GRADIENT AND TERMINATE IF ZERO                DFMC 990
      GNRM=0.D0                                                         DFMC1000
      DO 2 J=1,N                                                        DFMC1010
    2 GNRM=GNRM+G(J)*G(J)                                               DFMC1020
      IF(GNRM)46,46,3                                                   DFMC1030
C                                                                       DFMC1040
C       EACH TIME THE ITERATION LOOP IS EXECUTED , THE FIRST STEP WILL DFMC1050
C       BE IN DIRECTION OF STEEPEST DESCENT                             DFMC1060
    3 IF(II-1)4,4,6                                                     DFMC1070
    4 DO 5 J=1,N                                                        DFMC1080
    5 H(J)=-G(J)                                                        DFMC1090
      GO TO 8                                                           DFMC1100
C                                                                       DFMC1110
C       FURTHER DIRECTION VECTORS H WILL BE CHOOSEN CORRESPONDING       DFMC1120
C       TO THE CONJUGATE GRADIENT METHOD                                DFMC1130
    6 AMBDA=GNRM/OLDG                                                   DFMC1140
      DO 7 J=1,N                                                        DFMC1150
    7 H(J)=AMBDA*H(J)-G(J)                                              DFMC1160
C                                                                       DFMC1170
C       COMPUTE TESTVALUE FOR DIRECTIONAL VECTOR AND DIRECTIONAL        DFMC1180
C       DERIVATIVE                                                      DFMC1190
    8 CY=0.D0                                                           DFMC1200
      HNRM=0.D0                                                         DFMC1210
      DO 9 J=1,N                                                        DFMC1220
      K=J+N                                                             DFMC1230
C                                                                       DFMC1240
C       SAVE ARGUMENT VECTOR                                            DFMC1250
      H(K)=X(J)                                                         DFMC1260
      HNRM=HNRM+DABS(H(J))                                              DFMC1270
    9 CY=CY+H(J)*G(J)                                                   DFMC1280
C                                                                       DFMC1290
C       CHECK WHETHER FUNCTION WILL DECREASE STEPPING ALONG H AND       DFMC1300
C       SKIP LINEAR SEARCH ROUTINE IF NOT                               DFMC1310
      IF(CY)10,42,42                                                    DFMC1320
C                                                                       DFMC1330
C       COMPUTE SCALE FACTOR USED IN LINEAR SEARCH SUBROUTINE           DFMC1340
   10 SNRM=1.D0/HNRM                                                    DFMC1350
C                                                                       DFMC1360
C       SEARCH MINIMUM ALONG DIRECTION H                                DFMC1370
C                                                                       DFMC1380
C       SEARCH ALONG H FOR POSITIVE DIRECTIONAL DERIVATIVE              DFMC1390
      FY=F                                                              DFMC1400
      ALFA=2.D0*(EST-F)/CY                                              DFMC1410
      AMBDA=SNRM                                                        DFMC1420
C                                                                       DFMC1430
C       USE ESTIMATE FOR STEPSIZE ONLY IF IT IS POSITIVE AND LESS THAN DFMC1440
C       SNRM. OTHERWISE TAKE SNRM AS STEPSIZE.                          DFMC1450
      IF(ALFA)13,13,11                                                  DFMC1460
   11 IF(ALFA-AMBDA)12,12,13                                            DFMC1470
   12 AMBDA=ALFA                                                        DFMC1480
   13 ALFA=0.D0                                                         DFMC1490
C                                                                       DFMC1500
C       SAVE FUNCTION AND DERIVATIVE VALUES FOR OLD ARGUMENT            DFMC1510
   14 FX=FY                                                             DFMC1520
      CX=CY                                                             DFMC1530
C                                                                       DFMC1540
C       STEP ARGUMENT ALONG H                                           DFMC1550
      DO 15 I=1,N                                                       DFMC1560
   15 X(I)=X(I)+AMBDA*H(I)                                              DFMC1570
C                                                                       DFMC1580
C       COMPUTE FUNCTION VALUE AND GRADIENT FOR NEW ARGUMENT            DFMC1590
      CALL FUNCT(N,X,F,G)                                               DFMC1600
      FY=F                                                              DFMC1610
C                                                                       DFMC1620
C       COMPUTE DIRECTIONAL DERIVATIVE CY FOR NEW ARGUMENT.  TERMINATE DFMC1630
C       SEARCH, IF CY POSITIVE. IF CY IS ZERO THE MINIMUM IS FOUND      DFMC1640
      CY=0.D0                                                           DFMC1650
      DO 16 I=1,N                                                       DFMC1660
   16 CY=CY+G(I)*H(I)                                                   DFMC1670
      IF(CY)17,28,20                                                    DFMC1680
C                                                                       DFMC1690
C       TERMINATE SEARCH ALSO IF THE FUNCTION VALUE INDICATES THAT      DFMC1700
C       A MINIMUM HAS BEEN PASSED                                       DFMC1710
   17 IF(FY-F)18,20,20                                                  DFMC1720
C                                                                       DFMC1730
C       REPEAT SEARCH AND DOUBLE STEPSIZE FOR FURTHER SEARCHES          DFMC1740
   18 AMBDA=AMBDA+ALFA                                                  DFMC1750
      ALFA=AMBDA                                                        DFMC1760
C                                                                       DFMC1770
C       TERMINATE IF THE CHANGE IN ARGUMENT GETS VERY LARGE             DFMC1780
      IF(HNRM*AMBDA-1.D10)14,14,19                                      DFMC1790
C                                                                       DFMC1800
C       LINEAR SEARCH TECHNIQUE INDICATES THAT NO MINIMUM EXISTS        DFMC1810
   19 IER=2                                                             DFMC1820
C                                                                       DFMC1821
C       RESTORE OLD VALUES OF FUNCTION AND ARGUMENTS                    DFMC1822
      F=OLDF                                                            DFMC1823
      DO 100 J=1,N                                                      DFMC1824
      G(J)=H(J)                                                         DFMC1825
      K=N+J                                                             DFMC1826
  100 X(J)=H(K)                                                         DFMC1827
      RETURN                                                            DFMC1830
C       END OF SEARCH LOOP                                              DFMC1840
C                                                                       DFMC1850
C       INTERPOLATE CUBICALLY IN THE INTERVAL DEFINED BY THE SEARCH     DFMC1860
C       ABOVE AND COMPUTE THE ARGUMENT X FOR WHICH THE INTERPOLATION    DFMC1870
C       POLYNOMIAL IS MINIMIZED                                         DFMC1880
C                                                                       DFMC1890
   20 T=0.D0                                                            DFMC1900
   21 IF(AMBDA)22,38,22                                                 DFMC1910
   22 Z=3.D0*(FX-FY)/AMBDA+CX+CY                                        DFMC1920
      ALFA=DMAX1(DABS(Z),DABS(CX),DABS(CY))                             DFMC1930
      CALFA=Z/ALFA                                                      DFMC1940
      CALFA=CALFA*(CALFA-CX/ALFA*CY/ALFA                                DFMC1950
      IF(CALFA)23,23,27                                                 DFMC1960
C                                                                       DFMC1970
C       RESTORE OLD VALUES OF FUNCTION AND ARGUMENTS                    DFMC1980
   23 DO 24 J=1,N                                                       DFMC1990
      K=N+J                                                             DFMC2000
   24 X(J)=H(K)                                                         DFMC2010
      CALL FUNCT(N,X,F,G)                                               DFMC2020
C                                                                       DFMC2030
C       TEST FOR REPEATED FAILURE OF ITERATION                          DFMC2040
   25 IF(IER)47,26,47                                                   DFMC2050
   26 IER=-1                                                            DFMC2060
      GOTO 1                                                            DFMC2070
   27 W=ALFA*DSQRT(CALFA)                                               DFMC2080
      ALFA=CY-CX+W+W                                                    DFMC2090
      IF(ALFA)270,271,270                                               DFMC2091
  270 ALFA=(CY-2*W)/ALFA                                                DFMC2092
      GO TO 272                                                         DFMC2093
  271 ALFA=(Z+CY-W)/(Z+CX+Z+CY)                                         DFMC2094
  272 ALFA=ALFA*AMBDA                                                   DFMC2095
      DO 28 I=1,N                                                       DFMC2100
   28 X(I)=X(I)+(T-ALFA)*H(I)                                           DFMC2110
C                                                                       DFMC2120
```

```
C         TERMINATE, IF THE VALUE OF THE ACTUAL FUNCTION AT X IS LESS     DFMC2130
C         THAN THE FUNCTION VALUES AT THE INTERVAL ENDS, OTHERWISE REDUCEDFMC2140
C         THE INTERVAL BY CHOOSING ONE END-POINT EQUAL TO X AND REPEAT    DFMC2150
C         THE INTERPOLATION.  WHICH END-POINT IS CHOSEN DEPENDS ON THE    DFMC2160
C         VALUE OF THE FUNCTION AND ITS GRADIENT AT X                     DFMC2170
C                                                                         DFMC2180
      CALL FUNCT(N,X,F,G)                                                 DFMC2190
      IF(F-FX)25,25,30                                                    DFMC2200
   25 IF(F-FY)36,26,30                                                    DFMC2210
C                                                                         DFMC2220
C         COMPUTE DIRECTIONAL DERIVATIVE                                  DFMC2230
   30 DALFA=0.D0                                                          DFMC2240
      DO 31 I=1,N                                                         DFMC2250
   31 DALFA=DALFA+G(I)*H(I)                                               DFMC2260
      IF(DALFA)32,35,35                                                   DFMC2270
   32 IF(F-FX)34,33,35                                                    DFMC2280
   33 IF(DX-DALFA)34,38,34                                                DFMC2290
   34 FX=F                                                                DFMC2300
      DX=DALFA                                                            DFMC2310
      T=ALFA                                                              DFMC2320
      APREDA=ALFA                                                         DFMC2330
      GO TO 21                                                            DFMC2340
   35 IF(FY-F)37,36,37                                                    DFMC2350
   36 IF(DY-DALFA)37,38,37                                                DFMC2360
   37 FY=F                                                                DFMC2370
      DY=DALFA                                                            DFMC2380
      APREDA=APREDA-ALFA                                                  DFMC2390
      GO TO 20                                                            DFMC2400
C                                                                         DFMC2410
C         TERMINATE, IF FUNCTION HAS NOT DECREASED DURING LAST ITERATION DFMC2420
C         OTHERWISE SAVE GRADIENT NORM                                    DFMC2430
   38 IF(DLDG-FH*EPS)15,25,25                                             DFMC2440
   39 DLDG=GNRM                                                           DFMC2450
C                                                                         DFMC2460
C         COMPUTE DIFFERENCE OF NEW AND OLD ARGUMENT VECTOR               DFMC2470
      T=0.D0                                                              DFMC2480
      DO 40 J=1,N                                                         DFMC2490
      K=J+N                                                               DFMC2500
      H(K)=X(J)-H(K)                                                      DFMC2510
   40 T=T+DABS(H(K))                                                      DFMC2520
C                                                                         DFMC2530
C         TEST LENGTH OF DIFFERENCE VECTOR IF AT LEAST N+1 ITERATIONS     DFMC2540
C         HAVE BEEN EXECUTED. TERMINATE, IF LENGTH IS LESS THAN EPS       DFMC2550
      IF(KCUNT-N)42,41,41                                                 DFMC2560
   41 IF(T-EPS)45,45,42                                                   DFMC2570
C                                                                         DFMC2580
C         TERMINATE, IF NUMBER OF ITERATIONS WOULD EXCEED  LIMIT          DFMC2590
   42 IF(KCUNT-LIMIT)43,44,4                                              DFMC2600
   43 IER=0                                                               DFMC2610
C         END OF ITERATION CYCLE                                          DFMC2620
C                                                                         DFMC2630
C         START NEXT ITERATION CYCLE                                      DFMC2640
      GO TO 1                                                             DFMC2650
C                                                                         DFMC2660
C         NO CONVERGENCE AFTER  LIMIT  ITERATIONS                         DFMC2670
   44 IER=1                                                               DFMC2680
      IF(GNRM-EPS)46,46,47                                                DFMC2690
C                                                                         DFMC2700
C         TEST FOR SUFFICIENTLY SMALL GRADIENT                            DFMC2710
   45 IF(GNRM-EPS)46,46,25                                                DFMC2720
   46 IER=0                                                               DFMC2730
   47 RETURN                                                              DFMC2740
      END                                                                 DFMC2750
```

PERMUTATIONS

Let n be a positive integer and let $N = \{1, \ldots, n\}$. A permutation on N is a one-one function from N onto N. In what follows, all permutations will be on N.

Suppose P and Q are permutations. We define a new permutation $R = Q \cdot P$ by $R(j) = (Q \cdot P)(j) = Q(P(j))$ for $j \in N$. Note that in general $P \cdot Q \neq Q \cdot P$, although $\cdot$ is associative. The operation $\cdot$ is ordinary function composition. The identity permutation is the identity function I on N, defined by $I(j) = j$ for $j \in N$. An inverse of a permutation P is a permutation P^{-1} with the property that $P \cdot P^{-1} = P^{-1} \cdot P = I$. It is easily proved that every permutation has a unique inverse; indeed, if P is a permutation, its inverse P^{-1} is given by $P^{-1}(j) = s$ where s is such that $P(s) = j$, $j \in N$. Note also that if P and Q are permutations, $(Q \cdot P)^{-1} = P^{-1} \cdot Q^{-1}$. The conjugate of Q by P is the permutation $P \cdot Q \cdot P^{-1}$.

A transposition is a permutation that moves at most two elements of N; that is, T is a transposition if and only if integers i and j exist such that $T(k) = k$ for $k \neq i, j$. It may easily be proved that every permutation can be written as a product $T_1 \cdot T_2 \cdot \ldots \cdot T_n$ of transpositions where for $i = 1, \ldots, n$ there exists $k_i \in N$ such that:

$$T_i(j) = \begin{cases} k_i & \text{if } j = i \\ i & \text{if } j = k_i \\ j & \text{if } j \neq i, k_i \end{cases}$$

Every transposition is its own inverse.

If P is a permutation, we will identify P and the symbol

$$\begin{pmatrix} 1 & 2 & \ldots & n \\ p_1 & p_2 & \ldots & p_n \end{pmatrix}$$

where

$$p_i = P(i) \text{ for } i = 1, \ldots, n.$$

At times we will also identify the permutation P and the vector $(p_1, \ldots, p_n)$. Whenever this is the case we will write "... permutation $P = (p_1, \ldots, p_n)$ ...", or "permutation vector $P = (p_1, \ldots, p_n)$".

Similarly, if T is the transposition given by:

$$T(j) = \begin{cases} j & \text{if } j \neq i, k \\ i & \text{if } j = k \\ k & \text{if } j = i \end{cases}$$

then we identify T and the symbol (i, k). (This leads to no difficulty if $n = 2$, since in this case all permutations are also transpositions.)

We note above that any permutation P can be written as a product $T_1 \cdot \ldots \cdot T_n$ of transpositions of a certain form. In the new notation we have $P = (1, t_1) \cdot \ldots \cdot (n, t_n)$, $t_i \in N$, $i = 1, \ldots, n$. The t_i are generally not distinct. At times we will identify the product $(1, t_1) \cdot \ldots \cdot (n, t_n)$ and the vector $(t_1, \ldots, t_n)$. Whenever this identification is meant we will write "... transposition vector $P = (t_1, \ldots, t_n)$".

Permutations can be used to induce rearrangements of ordered collections of objects. The convention that we will use is illustrated by the following example. Suppose that A is an m by n matrix and C_i is the ith column of A for $i = 1, \ldots, n$. If the permutation $P = (p_1, \ldots, p_n)$ is applied to the columns of A, there results the m by n matrix A' whose p_i^{th} column, C_{p_i}, is C_i for $i = 1, \ldots, n$.

Examples

Suppose $n = 5$, and let P and Q be the permutations given by:

$P(1) = 3 \quad P(2) = 1 \quad P(3) = 5 \quad P(4) = 2 \quad P(5) = 4$

and

$Q(1) = 1 \quad Q(2) = 3 \quad Q(3) = 5 \quad Q(4) = 2 \quad Q(5) = 4$

Then P and Q may also be written as:

$$P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 5 & 2 & 4 \end{pmatrix} \quad \text{and} \quad Q = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 5 & 2 & 4 \end{pmatrix}$$

or as the permutation vectors:

$$P = (3, 1, 5, 2, 4) \quad \text{and} \quad Q = (1, 3, 5, 2, 4)$$

Now:

$$Q \cdot P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 5 & 2 & 4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 5 & 2 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 4 & 3 & 2 \end{pmatrix}$$

while:

$$P \cdot Q = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 5 & 2 & 4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 5 & 2 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 5 & 4 & 1 & 2 \end{pmatrix}$$

P^{-1} is the permutation given by

$$P^{-1}(1) = 2 \quad P^{-1}(2) = 4 \quad P^{-1}(3) = 1 \quad P^{-1}(4) = 5$$

$$P^{-1}(5) = 3$$

or

$$P^{-1} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 4 & 1 & 5 & 3 \end{pmatrix}$$

or the permutation vector $P = (2,4,1,5,3)$.

We now write P as a product of transpositions in the standard form defined above. We have:

$$P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 5 & 2 & 4 \end{pmatrix} = (1,3) \cdot \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 5 & 2 & 4 \end{pmatrix}$$

$$= (1,3) \cdot \left[(2,3) \cdot \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 5 & 3 & 4 \end{pmatrix} \right]$$

$$= (1,3) \cdot (2,3) \cdot$$

$$\left[(3,5) \cdot \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 5 & 4 \end{pmatrix} \right]$$

$$= (1,3) \cdot (2,3) \cdot (3,5) \cdot$$

$$\left[(4,5) \cdot \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix} \right]$$

$$= (1,3) \cdot (2,3) \cdot (3,5) \cdot (4,5) \cdot (5,5).$$

Hence P is equivalent to the transposition vector $(3,3,5,5,5)$.

Subroutine PPRCN

Subroutine PPRCN computes the permutation vector that is the composition of two given permutation vectors, and the permutation vector that is the conjugate of a given permutation vector by another permutation vector. (See the general discussion for definitions and notation.)

1. Mathematical background

Suppose that $IP1 = (p_1, \ldots, p_n)$ and $IP2 = (q_1, \ldots, q_n)$ are permutation vectors on $N = \{1, \ldots, n\}$.

a. The composition $IP3 = IP2 \cdot IP1$ of IP1 and IP2 is given by $IP3 = (r_1, \ldots, r_n)$ where $r_i = q_{p_i}$ for $i = 1, \ldots, n$.

b. The conjugate $IP3 = IP1 \cdot IP2 \cdot IP1^{-1}$ of IP2 by IP1 is given by $IP3 = (r_1, \ldots, r_n)$ where $r_{p_i} = p_{q_i}$ for $i = 1, \ldots, n$.

2. Programming considerations

Depending on whether the input parameter IPAR is nonnegative or negative, subroutine PPRCN computes either the composition of IP1 and IP2 or the conjugate of IP2 by IP1.

Subroutine PERM is used to check that n is positive and that IP1 and IP2 are permutation vectors on N. If n is not positive or if IP1 and IP2 are not both permutation vectors on N, computation is terminated as soon as the error is detected, and the error parameter IER is set to -1 or 1, respectively. If there is no error, IER is set to zero.

The computation of the resulting vector IP3 proceeds as follows.

a. If IPAR is nonnegative, for each i, $i = 1, \ldots, n$, IP3(i) is set to IP2(IP1(i)).

b. If IPAR is negative, initially subroutine PERM is used to compute $IP1^{-1}$, which is stored in IP3. Then, for $i = 1, \ldots, n$, IP3(i) is set to IP1 (IP2 (IP3(i))).

IP3 cannot have the same storage allocation as IP1 or IP2. IP1 and IP2 are returned unchanged.

```
C
C     ..................................................................
C
C        SUBROUTINE PPRCN
C
C        PURPOSE
C           TO COMPUTE, GIVEN TWO PERMUTATION VECTORS IP1 AND IP2, THE
C           COMPOSITION IP2(IP1) AND THE CONJUGATE IP1(IP2(IP1 INVERSE))
C           OF IP2 BY IP1.  (SEE THE GENERAL DISCUSSION FOR DEFINITIONS
C           AND NOTATION.)
C
C        USAGE
C           CALL PPRCN(IP1,IP2,IP3,N,IPAR,IER)
C
C        DESCRIPTION OF PARAMETERS
C           IP1  - GIVEN PERMUTATION VECTOR (DIMENSION N)
C           IP2  - GIVEN PERMUTATION VECTOR (DIMENSION N)
C           IP3  - RESULTING PERMUTATION VECTOR (DIMENSION N)
C           N    - DIMENSION OF VECTORS IP1, IP2 AND IP3
C           IPAR - INPUT PARAMETER
C                  IPAR NON-NEGATIVE - COMPUTE IP2(IP1)
C                  IPAR NEGATIVE     - COMPUTE IP1(IP2(IP1 INVERSE))
C           IER  - RESULTING ERROR PARAMETER
C                  IER=-1  -  N IS NOT POSITIVE
C                  IER= 0  -  NO ERROR
C                  IER= 1  -  IP1 AND IP2 ARE NOT BOTH PERMUTATION
C                             VECTORS ON 1,...,N
C
C        REMARKS
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.
C           (2)  IF IER=1, THEN COMPUTATION HAS BEEN UNSUCCESSFUL DUE TO
C                ERROR AND THE PARTIAL RESULTS FOUND IN IP2 ARE USELESS.
C           (3)  IP3 CANNOT HAVE THE SAME STORAGE ALLOCATION AS IP1 OR
C                IP2.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           PERM
C
C        METHOD
C           SUBROUTINE PERM IS USED TO CHECK THAT IP1 AND IP2 ARE PERMU-
C           TATION VECTORS.  IF IP2(IP1) IS COMPUTED, IP3(I) IS SET TO
C           IP2(IP1(I)) FOR I=1,...,N.  IF IP1(IP2(IP1 INVERSE)) IS
C           COMPUTED, FIRST IP3 IS SET TO IP1 INVERSE BY SUBROUTINE PERM
C           AND THEN IP3(I) IS SET TO IP1(IP2(IP3(I))) FOR I=1,...,N.
C
C     ..................................................................
C
      SUBROUTINE PPRCN(IP1,IP2,IP3,N,IPAR,IER)
C
C
      DIMENSION IP1(1),IP2(1),IP3(1)
C
C        CHECK THAT N IS POSITIVE AND THAT IP2 IS A PERMUTATION VECTOR
      CALL PERM(IP2,IP3,N,-1,IER)
C
C        TEST IER TO SEE IF THERE IS AN ERROR
      IF(IER)7,1,7
C
C        CHECK THAT IP1 IS A PERMUTATION VECTOR AND COMPUTE IP1 INVERSE
    1 CALL PERM(IP1,IP3,N,-1,IER)
C
C        TEST IER TO SEE IF THERE IS AN ERROR
      IF(IER)7,2,7


C
C        TEST IPAR FOR THE DESIRED OPERATION
    2 IF(IPAR)3,5,5
C
C        COMPUTE IP1(IP2(IP1 INVERSE))
    3 DO 4 I=1,N
      K=IP3(I)
      J=IP2(K)
    4 IP3(I)=IP1(J)
      RETURN
C
C        COMPUTE IP2(IP1)
    5 DO 6 I=1,N
      K=IP1(I)
    6 IP3(I)=IP2(K)
    7 RETURN
      END
```

Subroutine PERM

Subroutine PERM computes the permutation vector that is the inverse of a given permutation vector, the permutation vector that is equivalent to a given transposition vector, and a transposition vector that is equivalent to a given permutation vector. (See the general discussion for definitions and notation.)

1. Mathematical background

Let $N = \{1, \ldots, n\}$.

a. If $IP1 = (p_1, \ldots, p_n)$ is a permutation vector on N, its inverse, IP2, is given by $IP2 = (r_1, \ldots, r_n)$ where $r_{p_i} = i$ for $i = 1, \ldots, n$.

b. Let $IP1 = (t_1, \ldots, t_n)$ be a transposition vector on N. The computation of the permutation vector that is equivalent to IP1 is based on the identity:

$$\begin{pmatrix} 1 & \ldots & n \\ s_1 & \ldots & s_n \end{pmatrix} \cdot (i, v_i) = \begin{pmatrix} 1 & \ldots & i & \ldots & v_i & \ldots & n \\ s_1 & \ldots & s_{v_i} & \ldots & s_i & \ldots & s_n \end{pmatrix} \tag{1}$$

Applying (1) for $i = 1, \ldots, n$ to the right-hand side of the identity

$$(1, t_1) \cdot \ldots \cdot (n, t_n) = \begin{pmatrix} 1 & \ldots & n \\ 1 & \ldots & n \end{pmatrix} \cdot (1, t_1) \cdot \ldots \cdot (n, t_n)$$

there results

$$(1, t_1) \cdot \ldots \cdot (n, t_n) = \begin{pmatrix} 1 & \ldots & n \\ q_1 & \ldots & q_n \end{pmatrix}$$

The desired permutation vector, IP2, is $IP2 = (q_1, \ldots, q_n)$.

c. Let $IP1 = (p_1, \ldots, p_n)$ be a permutation vector on N. The computation of a transposition vector that is equivalent to IP1 is based on the identity:

$$\begin{pmatrix} 1 & \ldots & n \\ s_1 & \ldots & s_n \end{pmatrix} = (i, s_i) \cdot \begin{pmatrix} 1 & \ldots & i & \ldots & k & \ldots & n \\ s_1 & \ldots & i & \ldots & s_i & \ldots & s_n \end{pmatrix} \tag{2}$$

for $i = 1, \ldots, n$ where for each i, k is such that $p_k = i$. Using (2) with i successively $1, \ldots, n$ we may write:

$$\begin{pmatrix} 1 & \ldots & n \\ p_1 & \ldots & p_n \end{pmatrix} = (1, t_1) \cdot \ldots \cdot (n, t_n) \cdot \begin{pmatrix} 1 & \ldots & n \\ 1 & \ldots & n \end{pmatrix}$$

$$= (1, t_1) \cdot \ldots \cdot (n, t_n)$$

for some $t_i \in N$. The desired transposition vector is $IP2=(t_1,\ldots,t_n)$.

2. Programming considerations

Depending on whether the input parameter IPAR is negative, zero, or positive, subroutine PERM performs the computation described above in 1a, 1b, or 1c, respectively.

If n is not positive, there is no computation and the error parameter IER is set to -1. If IPAR is not zero and IP1 is not a permutation vector on N, or if IPAR is zero and IP1 is not a transposition vector on N, computation is terminated as soon as the error is detected and IER is set to 1. If there is no error, IER is set to zero.

The computation of the resulting vector IP2 proceeds as follows.

a. If IPAR is negative, for each i, $i=1,\ldots,n$, IP2(IP1(i)) is set to i.

b. If IPAR is zero, initially IP2(i) is set to i for $i=1,\ldots,n$, and then for $i=1,\ldots,n$ in that order, IP2(i) and IP2(IP1(i)) are interchanged.

c. If IPAR is positive, initially IP1 is moved to IP2, and then the following two steps are repeated for i successively $1,\ldots,n$:

(1) Find the smallest $j \geq i$ such that IP2(j)=i.

(2) Set IP2(j) to IP2(i).

IP2 cannot have the same storage allocation as IP1. IP1 is returned unchanged.

```
C                                                                       PERM  10
C     ..................................................................PERM  20
C                                                                       PERM  30
C     SUBROUTINE PERM                                                   PERM  40
C                                                                       PERM  50
C     PURPOSE                                                           PERM  60
C        TO COMPUTE THE PERMUTATION VECTOR THAT IS INVERSE TO A GIVENPERM  70
C        PERMUTATION VECTOR, THE PERMUTATION VECTOR THAT IS EQUIVA-  PERM  80
C        LENT TO A GIVEN TRANSPOSITION VECTOR AND A TRANSPOSITION    PERM  90
C        VECTOR THAT IS EQUIVALENT TO A GIVEN PERMUTATION VECTOR.    PERM 100
C        (SEE THE GENERAL DISCUSSION FOR DEFINITIONS AND NOTATION.)  PERM 110
C                                                                    PERM 120
C                                                                    PERM 130
C     USAGE                                                          PERM 140
C        CALL PERM(IP1,IP2,N,IPAR,IER)                               PERM 150
C                                                                    PERM 160
C     DESCRIPTION OF PARAMETERS                                      PERM 170
C        IP1  - GIVEN PERMUTATION OR TRANSPOSITION VECTOR            PERM 180
C               (DIMENSION N)                                        PERM 190
C        IP2  - RESULTING PERMUTATION OR TRANSPOSITION VECTOR        PERM 200
C               (DIMENSION N)                                        PERM 210
C        N    - DIMENSION OF VECTORS IP1 AND IP2                     PERM 220
C        IPAR - INPUT PARAMETER                                      PERM 230
C               IPAR NEGATIVE - COMPUTE THE PERMUTATION VECTOR IP2   PERM 240
C                               THAT IS THE INVERSE OF THE PERMUTA-  PERM 250
C                               TION VECTOR IP1                      PERM 260
C               IPAR  =  ZERO - COMPUTE THE PERMUTATION VECTOR IP2   PERM 270
C                               THAT IS EQUIVALENT TO THE TRANSPOSI- PERM 280
C                               TION VECTOR IP1                      PERM 290
C               IPAR POSITIVE - COMPUTE A TRANSPOSITION VECTOR IP2   PERM 300
C                               THAT IS EQUIVALENT TO THE PERMUTATIONPERM 310
C                               VECTOR IP1                           PERM 320
C        IER  - RESULTING ERROR PARAMETER                            PERM 330
C               IER=-1  -  N IS NOT POSITIVE                         PERM 340
C               IER= 0  -  NO ERROR                                  PERM 350
C               IER= 1  -  IP1 IS EITHER NOT A PERMUTATION VECTOR OR PERM 360
C                          NOT A TRANSPOSITION VECTOR ON 1,...,N,    PERM 370
C                          DEPENDING ON WHETHER IPAR IS NON-ZERO OR  PERM 380
C                          ZERO, RESPECTIVELY                        PERM 390
C                                                                    PERM 400
C     REMARKS                                                        PERM 410
C        (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.               PERM 420
C        (2)  IF IER=1, THEN COMPUTATION HAS BEEN UNSUCCESSFUL DUE TOPERM 430
C             ERROR AND THE PARTIAL RESULTS FOUND IN IP2 ARE USELESS.PERM 440
C        (3)  IP2 CANNOT HAVE THE SAME STORAGE ALLOCATION AS IP1.    PERM 450
C                                                                    PERM 460
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PERM 470
C        NONE                                                        PERM 480
C                                                                    PERM 490
C     METHOD                                                         PERM 500
C        (1)  IPAR NEGATIVE - FOR EACH I, I=1,...,N, IP2(IP1(I)) IS  PERM 510
C                             SET TO I.                              PERM 520
C        (2)  IPAR  =  ZERO - INITIALLY IP2(I) IS SET TO I FOR       PERM 530
C                             I=1,...,N.  THEN, FOR I=1,...,N IN THATPERM 540
C                             ORDER, IP2(I) AND IP2(IP1(I)) ARE     PERM 550
C                             INTERCHANGED.                          PERM 560
C        (3)  IPAR POSITIVE - INITIALLY IP1 IS MOVED TO IP2.  THEN   PERM 570
C                             THE FOLLOWING TWO STEPS ARE REPEATED   PERM 580
C                             FOR I SUCCESSIVELY EQUAL TO 1,...,N.   PERM 590
C                             (A) FIND THE SMALLEST J GREATER THAN ORPERM 600
C                                 EQUAL TO I SUCH THAT IP2(J)=I.     PERM 610
C                             (B) SET IP2(J) TO IP2(I).              PERM 620
C     ..................................................................PERM 630
```

```
C                                                                       PERM 640
C     SUBROUTINE PERM(IP1,IP2,N,IPAR,IER)                               PERM 650
C                                                                       PERM 660
C                                                                       PERM 670
      DIMENSION IP1(1),IP2(1)                                           PERM 680
C                                                                       PERM 690
C        TEST DIMENSION                                                 PERM 700
      IF(N)19,19,1                                                      PERM 710
C                                                                       PERM 720
C        TEST IPAR TO DETERMINE WHETHER IP1 IS TO BE INTERPRETED AS     PERM 730
C        A PERMUTATION VECTOR OR AS A TRANSPOSITION VECTOR              PERM 740
C                                                                       PERM 750
    1 IF(IPAR)2,13,2                                                    PERM 760
C                                                                       PERM 770
C        CHECK THAT IP1 IS A PERMUTATION VECTOR AND COMPUTE IP1 INVERSE PERM 780
    2 DO 3 I=1,N                                                        PERM 790
    3 IP2(I)=0                                                          PERM 800
      DO 6 I=1,N                                                        PERM 810
      K=IP1(I)                                                          PERM 820
      IF(K-N)4,5,20                                                     PERM 830
    4 IF(K)20,20,5                                                      PERM 840
    5 IF(IP2(K))20,6,20                                                 PERM 850
    6 IP2(K)=I                                                          PERM 860
C                                                                       PERM 870
C        TEST IPAR FOR THE DESIRED OPERATION                            PERM 880
      IF(IPAR)12,7,7                                                    PERM 890
C                                                                       PERM 900
C        COMPUTE TRANSPOSITION VECTOR IP2 FOR PERMUTATION VECTOR IP1    PERM 910
    7 DO 8 I=1,N                                                        PERM 920
    8 IP2(I)=IP1(I)                                                     PERM 930
      NN=N-1                                                            PERM 940
      IF(NN)12,12,9                                                     PERM 950
    9 DO 11 I=1,NN                                                      PERM 960
      DO 10 J=1,NN                                                      PERM 970
      IF(IP2(J)-I)10,11,10                                              PERM 980
   10 CONTINUE                                                          PERM 990
      J=N                                                               PERM1000
   11 IP2(J)=IP2(I)                                                     PERM1010
C                                                                       PERM1020
C        NORMAL RETURN - NO ERROR                                       PERM1030
   12 IER=0                                                             PERM1040
      RETURN                                                            PERM1050
C                                                                       PERM1060
C        COMPUTE PERMUTATION VECTOR IP2 FOR TRANSPOSITION VECTOR IP1    PERM1070
   13 DO 14 I=1,N                                                       PERM1080
   14 IP2(I)=I                                                          PERM1090
      DO 18 I=1,N                                                       PERM1100
      K=IP1(I)                                                          PERM1110
      IF(K-I)15,18,16                                                   PERM1120
   15 IF(K)20,20,17                                                     PERM1130
   16 IF(N-K)20,17,17                                                   PERM1140
   17 J=IP2(I)                                                          PERM1150
      IP2(I)=IP2(K)                                                     PERM1160
      IP2(K)=J                                                          PERM1170
   18 CONTINUE                                                          PERM1180
      GO TO 12                                                          PERM1190
C                                                                       PERM1200
C        ERROR RETURN - N IS NOT POSITIVE                               PERM1210
   19 IER=-1                                                            PERM1220
      RETURN                                                            PERM1230
C                                                                       PERM1240
C        ERROR RETURN - IP1 IS EITHER NOT A PERMUTATION VECTOR          PERM1250
C                       OR NOT A TRANSPOSITION VECTOR                   PERM1260
   20 IER=1                                                             PERM1270
      RETURN                                                            PERM1280
      END
```

Subroutines TEAS and DTEAS

The problem is to assign a scalar value S to a given sequence $x_1, x_2, x_3, \ldots$. In case of convergence of the sequence, S should be the limit. The Epsilon algorithm possibly increases the speed of convergence and determines an approximate value FIN of S in terms of the given members of the finite sequence $x_1, x_2, x_3, \ldots, x_N$.

1. Theoretical background

The value FIN is calculated by means of the Epsilon algorithm (see Figure 16).

The first column (with superscript 1) is identical to the given sequence:

$$x_i^{(1)} = x_i \qquad (i = 1, 2, 3, \ldots)$$

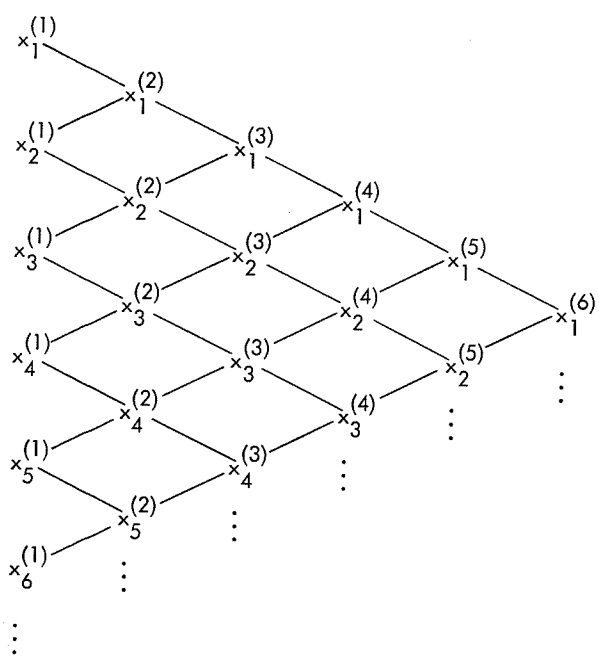


Figure 16. The complete Epsilon array

The values in the third column (with superscript 3) are derived from three consecutive members of the sequence; for example, $x_2^{(3)}$ is derived from $x_2^{(1)}$, $x_3^{(1)}$ and $x_4^{(1)}$. Generally, the column with superscript $2k + 1$ ($k = 1, 2, 3, \ldots$) contains quantities derived from $2k + 1$ consecutive members of the sequence. The intermediate values with even superscripts are auxiliary quantities.

The Epsilon algorithm is based on the following relationship:

$$x_i^{(k)} = x_{i+1}^{(k-2)} + \frac{1}{x_{i+1}^{(k-1)} - x_i^{(k-1)}} \qquad \begin{matrix}(k = 2, 3, 4, \ldots) \\ (i = 1, 2, 3, \ldots)\end{matrix} \qquad (1)$$

$$x_i^{(o)} = 0$$

With this fundamental relationship the complete Epsilon array shown in Figure 16 is built up from left to right.

2. Programming considerations

Subroutines TEAS and DTEAS generate only the first five columns of the complete Epsilon array; this E-2 transformation is used in an iterative way. Within one iteration the quantities in the upward diagonals are computed in a DO-loop by means of the algorithmic relationship (1). This process requires six auxiliary storage boxes (W1, W2, W3, W4, W5, W6).

The values in the fifth column $x_1^{(5)}, x_2^{(5)}, \ldots, x_{NEW-4}^{(5)}$ are successively stored in x(1), x(2), ... , x(NEW-4) as long as they are finite.

If a term $x_k^{(5)}$ in the fifth column becomes formally infinite, the values $x_1^{(5)}$ up to $x_k^{(5)}$ are cancelled and $x_{k+1}^{(5)}$ is stored in x(1), $x_{k+2}^{(5)}$ in x(2), and so on. Hence, after one iteration no component of the transformed vector x becomes infinite except possibly the last.

Consecutive equal terms in the column with superscript k(k=1, 2, 3) are reproduced in the column with superscript k+2. The basic mesh of the E-2 transformation is shown in Figure 17.

As a nonlinear algorithm the E-2 transformation is sensitive to roundoff errors; therefore special procedures are incorporated to avoid undue loss of accuracy:

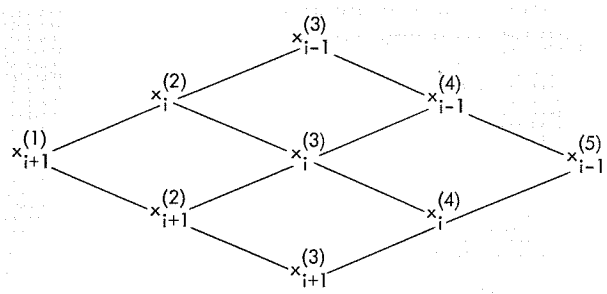


Figure 17. Basic mesh of the E-2 transformation

a. If there is an intolerable loss of significance in the calculation of a term in the third or fifth column, TEAS or DTEAS transfers out of the diagonal loop and uses for further calculation only those components of vector x, which have already been transformed.

b. If two consecutive quantities in the second column $x_i^{(2)}$ and $x_{i+1}^{(2)}$ (see Figure 17) are nearly equal, $x_i^{(3)}$ is large and ill-determined. Therefore, if:

$$\left.\begin{array}{l} \left|x_{i+1}^{(2)} - x_i^{(2)}\right| \leqq 10^{-4} \cdot \left|x_{i+1}^{(2)}\right| \\ \qquad\qquad \text{in single precision or} \\ \left|x_{i+1}^{(2)} - x_i^{(2)}\right| \leqq 10^{-12} \cdot \left|x_{i+1}^{(2)}\right| \\ \qquad\qquad \text{in double precision} \end{array}\right\} \quad (2)$$

subroutine TEAS or DTEAS calculates the iterated member of the sequence $x_{i-1}^{(5)}$ with a singular rule:

$$x_{i-1}^{(5)} = \frac{x_i^{(3)} \cdot W7}{1 + W7} \qquad (3)$$

where $$W7 = \frac{x_{i+1}^{(3)}}{x_i^{(3)} - x_{i+1}^{(3)}} + \frac{x_{i-1}^{(3)}}{x_i^{(3)} - x_{i-1}^{(3)}} - \frac{x_{i+1}^{(1)}}{x_i^{(3)} - x_{i+1}^{(1)}}$$

This requires two more auxiliary storage locations W7 and T.

c. If $x_i^{(2)}$ and $x_{i+1}^{(2)}$ are exactly equal, $x_i^{(3)}$ becomes infinite, and the singular rule for the Epsilon algorithm is simply:

$$x_{i-1}^{(5)} = x_{i+1}^{(3)} + x_{i-1}^{(3)} - x_{i+1}^{(1)} \qquad (4)$$

d. If three or more consecutive terms in the second column are equal, singular rule (4) cannot be applied; the corresponding iterated members of the sequence in the fifth column become formally infinite.

Iteration is terminated if one of the following conditions holds true:

a. Three times in succession the relative difference between neighboring transformed terms $X(I) = x_i$ is not greater than EPS; that is,

$$\left|x_i - x_{i+1}\right| \leqq \text{EPS} \cdot \left|x_{i+1}\right| \qquad (i = k, k+1, k+2) \qquad (5)$$

In case $\left|x_{i+1}\right| \leqq 1$, the relative difference (5) is replaced by the absolute

$$\left|x_i - x_{i+1}\right| \leqq \text{EPS} \qquad (i = k, k+1, k+2) \qquad (6)$$

b. The number of the transformed members of sequence (NEW-IMIN) is less than six.

In the first case x_{k+2} is returned as result value FIN, and IER = 0 as the error parameter. In the second case the values IER = 1 and FIN = x_1 are returned.

Some remarks are in order. First, the user must specify N--that is, the dimension of the input vector x--so that the character of the infinite sequence is recognizable. Second, subroutine TEAS or DTEAS works satisfactorily if the convergence of the given sequence is a geometric one; that is,

$$x_n - S = A \cdot \lambda^n + O(\lambda^n), \qquad |\lambda| < 1 \qquad (7)$$

For reference see:

(1) Shanks, Algorithm 215, CACM 1963, no. 11, p. 662.

(2) P. Wynn, "Singular Rules for certain nonlinear Algorithms," BIT vol. 3, 1963, pp. 175 - 195.

```
C                                                                       TEAS  10
C     ...................................................................TEAS  20
C                                                                       TEAS  30
C        SUBROUTINE TEAS                                                TEAS  40
C                                                                       TEAS  50
C        PURPOSE                                                        TEAS  60
C           CALCULATE THE LIMIT OF A GIVEN SEQUENCE BY MEANS OF THE     TEAS  70
C           EPSILON-ALGORITHM.                                          TEAS  80
C                                                                       TEAS  90
C        USAGE                                                          TEAS 100
C           CALL TEAS(X,N,FIN,EPS,IER)                                  TEAS 110
C                                                                       TEAS 120
C        DESCRIPTION OF PARAMETERS                                      TEAS 130
C           X      - VECTOR WHOSE COMPONENTS ARE TERMS OF THE GIVEN     TEAS 140
C                    SEQUENCE. ON RETURN THE COMPONENTS OF VECTOR X     TEAS 150
C                    ARE DESTROYED.                                     TEAS 160
C           N      - DIMENSION OF INPUT VECTOR X.                       TEAS 170
C           FIN    - RESULTANT SCALAR CONTAINING ON RETURN THE LIMIT    TEAS 180
C                    OF THE GIVEN SEQUENCE.                             TEAS 190
C           EPS    - AN INPUT VALUE, WHICH SPECIFIES THE UPPER BOUND    TEAS 200
C                    OF THE RELATIVE (ABSOLUTE) ERROR IF THE COMPONENTS TEAS 210
C                    OF X ARE ABSOLUTELY GREATER (LESS) THAN ONE.       TEAS 220
C                    CALCULATION IS TERMINATED AS SOON AS THREE TIMES INTEAS 230
C                    SUCCESSION THE RELATIVE (ABSOLUTE) DIFFERENCE      TEAS 240
C                    BETWEEN NEIGHBOURING TERMS IS NOT GREATER THAN EPS.TEAS 250
C           IER    - RESULTANT ERROR PARAMETER CODED IN THE FOLLOWING   TEAS 260
C                    FORM                                               TEAS 270
C                      IER=0  - NO ERROR                                TEAS 280
C                      IER=1  - REQUIRED ACCURACY NOT REACHED WITH      TEAS 290
C                               MAXIMAL NUMBER OF ITERATIONS            TEAS 300
C                      IER=-1 - INTEGER N IS LESS THAN TEN.             TEAS 310
C                                                                       TEAS 320
C        REMARKS                                                        TEAS 330
C           NO ACTION BESIDES ERROR MESSAGE IN CASE N LESS THAN TEN.    TEAS 340
C           THE CHARACTER OF THE GIVEN INFINITE SEQUENCE MUST BE        TEAS 350
C           RECOGNIZABLE BY THOSE N COMPONENTS OF THE INPUT VECTOR X.   TEAS 360
C                                                                       TEAS 370
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TEAS 380
C           NONE                                                        TEAS 390
C                                                                       TEAS 400
C        METHOD                                                         TEAS 410
C           THE CONVERGENCE OF THE GIVEN SEQUENCE IS ACCELERATED BY     TEAS 420
C           MEANS OF THE E(2)-TRANSFORMATION, USED IN AN ITERATIVE WAY. TEAS 430
C           FOR REFERENCE, SEE                                          TEAS 440
C           ALGORITHM 215,SHANKS, CACM 1963, NO. 11, PP. 662. AND       TEAS 450
C           P. WYNN, SINGULAR RULES FOR CERTAIN NON-LINEAR ALGORITHMS   TEAS 460
C           BIT VOL. 3, 1963, PP. 175-195.                              TEAS 470
C                                                                       TEAS 480
C     ...................................................................TEAS 490
C                                                                       TEAS 500
      SUBROUTINE TEAS(X,N,FIN,EPS,IER)                                  TEAS 510
C                                                                       TEAS 520
      DIMENSION X(1)                                                    TEAS 530
C                                                                       TEAS 540
C        TEST ON WRONG INPUT PARAMETER N                                TEAS 550
C                                                                       TEAS 560
      NEW=N                                                             TEAS 570
      IF(NEW-10)1,2,2                                                   TEAS 580
    1 IER=-1                                                            TEAS 590
      RETURN                                                            TEAS 600
C                                                                       TEAS 610
C        CALCULATE INITIAL VALUES FOR THE EPSILON ARRAY                 TEAS 620
C                                                                       TEAS 630
    2 ISW1=0                                                            TEAS 640
      ISW2=0                                                            TEAS 650
      W1=1.E38                                                          TEAS 660
      W7=X(4)-X(3)                                                      TEAS 670
      IF(W7)3,4,3                                                       TEAS 680
    3 W1=1./W7                                                          TEAS 690
C                                                                       TEAS 700
    4 W5=1.E38                                                          TEAS 710
      W7=X(2)-X(1)                                                      TEAS 720
      IF(W7)5,6,5                                                       TEAS 730
    5 W5=1./W7                                                          TEAS 740
C                                                                       TEAS 750
    6 W4=X(3)-X(2)                                                      TEAS 760
      IF(W4)9,7,9                                                       TEAS 770
    7 W4=1.E38                                                          TEAS 780
      T=X(2)                                                            TEAS 790
      W2=X(3)                                                           TEAS 800
    8 W3=1.E38                                                          TEAS 810
      GO TO 17                                                          TEAS 820
C                                                                       TEAS 830
    9 W4=1./W4                                                          TEAS 840
C                                                                       TEAS 850
      T=1.E38                                                           TEAS 860
      W7=W4-W5                                                          TEAS 870
      IF(W7)10,11,10                                                    TEAS 880
   10 T=X(2)+1./W7                                                      TEAS 890
C                                                                       TEAS 900
   11 W2=W1-W4                                                          TEAS 910
      IF(W2)15,12,15                                                    TEAS 920
   12 W2=1.E38                                                          TEAS 930
      IF(T-1.E38)13,14,14                                               TEAS 940
   13 ISW2=1                                                            TEAS 950
   14 W3=W4                                                             TEAS 960
      GO TO 17                                                          TEAS 970
C                                                                       TEAS 980
   15 W2=X(3)+1./W2                                                     TEAS 990
      W7=W2-T                                                           TEAS1000
      IF(W7)16,8,16                                                     TEAS1010
   16 W3=W4+1./W7                                                       TEAS1020
C                                                                       TEAS1030
   17 ISW1=ISW2                                                         TEAS1040
      ISW2=0                                                            TEAS1050
      IMIN=4                                                            TEAS1060
C                                                                       TEAS1070
C        CALCULATE DIAGONALS OF THE EPSILON ARRAY IN A DO-LOOP          TEAS1080
C                                                                       TEAS1090
      DO 40 I=5,NEW                                                     TEAS1100
      IAUS=I-IMIN                                                       TEAS1110
      W4=1.E38                                                          TEAS1120
      W5=X(I-1)                                                         TEAS1130
      W7=X(I)-X(I-1)                                                    TEAS1140
      IF(W7)18,24,18                                                    TEAS1150
   18 W4=1./W7                                                          TEAS1160
C                                                                       TEAS1170
      IF(W1-1.E38)19,25,25                                              TEAS1180
   19 W6=W4-W1                                                          TEAS1190
C                                                                       TEAS1200
C        TEST FOR NECESSITY OF A SINGULAR RULE                          TEAS1210
C                                                                       TEAS1220
      IF(ABS(W6)-ABS(W4)*1.E-4)20,20,22                                 TEAS1230
   20 ISW2=1                                                            TEAS1240
      IF(W6)22,21,22                                                    TEAS1250
   21 W5=1.E38                                                          TEAS1260
      W6=W1                                                             TEAS1270
      IF(W2-1.E38)28,26,26                                              TEAS1280
   22 W5=X(I-1)+1./W6                                                   TEAS1290
C                                                                       TEAS1300
C        FIRST TEST FOR LOSS OF SIGNIFICANCE                            TEAS1310
C                                                                       TEAS1320
      IF(ABS(W5)-ABS(X(I-1))*1.E-5)23,24,24                             TEAS1330
   23 IF(W5)36,24,36                                                    TEAS1340
C                                                                       TEAS1350
   24 W7=W5-W2                                                          TEAS1360
      IF(W7)27,25,27                                                    TEAS1370
   25 W6=1.E38                                                          TEAS1380
   26 ISW2=0                                                            TEAS1390
      X(IAUS)=W2                                                        TEAS1400
      GO TO 37                                                          TEAS1410
   27 W6=W1+1./W7                                                       TEAS1420
   28 IF(ISW1-1)33,29,29                                                TEAS1430
C                                                                       TEAS1440
C        CALCULATE X(IAUS) WITH HELP OF SINGULAR RULE                   TEAS1450
C                                                                       TEAS1460
   29 IF(W2-1.E38)30,32,32                                              TEAS1470
   30 W7=W5/(W2-W5)+T/(W2-T)+X(I-2)/(X(I-2)-W2)                         TEAS1480
      IF(1.+W7)31,38,31                                                 TEAS1490
   31 X(IAUS)=W7*W2/(1.+W7)                                             TEAS1500
      GO TO 39                                                          TEAS1510
C                                                                       TEAS1520
   32 X(IAUS)=W5+T-X(I-2)                                               TEAS1530
      GO TO 39                                                          TEAS1540
C                                                                       TEAS1550
   33 W7=W6-W3                                                          TEAS1560
      IF(W7)34,38,34                                                    TEAS1570
   34 X(IAUS)=W2+1./W7                                                  TEAS1580
C                                                                       TEAS1590
C        SECOND TEST FOR LOSS OF SIGNIFICANCE                           TEAS1600
C                                                                       TEAS1610
      IF(ABS(X(IAUS))-ABS(W2)*1.E-5)35,37,37                            TEAS1620
   35 IF(X(IAUS))36,37,36                                               TEAS1630
C                                                                       TEAS1640
   36 NEW=IAUS-1                                                        TEAS1650
      ISW2=0                                                            TEAS1660
      GO TO 41                                                          TEAS1670
C                                                                       TEAS1680
   37 IF(W2-1.E38)39,38,38                                              TEAS1690
   38 X(IAUS)=1.E38                                                     TEAS1700
      IMIN=I                                                            TEAS1710
C                                                                       TEAS1720
   39 W1=W4                                                             TEAS1730
      T=W2                                                              TEAS1740
      W2=W5                                                             TEAS1750
      W3=W6                                                             TEAS1760
      ISW1=ISW2                                                         TEAS1770
   40 ISW2=0                                                            TEAS1780
C                                                                       TEAS1790
      NEW=NEW-IMIN                                                      TEAS1800
C                                                                       TEAS1810
C        TEST FOR ACCURACY                                              TEAS1820
C                                                                       TEAS1830
   41 IEND=NEW-1                                                        TEAS1840
      DO 47 I=1,IEND                                                    TEAS1850
      W1=ABS(X(I)-X(I+1))                                               TEAS1860
      W2=ABS(X(I+1))                                                    TEAS1870
      IF(W1-EPS)44,44,42                                                TEAS1880
   42 IF(W2-1.)46,46,43                                                 TEAS1890
   43 IF(W1-EPS*W2)44,44,46                                             TEAS1900
   44 ISW2=ISW2+1                                                       TEAS1910
      IF(3-ISW2)45,45,47                                                TEAS1920
   45 FIN=X(I)                                                          TEAS1930
      IER=0                                                             TEAS1940
      RETURN                                                            TEAS1950
C                                                                       TEAS1960
   46 ISW2=0                                                            TEAS1970
   47 CONTINUE                                                          TEAS1980
C                                                                       TEAS1990
      IF(NEW-6)48,2,2                                                   TEAS2000
   48 FIN=X(NEW)                                                        TEAS2010
      IER=1                                                             TEAS2020
      RETURN                                                            TEAS2030
      END                                                               TEAS2040
```

```
C
C     ..................................................................
C
C        SUBROUTINE DTEAS
C
C        PURPOSE
C           CALCULATE THE LIMIT OF A GIVEN SEQUENCE BY MEANS OF THE
C           EPSILON-ALGORITHM.
C
C        USAGE
C           CALL DTEAS(X,N,FIN,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      - DOUBLE PRECISION VECTOR WHOSE COMPONENTS ARE TERMS
C                    OF THE GIVEN SEQUENCE. ON RETURN THE COMPONENTS OF
C                    VECTOR X ARE DESTROYED.
C           N      - DIMENSION OF INPUT VECTOR X.
C           FIN    - RESULTANT SCALAR IN DOUBLE PRECISION CONTAINING ON
C                    RETURN THE LIMIT OF THE GIVEN SEQUENCE.
C           EPS    - SINGLE PRECISION INPUT VALUE, WHICH SPECIFIES THE
C                    UPPER BOUND OF THE RELATIVE (ABSOLUTE) ERROR IF THE
C                    COMPONENTS OF X ARE ABSOLUTELY GREATER (LESS) THAN
C                    ONE.
C                    CALCULATION IS TERMINATED AS SOON AS THREE TIMES IN
C                    SUCCESSION THE RELATIVE (ABSOLUTE) DIFFERENCE
C                    BETWEEN NEIGHBOURING TERMS IS NOT GREATER THAN EPS.
C           IER    - RESULTANT ERROR PARAMETER CODED IN THE FOLLOWING
C                    FORM
C                     IER=0  - NO ERROR
C                     IER=1  - REQUIRED ACCURACY NOT REACHED WITH
C                              MAXIMAL NUMBER OF ITERATIONS
C                     IER=-1 - INTEGER N IS LESS THAN TEN.
C
C        REMARKS
C           NO ACTION BESIDES ERROR MESSAGE IN CASE N LESS THAN TEN.
C           THE CHARACTER OF THE GIVEN INFINITE SEQUENCE MUST BE
C           RECOGNIZABLE BY THOSE N COMPONENTS OF THE INPUT VECTOR X.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           THE CONVERGENCE OF THE GIVEN SEQUENCE IS ACCELERATED BY
C           MEANS OF THE E(2)-TRANSFORMATION, USED IN AN ITERATIVE WAY.
C           FOR REFERENCE, SEE
C           ALGORITHM 215,SHANKS, CACM 1963, NO. 11, PP. 662. AND
C           P. WYNN, SINGULAR RULES FOR CERTAIN NON-LINEAR ALGORITHMS
C           BIT VOL. 3, 1963, PP. 175-195.
C
C     ..................................................................
C
      SUBROUTINE DTEAS(X,N,FIN,EPS,IER)
C
      DIMENSION X(1)
      DOUBLE PRECISION X,FIN,W1,W2,W3,W4,W5,W6,W7,T
C
C        TEST ON WRONG INPUT PARAMETER N
C
      NEW=N
      IF(NEW-10)1,2,2
    1 IER=-1
      RETURN
C
C        CALCULATE INITIAL VALUES FOR THE EPSILON ARRAY
C
    2 ISW1=0
      ISW2=0
      W1=1.D38
      W7=X(4)-X(3)
      IF(W7)3,4,3
    3 W1=1.D0/W7
C
    4 W5=1.D38
      W7=X(2)-X(1)
      IF(W7)5,6,5
    5 W5=1.D0/W7
C
    6 W4=X(3)-X(2)
      IF(W4)9,7,9
    7 W4=1.D38
      T=X(2)
      W2=X(3)
    8 W3=1.D38
      GO TO 17
C
    9 W4=1.D0/W4
C
      T=1.D38
      W7=W4-W5
      IF(W7)10,11,10
   10 T=X(2)+1.D0/W7
C
   11 W2=W1-W4
      IF(W2)15,12,15
   12 W2=1.D38
      IF(T-1.D38)13,14,14
   13 ISW2=1
   14 W3=W4
      GO TO 17
C
   15 W2=X(3)+1.D0/W2
      W7=W2-T
      IF(W7)16,8,16
   16 W3=W4+1.D0/W7
C
   17 ISW1=ISW2
      ISW2=0
      IMIN=4
C
C        CALCULATE DIAGONALS OF THE EPSILON ARRAY IN A DO-LOOP
C
      DO 40 I=5,NEW
      IAUS=I-IMIN
      W4=1.D38
      W5=X(I-1)
      W7=X(I)-X(I-1)
      IF(W7)18,24,18
   18 W4=1.D0/W7
C
      IF(W1-1.D38)19,25,25
   19 W6=W4-W1
C
C        TEST FOR NECESSITY OF A SINGULAR RULE
C
      IF(DABS(W6)-DABS(W4)*1.D-12)20,20,22
   20 ISW2=1
      IF(W6)22,21,22
   21 W5=1.D38
      W6=W1
      IF(W2-1.D38)28,26,26
   22 W5=X(I-1)+1.D0/W6
C
C        FIRST TEST FOR LOSS OF SIGNIFICANCE
C
      IF(DABS(W5)-DABS(X(I-1))*1.D-10)23,24,24
   23 IF(W5)36,24,36
C
   24 W7=W5-W2
      IF(W7)27,25,27
   25 W6=1.D38
   26 ISW2=0
      X(IAUS)=W2
      GO TO 37
   27 W6=W1+1.D0/W7
   28 IF(ISW1-1)33,29,29
C
C        CALCULATE X(IAUS) WITH HELP OF SINGULAR RULE
C
   29 IF(W2-1.D38)30,32,32
   30 W7=W5/(W2-W5)+T/(W2-T)+X(I-2)/(X(I-2)-W2)
      IF(1.D0+W7)31,38,31
   31 X(IAUS)=W7*W2/(1.D0+W7)
      GO TO 39
C
   32 X(IAUS)=W5+T-X(I-2)
      GO TO 39
C
   33 W7=W6-W3
      IF(W7)34,38,34
   34 X(IAUS)=W2+1.D0/W7
C
C        SECOND TEST FOR LOSS OF SIGNIFICANCE
C
      IF(DABS(X(IAUS))-DABS(W2)*1.D-10)35,37,37
   35 IF(X(IAUS))36,37,36
C
   36 NEW=IAUS-1
      ISW2=0
      GO TO 41
C
   37 IF(W2-1.D38)39,38,38
   38 X(IAUS)=1.D38
      IMIN=I
C
   39 W1=W4
      T=W2
      W2=W5
      W3=W6
      ISW1=ISW2
   40 ISW2=0
C
      NEW=NEW-IMIN
C
C        TEST FOR ACCURACY
C
   41 IEND=NEW-1
      DO 47 I=1,IEND
      HE1=DABS(X(I)-X(I+1))
      HE2=DABS(X(I+1))
      IF(HE1-EPS)46,44,42
   42 IF(HE2-1.)46,46,43
   43 IF(HE1-EPS*HE2)44,44,46
   44 ISW2=ISW2+1
      IF(3-ISW2)45,45,47
   45 FIN=X(I)
      IER=0
      RETURN
C
   46 ISW2=0
   47 CONTINUE
C
      IF(NEW-6)48,2,2
   48 FIN=X(NEW)
      IER=1
      RETURN
      END
```

Subroutines TEUL and DTEUL

These subroutines calculate the sum value of the series $\sum_{k=1}^{\infty} F_k$ by means of a refined Euler transformation.

The following abbreviations are used:

$$\Delta^0 F_i = F_i$$

$$\Delta^1 F_i = \frac{1}{2}\left[F_i + F_{i+1}\right]$$

$$\Delta^2 F_i = \frac{1}{2}\left[\Delta F_i + \Delta F_{i+1}\right] = \frac{1}{4}\left[F_i + 2F_{i+1} + F_{i+2}\right]$$

$$\Delta^3 F_i = \frac{1}{2}\left[\Delta^2 F_i + \Delta^2 F_{i+1}\right] = \frac{1}{8}\left[F_i + 3F_{i+1} + 3F_{i+2} + F_{i+3}\right]$$

Generally:

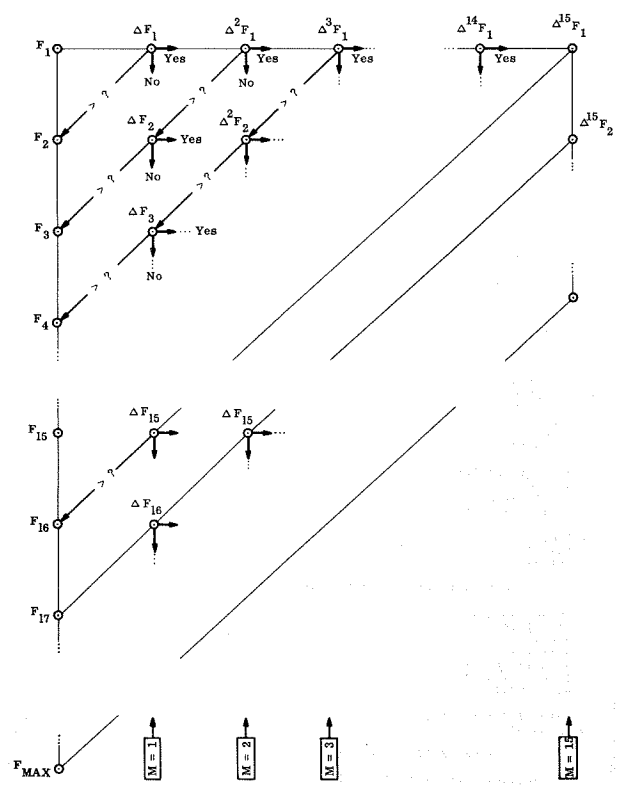
$$\Delta^m F_i = \frac{1}{2}\left[\Delta^{m-1} F_i + \Delta^{m-1} F_{i+1}\right] = \frac{1}{2^m}\left[F_i + \binom{m}{1}F_{i+1} + \binom{m}{2}F_{i+2} + \ldots + \binom{m}{m-1}F_{i+m-1} + \binom{m}{m}F_{i+m}\right]$$

Euler transformation means that the given infinite series will be transformed as follows:

$$\sum_{k=1}^{\infty} F_k = \frac{1}{2}\left[F_1 + \Delta F_1 + \Delta^2 F_1 + \Delta^3 F_1 + \Delta^4 F_1 + \ldots\right]$$

The transformed series will converge whenever the given series does, and to the same sum. It may even converge when the parent series does not; in this case the sum is called the Euler sum of the parent series.

A refined Euler transformation is explained by means of the following diagram.



The computation is done in this way. First the absolute value $|\Delta F_1|$ is compared with $|F_2|$ (see diagram). In case $|\Delta F_1| < |F_2|$, the value $\frac{1}{2}(\Delta F_1)$ is added to the initial value $\frac{1}{2}F_1$ (the beginning of the Euler transformation). Then $|\Delta^2 F_1|$ is compared with $|\Delta F_2|$ (see "YES" in diagram).

In case $|\Delta F_1| \geq |F_2|$, the value ΔF_1 is added to $\frac{1}{2}F_1$ and, as next $|\Delta F_2|$, is compared with $|F_3|$ (see "NO"). This means that F_1 is added unchanged, and the Euler transformation is delayed for one term, and so forth.

If in this sequence of comparisons $|\Delta^i F_k| < |\Delta^{i-1} F_{k+1}|$ is generally true, the value $\frac{1}{2}(\Delta^i F_k)$ is added to the current sum in the sense of the Euler transformation. If it is not true, the value $\Delta^i F_k$ is added and the Euler transformation is delayed for one term--that is, the terms F_1, F_2, ... , F_k are added unchanged, and the transformation actually starts with term F_{k+1}.

The highest term in the scheme of calculation may be $\Delta^{15} F_i$ ($i = 1, 2, 3, \ldots, MAX-15$), in case $MAX > 15$. Hence a maximum of fourteen terms of the series may be transformed in succession by Euler. If the procedure is not terminated up to this point, the index of the transformed terms is raised by 1, and a further unchanged term is added.

The last term of the parent series which may be used by the computation is F_{MAX}.

The procedure is terminated if with five successive terms of the series the relative change of the sum value is less than EPS, or if MAX terms of the series were used for calculation.

For reference see:

(1.) F. B. Hildebrand, Introduction to Numerical Analysis, McGraw-Hill, New York/Toronto/London, 1956, pp. 155 - 160.

(2.) P. Naur, "Report on the Algorithmic Language ALGOL 60", CACM, vol. 3, no. 5 (1960), pp. 311.


```
C                                                                       TEUL  10
C     ...................................................................TEUL  20
C                                                                       TEUL  30
C        SUBROUTINE TEUL                                                TEUL  40
C                                                                       TEUL  50
C        PURPOSE                                                        TEUL  60
C           COMPUTE THE SUM OF FCT(K) FOR K FROM ONE UP TO INFINITY.    TEUL  70
C                                                                       TEUL  80
C        USAGE                                                          TEUL  90
C           CALL TEUL(FCT,SUM,MAX,EPS,IER)                              TEUL 100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.               TEUL 110
C                                                                       TEUL 120
C        DESCRIPTION OF PARAMETERS                                      TEUL 130
C           FCT     - NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.     TEUL 140
C                     IT COMPUTES THE K-TH TERM OF THE SERIES TO ANY    TEUL 150
C                     GIVEN INDEX K.                                    TEUL 160
C           SUM     - RESULTANT VALUE CONTAINING ON RETURN THE SUM OF   TEUL 170
C                     THE GIVEN SERIES.                                 TEUL 180
C           MAX     - INPUT VALUE, WHICH SPECIFIES THE MAXIMAL NUMBER   TEUL 190
C                     OF TERMS OF THE SERIES THAT ARE RESPECTED.        TEUL 200
C           EPS     - INPUT VALUE, WHICH SPECIFIES THE UPPER BOUND OF   TEUL 210
C                     THE RELATIVE ERROR.                               TEUL 220
C                     SUMMATION IS STOPPED AS SOON AS FIVE TIMES IN     TEUL 230
C                     SUCCESSION THE ABSOLUTE VALUE OF THE TERMS OF THE TEUL 240
C                     TRANSFORMED SERIES ARE FOUND TO BE LESS THAN      TEUL 250
C                     EPS*(ABSOLUTE VALUE OF CURRENT SUM).              TEUL 260
C           IER     - RESULTANT ERROR PARAMETER CODED IN THE FOLLOWING  TEUL 270
C                     FORM                                              TEUL 280
C                      IER=0  - NO ERROR                                TEUL 290
C                      IER=1  - REQUIRED ACCURACY NOT REACHED WITH      TEUL 300
C                               MAXIMAL NUMBER OF TERMS                 TEUL 310
C                      IER=-1 - THE INTEGER MAX IS LESS THAN ONE.       TEUL 320
C                                                                       TEUL 330
C        REMARKS                                                        TEUL 340
C           NO ACTION BESIDES ERROR MESSAGE IN CASE MAX LESS THAN ONE.  TEUL 350
C                                                                       TEUL 360
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  TEUL 370
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(K) MUST BE FURNISHED   TEUL 380
C           BY THE USER.                                                TEUL 390
C                                                                       TEUL 400
C        METHOD                                                         TEUL 410
C           EVALUATION IS DONE BY MEANS OF A SUITABLY REFINED EULER     TEUL 420
C           TRANSFORMATION. FOR REFERENCE, SEE                          TEUL 430
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,         TEUL 440
C           MCGRAW/HILL, NEW YORK/TORONTO/LONDON, 1956, PP.155-160, AND TEUL 450
C           P. NAUR, REPORT ON THE ALGORITHMIC LANGUAGE ALGOL 60,       TEUL 460
C           CACM, VOL.3, ISS.5 (1960), PP.311.                          TEUL 470
C                                                                       TEUL 480
C     ...................................................................TEUL 490
C                                                                       TEUL 500
      SUBROUTINE TEUL (FCT,SUM,MAX,EPS,IER)                             TEUL 510
C                                                                       TEUL 520
      DIMENSION Y(15)                                                   TEUL 530
C                                                                       TEUL 540
C        TEST ON WRONG INPUT PARAMETER MAX                              TEUL 550
C                                                                       TEUL 560
      IF(MAX)1,1,2                                                      TEUL 570
    1 IER=-1                                                            TEUL 580
      GOTO 12                                                           TEUL 590
C                                                                       TEUL 600
C        INITIALIZE EULER TRANSFORMATION                                TEUL 610
C                                                                       TEUL 620
    2 IER=1                                                             TEUL 630
      I=1                                                               TEUL 640
      M=1                                                               TEUL 650
      N=1                                                               TEUL 660
      Y(1)=FCT(N)                                                       TEUL 670
      SUM=Y(1)*.5                                                       TEUL 680
C                                                                       TEUL 690
C        START EULER-LOOP                                               TEUL 700
C                                                                       TEUL 710
    3 J=0                                                               TEUL 720
    4 I=I+1                                                             TEUL 730
      IF(I-MAX)5,5,12                                                   TEUL 740
    5 N=I                                                               TEUL 750
      AMN=FCT(N)                                                        TEUL 760
      DO 6 K=1,M                                                        TEUL 770
      AMP=(AMN+Y(K))*.5                                                 TEUL 780
      Y(K)=AMN                                                          TEUL 790
    6 AMN=AMP                                                           TEUL 800
C                                                                       TEUL 810
C        CHECK EULER TRANSFORMATION                                     TEUL 820
C                                                                       TEUL 830
      IF(ABS(AMN)-ABS(Y(M)))7,9,9                                       TEUL 840
    7 IF(M-15)8,9,9                                                     TEUL 850
    8 M=M+1                                                             TEUL 860
      Y(M)=AMN                                                          TEUL 870
      AMN=.5*AMN                                                        TEUL 880
C                                                                       TEUL 890
C        UPDATE SUM                                                     TEUL 900
C                                                                       TEUL 910
    9 SUM=SUM+AMN                                                       TEUL 920
      IF(ABS(AMN)-EPS*ABS(SUM))10,10,3                                  TEUL 930
C                                                                       TEUL 940
C        TEST END OF PROCEDURE                                          TEUL 950
C                                                                       TEUL 960
   10 J=J+1                                                             TEUL 970
      IF(J-5)4,11,11                                                    TEUL 980
   11 IER=0                                                             TEUL 990
   12 RETURN                                                            TEUL1000
      END                                                               TEUL1010
```

```
C
C
C
```

Subroutines ALI and DALI

These subroutines perform an Aitken-Lagrange interpolation. Given the table x_i, y_i ($i = 1, 2, \ldots, n$) of argument and function values stored in the two one-dimensional storage arrays ARG and VAL, the problem is to interpolate function value Y at the given point X. It is assumed that the table represents a single-valued function $y = y(x)$ which is suitable for approximation by a polynomial+, and that $|x_i - X| \geq x_j - X|$ for $i > j$.++

Then, using the following formulas:

$$y_{1,2} = \frac{1}{x_2 - x_1} \cdot \begin{vmatrix} y_1 & x_1 - X \\ y_2 & x_2 - X \end{vmatrix} ;$$

$$y_{1,3} = \frac{1}{x_3 - x_1} \cdot \begin{vmatrix} y_1 & x_1 - X \\ y_3 & x_3 - X \end{vmatrix} ,$$

$$y_{1,2,3} = \frac{1}{x_3 - x_2} \cdot \begin{vmatrix} y_{1,2} & x_2 - X \\ y_{1,3} & x_3 - X \end{vmatrix} ;$$

$$y_{1,4} = \frac{1}{x_4 - x_1} \cdot \begin{vmatrix} y_1 & x_1 - X \\ y_4 & x_4 - X \end{vmatrix} ,$$

$$y_{1,2,4} = \frac{1}{x_4 - x_2} \cdot \begin{vmatrix} y_{1,2} & x_2 - X \\ y_{1,4} & x_4 - X \end{vmatrix} ,$$

$$y_{1,2,3,4} = \frac{1}{x_4 - x_3} \cdot \begin{vmatrix} y_{1,2,3} & x_3 - X \\ y_{1,2,4} & x_4 - X \end{vmatrix} ;$$

$$\cdots\cdots\cdots\cdots\cdots\cdots ,$$

it is possible to generate by row the triangular Aitken scheme shown in Figure 18.

$$\begin{array}{c|c||c|c|c|c|c}
x_1 & y_1 & & & & & \\
x_2 & y_2 & y_{1,2} & & & & \\
x_3 & y_3 & y_{1,3} & y_{1,2,3} & & & \\
x_4 & y_4 & y_{1,4} & y_{1,2,4} & y_{1,2,3,4} & & \\
\vdots & \vdots & \vdots & \vdots & & & \\
x_n & y_n & y_{1,n} & y_{1,2,n} & y_{1,2,3,n} & \ldots & y_{1,2,3,\ldots,n}
\end{array}$$

Figure 18. Aitken's scheme for Lagrange interpolation

All result values of row j can be stored in VAL(J). Thus, it is possible to generate the downward diagonal of Aitken's scheme in storage array VAL, using the machine-processable formula:

$$\text{VAL(J)} = \frac{\text{VAL(I)*(X-ARG(J))} - \text{VAL(J)*(X-ARG(I))}}{\text{ARG(I)} - \text{ARG(J)}}$$

$$(I = 1, 2, \ldots, J-1) \qquad (1)$$

for $J = 2, 3, \ldots,$ NDIM.

The procedure stops under the following conditions: if the difference of two successive values in the downward diagonal is absolutely less than a given tolerance ϵ (test starts at step $J = 3$), or if the absolute value of this difference stops diminishing, thus showing the influence of rounding errors (test starts at step $J = 5$, or for double-precision, $J = 8$), or if the procedure has worked through the whole triangular Aitken scheme, or if it discovers two table points with identical arguments.

Finally, Y is set equal to VAL(J) in case of monotonically decreasing absolute differences, or equal to VAL(J-1) if these differences start oscillating.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis, McGraw-Hill, New York/Toronto/London, 1956, pp. 49-50.

+This order in table x_i, y_i, ($i = 1, 2, \ldots, n$) could be generated by subroutines ATSG, ATSM, ATSE, DATSG, DATSM, DATSE.

++This means $y(x)$ should be closely approximated by a polynomial of low degree which has the table points with arguments next to x as nodes.

```
C
C     ..................................................................
C
C        SUBROUTINE ALI
C
C        PURPOSE
C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT AND FUNCTION
C           VALUES.
C
C        USAGE
C           CALL ALI (X,ARG,VAL,Y,NDIM,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      - THE ARGUMENT VALUE SPECIFIED BY INPUT.
C           ARG    - THE INPUT VECTOR (DIMENSION NDIM) OF ARGUMENT
C                    VALUES OF THE TABLE (NOT DESTROYED).
C           VAL    - THE INPUT VECTOR (DIMENSION NDIM) OF FUNCTION
C                    VALUES OF THE TABLE (DESTROYED).
C           Y      - THE RESULTING INTERPOLATED FUNCTION VALUE.
C           NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
C                    POINTS IN TABLE (ARG,VAL).
C           EPS    - AN INPUT CONSTANT WHICH IS USED AS UPPER BOUND
C                    FOR THE ABSOLUTE ERROR.
C           IER    - A RESULTING ERROR PARAMETER.
C
C        REMARKS
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C               SUBROUTINES ATSG, ATSM OR ATSE COULD BE USED IN A
C               PREVIOUS STAGE.
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C               THAN 1.
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C               (NDIM-1) STEPS. FURTHER IT IS TERMINATED IF THE
C               PROCEDURE DISCOVERS TWO ARGUMENT VALUES IN VECTOR ARG
C               WHICH ARE IDENTICAL. DEPENDENT ON THESE FOUR CASES,
C               ERROR PARAMETER IER IS CODED IN THE FOLLOWING FORM
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                        ACCURACY (NO ERROR).
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                        NDIM IS LESS THAN 3, OR THE REQUIRED ACCURACY
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                        TABLE. NDIM SHOULD BE INCREASED.
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                        IN VECTOR ARG WHICH ARE IDENTICAL.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           INTERPOLATION IS DONE BY MEANS OF AITKENS SCHEME OF
C           LAGRANGE INTERPOLATION. ON RETURN Y CONTAINS AN INTERPOLATED
C           FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK
C           (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.49-50.
C
C     ..................................................................
C
      SUBROUTINE ALI(X,ARG,VAL,Y,NDIM,EPS,IER)
C
C
      DIMENSION ARG(1),VAL(1)
      IER=2
      DELT2=0.
      IF(NDIM-1)9,7,1
C
C     START OF AITKEN-LOOP
    1 DO 6 J=2,NDIM
      DELT1=DELT2
      IEND=J-1
      DO 2 I=1,IEND
      H=ARG(I)-ARG(J)
      IF(H)2,13,2
    2 VAL(J)=(VAL(I)*(X-ARG(J))-VAL(J)*(X-ARG(I)))/H
      DELT2=ABS(VAL(J)-VAL(IEND))
      IF(J-2)6,6,3
    3 IF(DELT2-EPS)10,10,4
    4 IF(J-5)6,5,5
    5 IF(DELT2-DELT1)6,11,11
    6 CONTINUE
C     END OF AITKEN-LOOP
C
    7 J=NDIM
    8 Y=VAL(J)
    9 RETURN
C
C     THERE IS SUFFICIENT ACCURACY WITHIN NDIM-1 ITERATION STEPS
   10 IER=0
      GOTO 8
C
C     TEST VALUE DELT2 STARTS OSCILLATING
   11 IER=1
   12 J=IEND
      GOTO 8
C
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG
   13 IER=3
      GOTO 12
      END
```

```
C
C     ..................................................................
C
      SUBROUTINE DALI
C
C     PURPOSE
C        TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
C        X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT AND FUNCTION
C        VALUES.
C
C     USAGE
C        CALL DALI (X,ARG,VAL,Y,NDIM,EPS,IER)
C
C     DESCRIPTION OF PARAMETERS
C        X      - DOUBLE PRECISION ARGUMENT VALUE SPECIFIED BY INPUT.
C        ARG    - DOUBLE PRECISION INPUT VECTOR (DIMENSION NDIM) OF
C                 ARGUMENT VALUES OF THE TABLE (NOT DESTROYED).
C        VAL    - DOUBLE PRECISION INPUT VECTOR (DIMENSION NDIM) OF
C                 FUNCTION VALUES OF THE TABLE (DESTROYED).
C        Y      - RESULTING INTERPOLATED DOUBLE PRECISION FUNCTION
C                 VALUE.
C        NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
C                 POINTS IN TABLE (ARG,VAL).
C        EPS    - SINGLE PRECISION INPUT CONSTANT WHICH IS USED AS
C                 UPPER BOUND FOR THE ABSOLUTE ERROR.
C                 FOR THE ABSOLUTE ERROR.
C        IER    - A RESULTING ERROR PARAMETER.
C
C     REMARKS
C        (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C            FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C            DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C            SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C            SUBROUTINES DATSG, DATSM OR DATSE COULD BE USED IN A
C            PREVIOUS STAGE.
C        (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C            THAN 1.
C        (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C            BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C            ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C            VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C            (NDIM-1) STEPS. FURTHER IT IS TERMINATED IF THE
C            PROCEDURE DISCOVERS TWO ARGUMENT VALUES IN VECTOR ARG
C            WHICH ARE IDENTICAL. DEPENDENT ON THESE FOUR CASES,
C            ERROR PARAMETER IER IS CODED IN THE FOLLOWING FORM
C             IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                     ACCURACY (NO ERROR).
C             IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                     ACCURACY BECAUSE OF ROUNDING ERRORS.
C             IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                     NDIM IS LESS THAN 3, OR THE REQUIRED ACCURACY
C                     COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                     TABLE. NDIM SHOULD BE INCREASED.
C             IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                     IN VECTOR ARG WHICH ARE IDENTICAL.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        NONE
C
C     METHOD
C        INTERPOLATION IS DONE BY MEANS OF AITKENS SCHEME OF
C        LAGRANGE INTERPOLATION. ON RETURN Y CONTAINS AN INTERPOLATED
C        FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK
C        (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE
C        F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
C        MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.49-50.
C
C     ..................................................................
C
      SUBROUTINE DALI(X,ARG,VAL,Y,NDIM,EPS,IER)
C
C
      DIMENSION ARG(1),VAL(1)
      DOUBLE PRECISION ARG,VAL,X,Y,H
      IER=2
      DELT2=0.
      IF(NDIM-1)9,7,1
C
C     START OF AITKEN-LOOP
    1 DO 6 J=2,NDIM
      DELT1=DELT2
      IEND=J-1
      DO 2 I=1,IEND
      H=ARG(I)-ARG(J)
      IF(H)2,13,2
    2 VAL(J)=(VAL(I)*(X-ARG(J))-VAL(J)*(X-ARG(I)))/H
      DELT2=DABS(VAL(J)-VAL(IEND))
      IF(J-2)6,6,3
    3 IF(DELT2-EPS)10,10,4
    4 IF(J-8)6,5,5
    5 IF(DELT2-DELT1)6,11,11
    6 CONTINUE
C     END OF AITKEN-LOOP
C
    7 J=NDIM
    8 Y=VAL(J)
    9 RETURN
C
C     THERE IS SUFFICIENT ACCURACY WITHIN NDIM-1 ITERATION STEPS
   10 IER=0
      GOTO 8
C
C     TEST VALUE DELT2 STARTS OSCILLATING
   11 IER=1
   12 J=IEND
      GOTO 8
C
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG
   13 IER=3
      GOTO 12
      END
```

Subroutines AHI and DAHI

These subroutines perform an Aitken-Hermite interpolation. Given the table x_i, y_i, y_i' ($i = 1, 2, \ldots, n$) of argument, function and derivative values, stored in the two one-dimensional storage arrays ARG and VAL, the problem is to interpolate function value Y at the given point X. It is assumed that the table represents a single-valued function $y = y(x)$ which is suitable for approximation by a polynomial[+], and that it is stored as shown in Figure 19 (first and second column) with $|x_i - X| \geq |x_j - X|$ for $i > j$.[++]

Before starting Aitken's iteration scheme, the components of vector VAL (that is, function and derivative values) are replaced by interpolation values z_i of first order (third column). This is done by the following machine-processable formulas:

$$\text{VAL(I)} = Y + \text{VAL(I+1)} * H_1$$

$$(I = 1,3,5,\ldots,2 * \text{NDIM-1}) \qquad (1)$$

$$\text{VAL(I+1)} = Y + (\text{VAL(I+2)}-Y)\,\frac{H_1}{H_1 - H_2}$$

$$(I = 1,3,5,\ldots,2 * \text{NDIM-3}) \qquad (2)$$

with

$$Y = \text{VAL(I)}, \quad H_1 = X - \text{ARG(J-1)}, \quad H_2 = X - \text{ARG(J)}$$

and

$$J = \frac{I+1}{2} + 1 .$$

Now it is possible to generate successively the upward diagonals of the triangular Aitken scheme, using the following formulas:

$$z_{1,2} = \frac{1}{x_2 - x_1} \cdot \begin{vmatrix} z_1 & x_1 - X \\ z_2 & x_2 - X \end{vmatrix} ;$$

$$z_{2,3} = \frac{1}{x_2 - x_1} \cdot \begin{vmatrix} z_2 & x_1 - X \\ z_3 & x_2 - X \end{vmatrix} ,$$

$$z_{1,2,3} = \frac{1}{x_2 - x_1} \cdot \begin{vmatrix} z_{1,2} & x_1 - X \\ z_{2,3} & x_2 - X \end{vmatrix} ;$$

$$z_{3,4} = \frac{1}{x_3 - x_2} \cdot \begin{vmatrix} z_3 & x_2 - X \\ z_4 & x_3 - X \end{vmatrix} ,$$

$$z_{2,3,4} = \frac{1}{x_3 - x_1} \cdot \begin{vmatrix} z_{2,3} & x_1 - X \\ z_{3,4} & x_3 - X \end{vmatrix} ;$$

$$z_{1,2,3,4} = \frac{1}{x_3 - x_1} \cdot \begin{vmatrix} z_{1,2,3} & x_1 - X \\ z_{2,3,4} & x_3 - X \end{vmatrix} ;$$

.....

All result values of an upward diagonal can be stored in positions of vector VAL with decreasing subscripts, using the machine-processable formula:

$$\text{VAL(K)} = \frac{\text{VAL(K)}*(X-H_1) - \text{VAL(K+1)}*(X-\text{ARG(L)})}{\text{ARG(L)} - H_1}$$

$$(J = 1,2,\ldots,I) \qquad (3)$$

with

$$K = I + 1 - J, \quad M = \left[\frac{I+3}{2}\right], \quad L = \left[\frac{K+1}{2}\right]$$

and

$$H_1 = \text{ARG(M)}$$

for $I = 1,2,\ldots,2 * \text{NDIM-2}$.

The procedure stops under the following circumstances: if the difference between two successive interpolated values VAL(1) is absolutely less than a given tolerance ε or if the absolute value of this difference stops diminishing, thus showing the influence of rounding errors (test starts at step $i = 5$, or for double-precision, $i = 8$), or if the procedure has worked through the whole triangular Aitken scheme, or if it discovers two table points with identical arguments.

[+]This order in table x_i, y_i, y_i ($i = 1, 2, \ldots, n$) could be generated by subroutines ATSG, ATSM, ATSE, DATSG, DATSM, DATSE.

[++]This means $y(x)$ should be closely approximated by a polynomial of low degree which has the table points with arguments next to x as nodes and which has the specified derivative values in these nodes.

$ARG(1) = x_1$	$VAL(1) = y_1$	$VAL(1) = z_1$	$z_{1,2}$	$z_{1,2,3}$	$z_{1,2,3,4}$
	$VAL(2) = y'_1$	$VAL(2) = z_2$	$z_{2,3}$	$z_{2,3,4}$	
$ARG(2) = x_2$	$VAL(3) - y_2$	$VAL(3) = z_3$	$z_{3,4}$		
	$VAL(4) = y'_2$	$VAL(4) = z_4$			
...	...	...	...	...	...
$ARG(NDIM) = x_n$	$VAL(2*NDIM-1) = y_n$	$VAL(2*NDIM-1) = z_{2n-1}$			
	$VAL(2*NDIM) = y'_n$				

Figure 19. Aitken's scheme for Hermite interpolation

Finally, Y is set equal to VAL(1) in case of monotonically decreasing absolute differences, or to VAL(1) of the step before if these differences start oscillating.

For reference see:

(1) F. B. Hildebrand, Introduction to Numerical Analysis, McGraw Hill, New York/Toronto, London, 1956, pp. 49-50, 352-356.

(2) Gershinsky/Levine, "Aitken-Hermite Interpolation", JACM, vol. 11, iss. 3 (1964), pp. 352-356.

```
C
C     ..................................................................
C
C        SUBROUTINE AHI
C
C        PURPOSE
C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT, FUNCTION, AND
C           DERIVATIVE VALUES.
C
C        USAGE
C           CALL AHI (X,ARG,VAL,Y,NDIM,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      - THE ARGUMENT VALUE SPECIFIED BY INPUT.
C           ARG    - THE INPUT VECTOR (DIMENSION NDIM) OF ARGUMENT
C                    VALUES OF THE TABLE (NOT DESTROYED).
C           VAL    - THE INPUT VECTOR (DIMENSION 2*NDIM) OF FUNCTION
C                    AND DERIVATIVE VALUES OF THE TABLE (DESTROYED).
C                    FUNCTION AND DERIVATIVE VALUES MUST BE STORED IN
C                    PAIRS, THAT MEANS BEGINNING WITH FUNCTION VALUE AT
C                    POINT ARG(1) EVERY FUNCTION VALUE MUST BE FOLLOWED
C                    BY THE VALUE OF DERIVATIVE AT THE SAME POINT.
C           Y      - THE RESULTING INTERPOLATED FUNCTION VALUE.
C           NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
C                    POINTS IN TABLE (ARG,VAL).
C           EPS    - AN INPUT CONSTANT WHICH IS USED AS UPPER BOUND
C                    FOR THE ABSOLUTE ERROR.
C           IER    - A RESULTING ERROR PARAMETER.
C
C        REMARKS
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C               SUBROUTINES ATSG, ATSM OR ATSE COULD BE USED IN A
C               PREVIOUS STAGE.
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C               THAN 1.
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C               (2*NDIM-2) STEPS. FURTHER IT IS TERMINATED IF THE
C               PROCEDURE DISCOVERS TWO ARGUMENT VALUES IN VECTOR ARG
C               WHICH ARE IDENTICAL. DEPENDENT ON THESE FOUR CASES,
C               ERROR PARAMETER IER IS CODED IN THE FOLLOWING FORM
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                        ACCURACY (NO ERROR).
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                        NDIM IS LESS THAN 2, OR THE REQUIRED ACCURACY
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                        TABLE. NDIM SHOULD BE INCREASED.
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                        IN VECTOR ARG WHICH ARE IDENTICAL.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           INTERPOLATION IS DONE BY MEANS OF AITKENS SCHEME OF
C           HERMITE INTERPOLATION. ON RETURN Y CONTAINS AN INTERPOLATED
C           FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK
C           (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.314-317, AND
C           GERSHINSKY/LEVINE, AITKEN-HERMITE INTERPOLATION,
C           JACM, VOL.11, ISS.3 (1964), PP.352-356.
C     ..................................................................
C
      SUBROUTINE AHI(X,ARG,VAL,Y,NDIM,EPS,IER)
C
C
      DIMENSION ARG(1),VAL(1)
      IER=2
      H2=X-ARG(1)
      IF(NDIM-1)2,1,3
    1 Y=VAL(1)+VAL(2)*H2
    2 RETURN
C
C     VECTOR ARG HAS MORE THAN 1 ELEMENT.
C     THE FIRST STEP PREPARES VECTOR VAL SUCH THAT AITKEN SCHEME CAN BE
C     USED.
    3 I=1
      DO 5 J=2,NDIM
      H1=H2
      H2=X-ARG(J)
      Y=VAL(I)
      VAL(I)=Y+VAL(I+1)*H1
      H=H1-H2
      IF(H)4,13,4
    4 VAL(I+1)=Y+(VAL(I+2)-Y)*H1/H
    5 I=I+2
      VAL(I)=VAL(I)+VAL(I+1)*H2
C     END OF FIRST STEP
C
C     PREPARE AITKEN SCHEME
      DELT2=0.
      IEND=I-1
C
C     START AITKEN-LOOP
      DO 9 I=1,IEND
      DELT1=DELT2
      Y=VAL(1)
      M=(I+3)/2
      H1=ARG(M)
      DO 6 J=1,I
      K=I+1-J
      L=(K+1)/2
      H=ARG(L)-H1
      IF(H)6,14,6
    6 VAL(K)=(VAL(K)*(X-H1)-VAL(K+1)*(X-ARG(L)))/H
      DELT2=ABS(Y-VAL(1))
      IF(DELT2-EPS)11,11,7
    7 IF(I-5)9,8,8
    8 IF(DELT2-DELT1)9,12,12
    9 CONTINUE
C     END OF AITKEN-LOOP
C
   10 Y=VAL(1)
      RETURN
C
C     THERE IS SUFFICIENT ACCURACY WITHIN 2*NDIM-2 ITERATION STEPS
   11 IER=0
      GOTO 10
C
C     TEST VALUE DELT2 STARTS OSCILLATING
   12 IER=1
      RETURN
C
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG
   13 Y=VAL(1)
   14 IER=3
      RETURN
      END
C
C     ..................................................................
C
C        SUBROUTINE DAHI
C
C        PURPOSE
C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT, FUNCTION, AND
C           DERIVATIVE VALUES.
C
C        USAGE
C           CALL DAHI (X,ARG,VAL,Y,NDIM,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      - DOUBLE PRECISION ARGUMENT VALUE SPECIFIED BY INPUT.
C           ARG    - DOUBLE PRECISION INPUT VECTOR (DIMENSION NDIM) OF
C                    ARGUMENT VALUES OF THE TABLE (NOT DESTROYED).
C           VAL    - DOUBLE PRECISION INPUT VECTOR (DIMENSION 2*NDIM) OF
C                    FUNCTION AND DERIVATIVE VALUES OF THE TABLE (DES-
C                    TROYED). FUNCTION AND DERIVATIVE VALUES MUST BE
C                    STORED IN PAIRS, THAT MEANS BEGINNING WITH FUNCTION
C                    VALUE AT POINT ARG(1) EVERY FUNCTION VALUE MUST BE
C                    FOLLOWED BY THE DERIVATIVE VALUE AT THE SAME POINT.
C           Y      - RESULTING INTERPOLATED DOUBLE PRECISION FUNCTION
C                    VALUE.
C           NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
C                    POINTS IN TABLE (ARG,VAL).
C           EPS    - SINGLE PRECISION INPUT CONSTANT WHICH IS USED AS
C                    UPPER BOUND FOR THE ABSOLUTE ERROR.
C           IER    - A RESULTING ERROR PARAMETER.
C
C        REMARKS
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C               SUBROUTINES DATSG, DATSM OR DATSE COULD BE USED IN A
C               PREVIOUS STAGE.
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C               THAN 1.
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C               (2*NDIM-2) STEPS. FURTHER IT IS TERMINATED IF THE
C               PROCEDURE DISCOVERS TWO ARGUMENT VALUES IN VECTOR ARG
C               WHICH ARE IDENTICAL. DEPENDENT ON THESE FOUR CASES,
C               ERROR PARAMETER IER IS CODED IN THE FOLLOWING FORM
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                        ACCURACY (NO ERROR).
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                        NDIM IS LESS THAN 2, OR THE REQUIRED ACCURACY
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                        TABLE. NDIM SHOULD BE INCREASED.
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                        IN VECTOR ARG WHICH ARE IDENTICAL.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
```

```
C                                                                       DAHI 620
C        METHOD                                                         DAHI 630
C           INTERPOLATION IS DONE BY MEANS OF AITKENS SCHEME OF         DAHI 640
C           HERMITE INTERPOLATION. ON RETURN Y CONTAINS AN INTERPOLATED DAHI 650
C           FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK  DAHI 660
C           (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE DAHI 670
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,         DAHI 680
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.314-317, AND DAHI 690
C           GERSHINSKY/LEVINE, AITKEN-HERMITE INTERPOLATION,            DAHI 700
C           JACM, VOL.11, ISS.3 (1964), PP.352-356.                     DAHI 710
C                                                                       DAHI 720
C     ..................................................................DAHI 730
C                                                                       DAHI 740
      SUBROUTINE DAHI(X,ARG,VAL,Y,NDIM,EPS,IER)                         DAHI 750
C                                                                       DAHI 760
C                                                                       DAHI 770
      DIMENSION ARG(1),VAL(1)                                           DAHI 780
      DOUBLE PRECISION ARG,VAL,X,Y,H,H1,H2                              DAHI 790
      IER=2                                                             DAHI 800
      H2=X-ARG(1)                                                       DAHI 810
      IF(NDIM-1)2,1,3                                                   DAHI 820
    1 Y=VAL(1)+VAL(2)*H2                                                DAHI 830
    2 RETURN                                                            DAHI 840
C                                                                       DAHI 850
C     VECTOR ARG HAS MORE THAN 1 ELEMENT.                               DAHI 860
C     THE FIRST STEP PREPARES VECTOR VAL SUCH THAT AITKEN SCHEME CAN BE DAHI 870
C     USED.                                                             DAHI 880
    3 I=1                                                               DAHI 890
      DO 5 J=2,NDIM                                                     DAHI 900
      H1=H2                                                             DAHI 910
      H2=X-ARG(J)                                                       DAHI 920
      Y=VAL(I)                                                          DAHI 930
      VAL(I)=Y+VAL(I+1)*H1                                              DAHI 940
      H=H1-H2                                                           DAHI 950
      IF(H)4,13,4                                                       DAHI 960
    4 VAL(I+1)=Y+(VAL(I+2)-Y)*H1/H                                      DAHI 970
    5 I=I+2                                                             DAHI 980
      VAL(I)=VAL(I)+VAL(I+1)*H2                                         DAHI 990
C     END OF FIRST STEP                                                 DAHI1000
C                                                                       DAHI1010
C     PREPARE AITKEN SCHEME                                             DAHI1020
      DELT2=0.                                                          DAHI1030
      IEND=I-1                                                          DAHI1040
C                                                                       DAHI1050
C     START AITKEN-LOOP                                                 DAHI1060
      DO 9 I=1,IEND                                                     DAHI1070
      DELT1=DELT2                                                       DAHI1080
      Y=VAL(1)                                                          DAHI1090
      M=(I+3)/2                                                         DAHI1100
      H1=ARG(M)                                                         DAHI1110
      DO 6 J=1,I                                                        DAHI1120
      K=I+1-J                                                           DAHI1130
      L=(K+1)/2                                                         DAHI1140
      H=ARG(L)-H1                                                       DAHI1150
      IF(H)6,14,6                                                       DAHI1160
    6 VAL(K)=(VAL(K)*(X-H1)-VAL(K+1)*(X-ARG(L)))/H                      DAHI1170
      DELT2=DABS(Y-VAL(1))                                              DAHI1180
      IF(DELT2-EPS)11,11,7                                              DAHI1190
    7 IF(I-8)9,8,8                                                      DAHI1200
    8 IF(DELT2-DELT1)9,12,12                                            DAHI1210
    9 CONTINUE                                                          DAHI1220
C     END OF AITKEN-LOOP                                                DAHI1230
C                                                                       DAHI1240
   10 Y=VAL(1)                                                          DAHI1250
      RETURN                                                            DAHI1260
C                                                                       DAHI1270
C     THERE IS SUFFICIENT ACCURACY WITHIN 2*NDIM-2 ITERATION STEPS      DAHI1280
   11 IER=0                                                             DAHI1290
      GOTO 10                                                           DAHI1300
C                                                                       DAHI1310
C     TEST VALUE DELT2 STARTS OSCILLATING                               DAHI1320
   12 IER=1                                                             DAHI1330
      RETURN                                                            DAHI1340
C                                                                       DAHI1350
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG             DAHI1360
   13 Y=VAL(1)                                                          DAHI1370
   14 IER=3                                                             DAHI1380
      RETURN                                                            DAHI1390
      END                                                               DAHI1400
```

Subroutines ACFI and DACFI

These subroutines perform a continued fraction interpolation. Given the table x_i, y_i ($i = 1, 2, \ldots, n$) of argument and function values, stored in the two one-dimensional storage arrays ARG and VAL, the problem is to interpolate function value Y at the given point X. It is assumed that the table represents a single-valued function $y = y(x)$ which is suitable for approximation by a rational function+, and that $|x_i - X| \geq x_j - X|$ for $i > j$.++

Then, using the following formulas:

$$y_{1,2} = \frac{x_2 - x_1}{y_2 - y_1};$$

$$y_{1,3} = \frac{x_3 - x_1}{y_3 - y_1}; \quad y_{1,2,3} = \frac{x_3 - x_2}{y_{1,3} - y_{1,2}};$$

$$y_{1,4} = \frac{x_4 - x_1}{y_4 - y_1}; \quad y_{1,2,4} = \frac{x_4 - x_2}{y_{1,4} - y_{1,2}};$$

$$y_{1,2,3,4} = \frac{x_4 - x_3}{y_{1,2,4} - y_{1,2,3}};$$

the triangular scheme of inverted differences shown in Figure 20 can be generated by row for the given table x_i, y_i.

All result values of row i can be stored in VAL(I). Thus, it is possible to generate the downward diagonal of the inverted differences scheme in storage array VAL, using the machine-processable formula:

$$\mathrm{VAL(I)} = \frac{\mathrm{ARG(I)} - \mathrm{ARG(J)}}{\mathrm{VAL(I)} - \mathrm{VAL(J)}} \quad (J = 1, 2, \ldots, I-1) \qquad (1)$$

for I = 2, 3, ..., NDIM.

+This order in table x_i, y_i ($i = 1, 2, \ldots, n$) could be generated by subroutines ATSG, ATSM, ATSE, DATSG, DATSM, DATSE.

++This means y(x) should be closely approximated by a rational function, with numerator and denominator of low degree, which has the table points with arguments next to x as nodes.

x_1	y_1					
x_2	y_2	$\underline{y_{1,2}}$				
x_3	y_3	$y_{1,3}$	$\underline{y_{1,2,3}}$			
x_4	y_4	$y_{1,4}$	$y_{1,2,4}$			
.	.	.	.	$\underline{y_{1,2,3,4}}$		
.	.	.	.	.		
.	.	.	.	.	...	
.	.	.	.	.		
x_n	y_n	$y_{1,n}$	$y_{1,2,n}$	$y_{1,2,3,n}$		$\underline{y_{1,2,3,\ldots,n}}$

Figure 20. Triangular scheme for fraction interpolation

If for J = I-1, VAL(I) is equal to the infinity element, table point ARG(I), VAL(I) (that is, x_i, y_i) gets interchanged with a table point ahead.

Now, after computation of each new component VAL(I), continued fraction interpolation generates the following set of parameters:

$$P_3 = VAL(I) * P_2 + (X - ARG(I-1)) * P_1 \qquad (2)$$

$$Q_3 = VAL(I) * Q_2 + (X - ARG(I-1)) * Q_1 \qquad (3)$$

and

$$Y = P_3/Q_3 \qquad (4)$$

starting with $P_1 = 1$, $P_2 = VAL(1)$, $Q_1 = 0$, $Q_2 = 1$. After having done this, it sets $P_1 = P_2$, $P_2 = P_3$, $Q_1 = Q_2$, $Q_2 = Q_3$ after each step.

The procedure stops under the following conditions: if the difference between two successive values of Y is absolutely less than a given tolerance ε, or if the absolute value of this difference starts oscillating (test of oscillating differences starts at step I = 8, or for double-precision, I = 10), or if I has become NDIM, or if the procedure discovers two table points with identical argument values.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis, McGraw-Hill, New York/Toronto/London, 1956, pp. 395-406.

```
C
C     ..................................................................
C
C        SUBROUTINE ACFI
C
C        PURPOSE
C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE
C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT AND FUNCTION
C           VALUES.
C
C        USAGE
C           CALL ACFI (X,ARG,VAL,Y,NDIM,EPS,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      - THE ARGUMENT VALUE SPECIFIED BY INPUT.
C           ARG    - THE INPUT VECTOR (DIMENSION NDIM) OF ARGUMENT
C                    VALUES OF THE TABLE (POSSIBLY DESTROYED).
C           VAL    - THE INPUT VECTOR (DIMENSION NDIM) OF FUNCTION
C                    VALUES OF THE TABLE (DESTROYED).
C           Y      - THE RESULTING INTERPOLATED FUNCTION VALUE.
C           NDIM   - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF
C                    POINTS IN TABLE (ARG,VAL).
C           EPS    - AN INPUT CONSTANT WHICH IS USED AS UPPER BOUND
C                    FOR THE ABSOLUTE ERROR.
C           IER    - A RESULTING ERROR PARAMETER.
C
C        REMARKS
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL),
C               SUBROUTINES ATSG, ATSM OR ATSE COULD BE USED IN A
C               PREVIOUS STAGE.
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS
C               THAN 1.
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER
C               (NDIM-1) STEPS (THE NUMBER OF POSSIBLE STEPS IS
C               DIMINISHED IF AT ANY STAGE INFINITY ELEMENT APPEARS IN
C               THE DOWNWARD DIAGONAL OF INVERTED-DIFFERENCES-SCHEME
C               AND IF IT IS IMPOSSIBLE TO ELIMINATE THIS INFINITY
C               ELEMENT BY INTERCHANGING OF TABLE POINTS).
C               FURTHER IT IS TERMINATED IF THE PROCEDURE DISCOVERS TWO
C               ARGUMENT VALUES IN VECTOR ARG WHICH ARE IDENTICAL.
C               DEPENDENT ON THESE FOUR CASES, ERROR PARAMETER IER IS
C               CODED IN THE FOLLOWING FORM
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED
C                        ACCURACY (NO ERROR).
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE
C                        NDIM IS LESS THAN 2, OR THE REQUIRED ACCURACY
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN
C                        TABLE. NDIM SHOULD BE INCREASED.
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES
C                        IN VECTOR ARG WHICH ARE IDENTICAL.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           INTERPOLATION IS DONE BY CONTINUED FRACTIONS AND INVERTED-
C           DIFFERENCES-SCHEME. ON RETURN Y CONTAINS AN INTERPOLATED
C           FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK
C           (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.395-406.
C
C     ..................................................................
C
      SUBROUTINE ACFI(X,ARG,VAL,Y,NDIM,EPS,IER)
C
C
      DIMENSION ARG(1),VAL(1)
      IER=2
      IF(NDIM)20,20,1
    1 Y=VAL(1)
      DELT2=0.
      IF(NDIM-1)20,20,2
C
C     PREPARATIONS FOR INTERPOLATION LOOP
    2 P2=1.
      P3=Y
      Q2=0.
      Q3=1.
C
C
C     START INTERPOLATION LOOP
      DO 16 I=2,NDIM
      II=0
      P1=P2
      P2=P3
      Q1=Q2
      Q2=Q3
      Z=Y
      DELT1=DELT2
      JEND=I-1
C
C     COMPUTATION OF INVERTED DIFFERENCES
    3 AUX=VAL(I)
      DO 10 J=1,JEND
      H=VAL(I)-VAL(J)
      IF(ABS(H)-1.E-6*ABS(VAL(I)))4,4,9
    4 IF(ARG(I)-ARG(J))5,17,5
    5 IF(J-JEND)8,6,6
C
C     INTERCHANGE ROW I WITH ROW I+II
    6 II=II+1
      III=I+II
      IF(III-NDIM)7,7,19
    7 VAL(I)=VAL(III)
      VAL(III)=AUX
      AUX=ARG(I)
      ARG(I)=ARG(III)
      ARG(III)=AUX
      GOTO 3
C
C     COMPUTATION OF VAL(I) IN CASE VAL(I)=VAL(J) AND J LESS THAN I-1
    8 VAL(I)=1.E75
      GOTO 10
C
C     COMPUTATION OF VAL(I) IN CASE VAL(I) NOT EQUAL TO VAL(J)
    9 VAL(I)=(ARG(I)-ARG(J))/H
   10 CONTINUE
C     INVERTED DIFFERENCES ARE COMPUTED
C
C     COMPUTATION OF NEW Y
      P3=VAL(I)*P2+(X-ARG(I-1))*P1
      Q3=VAL(I)*Q2+(X-ARG(I-1))*Q1
      IF(Q3)11,12,11
   11 Y=P3/Q3
      GOTO 13
   12 Y=1.E75
   13 DELT2=ABS(Z-Y)
      IF(DELT2-EPS)19,19,14
   14 IF(I-8)16,15,15
   15 IF(DELT2-DELT1)16,18,18
   16 CONTINUE
C     END OF INTERPOLATION LOOP
C
C
      RETURN
C
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG
   17 IER=3
      RETURN
C
C     TEST VALUE DELT2 STARTS OSCILLATING
   18 Y=Z
      IER=1
      RETURN
C
C     THERE IS SATISFACTORY ACCURACY WITHIN NDIM-1 STEPS
   19 IER=0
   20 RETURN
      END
```

```
C                                                                       DACF  10
C     ...................................................................DACF  20
C                                                                       DACF  30
C        SUBROUTINE DACFI                                               DACF  40
C                                                                       DACF  50
C        PURPOSE                                                        DACF  60
C           TO INTERPOLATE FUNCTION VALUE Y FOR A GIVEN ARGUMENT VALUE  DACF  70
C           X USING A GIVEN TABLE (ARG,VAL) OF ARGUMENT AND FUNCTION    DACF  80
C           VALUES.                                                     DACF  90
C                                                                       DACF 100
C        USAGE                                                          DACF 110
C           CALL DACFI (X,ARG,VAL,Y,NDIM,EPS,IER)                       DACF 120
C                                                                       DACF 130
C        DESCRIPTION OF PARAMETERS                                      DACF 140
C           X       - DOUBLE PRECISION ARGUMENT VALUE SPECIFIED BY INPUT.DACF 150
C           ARG     - DOUBLE PRECISION INPUT VECTOR (DIMENSION NDIM) OF DACF 160
C                     ARGUMENT VALUES OF THE TABLE (POSSIBLY DESTROYED). DACF 170
C           VAL     - DOUBLE PRECISION INPUT VECTOR (DIMENSION NDIM) OF DACF 180
C                     FUNCTION VALUES OF THE TABLE (DESTROYED).         DACF 190
C           Y       - RESULTING INTERPOLATED DOUBLE PRECISION FUNCTION  DACF 200
C                     VALUE.                                            DACF 210
C           NDIM    - AN INPUT VALUE WHICH SPECIFIES THE NUMBER OF      DACF 220
C                     POINTS IN TABLE (ARG,VAL).                        DACF 230
C           EPS     - SINGLE PRECISION INPUT CONSTANT WHICH IS USED AS  DACF 240
C                     UPPER BOUND FOR THE ABSOLUTE ERROR.               DACF 250
C           IER     - A RESULTING ERROR PARAMETER.                      DACF 260
C                                                                       DACF 270
C        REMARKS                                                        DACF 280
C           (1) TABLE (ARG,VAL) SHOULD REPRESENT A SINGLE-VALUED        DACF 290
C               FUNCTION AND SHOULD BE STORED IN SUCH A WAY, THAT THE   DACF 300
C               DISTANCES ABS(ARG(I)-X) INCREASE WITH INCREASING        DACF 310
C               SUBSCRIPT I. TO GENERATE THIS ORDER IN TABLE (ARG,VAL), DACF 320
C               SUBROUTINES DATSG, DATSM OR DATSE COULD BE USED IN A    DACF 330
C               PREVIOUS STAGE.                                         DACF 340
C           (2) NO ACTION BESIDES ERROR MESSAGE IN CASE NDIM LESS       DACF 350
C               THAN 1.                                                 DACF 360
C           (3) INTERPOLATION IS TERMINATED EITHER IF THE DIFFERENCE    DACF 370
C               BETWEEN TWO SUCCESSIVE INTERPOLATED VALUES IS           DACF 380
C               ABSOLUTELY LESS THAN TOLERANCE EPS, OR IF THE ABSOLUTE  DACF 390
C               VALUE OF THIS DIFFERENCE STOPS DIMINISHING, OR AFTER    DACF 400
C               (NDIM-1) STEPS (THE NUMBER OF POSSIBLE STEPS IS         DACF 410
C               DIMINISHED IF AT ANY STAGE INFINITY ELEMENT APPEARS IN  DACF 420
C               THE DOWNWARD DIAGONAL OF INVERTED-DIFFERENCES-SCHEME    DACF 430
C               AND IF IT IS IMPOSSIBLE TO ELIMINATE THIS INFINITY      DACF 440
C               ELEMENT BY INTERCHANGING OF TABLE POINTS).              DACF 450
C               FURTHER IT IS TERMINATED IF THE PROCEDURE DISCOVERS TWO DACF 460
C               ARGUMENT VALUES IN VECTOR ARG WHICH ARE IDENTICAL.      DACF 470
C               DEPENDENT ON THESE FOUR CASES, ERROR PARAMETER IER IS   DACF 480
C               CODED IN THE FOLLOWING FORM                             DACF 490
C                IER=0 - IT WAS POSSIBLE TO REACH THE REQUIRED          DACF 500
C                        ACCURACY (NO ERROR).                           DACF 510
C                IER=1 - IT WAS IMPOSSIBLE TO REACH THE REQUIRED        DACF 520
C                        ACCURACY BECAUSE OF ROUNDING ERRORS.           DACF 530
C                IER=2 - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE    DACF 540
C                        NDIM IS LESS THAN 2, OR THE REQUIRED ACCURACY  DACF 550
C                        COULD NOT BE REACHED BY MEANS OF THE GIVEN     DACF 560
C                        TABLE. NDIM SHOULD BE INCREASED.               DACF 570
C                IER=3 - THE PROCEDURE DISCOVERED TWO ARGUMENT VALUES   DACF 580
C                        IN VECTOR ARG WHICH ARE IDENTICAL.             DACF 590
C                                                                       DACF 600
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DACF 610
C           NONE                                                        DACF 620
C                                                                       DACF 630
C        METHOD                                                         DACF 640
C           INTERPOLATION IS DONE BY CONTINUED FRACTIONS AND INVERTED-  DACF 650
C           DIFFERENCES-SCHEME. ON RETURN Y CONTAINS AN INTERPOLATED    DACF 660
C           FUNCTION VALUE AT POINT X, WHICH IS IN THE SENSE OF REMARK  DACF 670
C           (3) OPTIMAL WITH RESPECT TO GIVEN TABLE. FOR REFERENCE, SEE DACF 680
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,         DACF 690
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.395-406.     DACF 700
C                                                                       DACF 710
C     ...................................................................DACF 720
C                                                                       DACF 730
      SUBROUTINE DACFI(X,ARG,VAL,Y,NDIM,EPS,IER)                        DACF 740
C                                                                       DACF 750
C                                                                       DACF 760
      DIMENSION ARG(1),VAL(1)                                           DACF 770
      DOUBLE PRECISION ARG,VAL,X,Y,Z,P1,P2,P3,Q1,Q2,Q3,AUX,H            DACF 780
      IER=2                                                             DACF 790
      IF(NDIM)20,20,1                                                   DACF 800
    1 Y=VAL(1)                                                          DACF 810
      DELT2=0.                                                          DACF 820
      IF(NDIM-1)20,20,2                                                 DACF 830
C                                                                       DACF 840
C     PREPARATIONS FOR INTERPOLATION LOOP                               DACF 850
    2 P2=1.D0                                                           DACF 860
      P3=Y                                                              DACF 870
      Q2=0.D0                                                           DACF 880
      Q3=1.D0                                                           DACF 890
C                                                                       DACF 900
C                                                                       DACF 910
C     START INTERPOLATION LOOP                                          DACF 920
      DO 16 I=2,NDIM                                                    DACF 930
      II=0                                                              DACF 940
      P1=P2                                                             DACF 950
      P2=P3                                                             DACF 960
      Q1=Q2                                                             DACF 970
      Q2=Q3                                                             DACF 980
      Z=Y                                                               DACF 990
      DELT1=DELT2                                                       DACF1000
      JEND=I-1                                                          DACF1010
C                                                                       DACF1020
C     COMPUTATION OF INVERTED DIFFERENCES                               DACF1030
    3 AUX=VAL(I)                                                        DACF1040
      DO 10 J=1,JEND                                                    DACF1050
      H=VAL(I)-VAL(J)                                                   DACF1060
      IF(DABS(H)-1.D-13*DABS(VAL(I)))4,4,9                              DACF1070
    4 IF(ARG(I)-ARG(J))5,17,5                                           DACF1080
    5 IF(J-JEND)8,6,6                                                   DACF1090
C                                                                       DACF1100
C     INTERCHANGE ROW I WITH ROW I+II                                   DACF1110
    6 II=II+1                                                           DACF1120
      III=I+II                                                          DACF1130
      IF(III-NDIM)7,7,19                                                DACF1140
    7 VAL(I)=VAL(III)                                                   DACF1150
      VAL(III)=AUX                                                      DACF1160
      AUX=ARG(I)                                                        DACF1170
      ARG(I)=ARG(III)                                                   DACF1180
      ARG(III)=AUX                                                      DACF1190
      GOTO 3                                                            DACF1200
C                                                                       DACF1210
C     COMPUTATION OF VAL(I) IN CASE VAL(I)=VAL(J) AND J LESS THAN I-1   DACF1220
    8 VAL(I)=1.D75                                                      DACF1230
      GOTO 10                                                           DACF1240
C                                                                       DACF1250
C     COMPUTATION OF VAL(I) IN CASE VAL(I) NOT EQUAL TO VAL(J)          DACF1260
    9 VAL(I)=(ARG(I)-ARG(J))/H                                          DACF1270
   10 CONTINUE                                                          DACF1280
C     INVERTED DIFFERENCES ARE COMPUTED                                 DACF1290
C                                                                       DACF1300
C     COMPUTATION OF NEW Y                                              DACF1310
      P3=VAL(I)*P2+(X-ARG(I-1))*P1                                      DACF1320
      Q3=VAL(I)*Q2+(X-ARG(I-1))*Q1                                      DACF1330
      IF(Q3)11,12,11                                                    DACF1340
   11 Y=P3/Q3                                                           DACF1350
      GOTO 13                                                           DACF1360
   12 Y=1.D75                                                           DACF1370
   13 DELT2=DABS(Z-Y)                                                   DACF1380
      IF(DELT2-EPS)19,19,14                                             DACF1390
   14 IF(I-10)16,15,15                                                  DACF1400
   15 IF(DELT2-DELT1)16,18,18                                           DACF1410
   16 CONTINUE                                                          DACF1420
C     END OF INTERPOLATION LOOP                                         DACF1430
C                                                                       DACF1440
C                                                                       DACF1450
      RETURN                                                            DACF1460
C                                                                       DACF1470
C     THERE ARE TWO IDENTICAL ARGUMENT VALUES IN VECTOR ARG             DACF1480
   17 IER=3                                                             DACF1490
      RETURN                                                            DACF1500
C                                                                       DACF1510
C     TEST VALUE DELT2 STARTS OSCILLATING                               DACF1520
   18 Y=Z                                                               DACF1530
      IER=1                                                             DACF1540
      RETURN                                                            DACF1550
C                                                                       DACF1560
C     THERE IS SATISFACTORY ACCURACY WITHIN NDIM-1 STEPS                DACF1570
   19 IER=0                                                             DACF1580
   20 RETURN                                                            DACF1590
      END                                                               DACF1600
```

Subroutines ATSG and DATSG

These subroutines perform a table selection out of a given general table.

Given a search argument x and a general table z_i, f_i (i = 1, 2, ..., irow) stored in two vectors Z and F (in case icol = 1), or z_i, f_i, f_i' (i = 1, 2, ..., irow) stored in vector Z and the two columns of matrix F (in case icol = 2); the problem is to select those n ($\leq$ irow) points z_{i_j}, f_{i_j} or z_{i_j}, f_{i_j}, f'_{i_j} (j = 1, 2, ..., n), respectively, which are next to x and to store their coordinates in two vectors ARG and VAL so that the distances $|x - z_{i_j}|$ increase with increasing subscript j. This problem may arise when using interpolation subroutines ALI, AHI, ACFI, DALI, DAHI, DACFI. The table for the case icol = 1 is shown in Figure 21, and that for case icol = 2 in Figure 22.

Selection is done in two parts. In the first part, auxiliary vector WORK with components $z_i - x$ (i = 1, 2, ..., irow) is generated and its greatest component, $\max_i |z_i - x|$, is stored in an auxiliary storage location. In each of the n steps of the second part the subscript i_j of the smallest component of vector WORK is searched for. Afterwards this component is replaced by a number greater than $\max_i |z_i - x|$.

Then, z_{i_j} goes into the storage location ARG(J) and:

In case icol = 1 -- f_{i_j} into the storage location VAL(J)

In case icol = 2 -- f_{i_j} into the storage location VAL(2 * J - 1) and f'_{i_j} into the storage location VAL(2 * J)

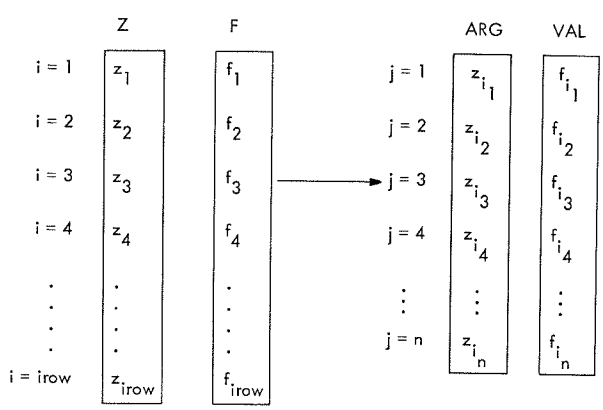


Figure 21. Table selection in case icol=1

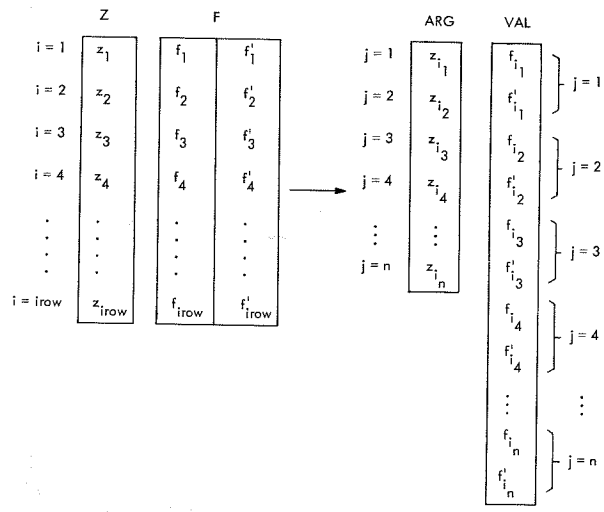


Figure 22. Table selection in case icol=2

If (by an error of the user) the dimension n of the table to be selected is greater than the dimension irow of the given table, the procedure selects only a maximum table of irow points. In order to avoid errors in further work with table (ARG, VAL), the user ought to check the correspondence between the selected table and its dimension by comparison of n and irow. This test may be done before or after calling subroutine ATSG or DATSG.

```
C
C     ..................................................................
C
C        SUBROUTINE ATSG
C
C        PURPOSE
C           NDIM POINTS OF A GIVEN GENERAL TABLE ARE SELECTED AND
C           ORDERED SUCH THAT ABS(ARG(I)-X).GE.ABS(ARG(J)-X) IF I.GT.J.
C
C        USAGE
C           CALL ATSG (X,Z,F,WORK,IROW,ICOL,ARG,VAL,NDIM)
C
C        DESCRIPTION OF PARAMETERS
C           X       - THE SEARCH ARGUMENT.
C           Z       - THE VECTOR OF ARGUMENT VALUES (DIMENSION IROW).
C           F       - IN CASE ICOL=1, F IS THE VECTOR OF FUNCTION VALUES
C                     (DIMENSION IROW).
C                     IN CASE ICOL=2, F IS AN IROW BY 2 MATRIX. THE FIRST
C                     COLUMN SPECIFIES THE VECTOR OF FUNCTION VALUES AND
C                     THE SECOND THE VECTOR OF DERIVATIVES.
C           WORK    - A WORKING STORAGE (DIMENSION IROW).
C           IROW    - THE DIMENSION OF VECTORS Z AND WORK AND OF EACH
C                     COLUMN IN MATRIX F.
C           ICOL    - THE NUMBER OF COLUMNS IN F (I.E. 1 OR 2).
C           ARG     - THE RESULTING VECTOR OF SELECTED AND ORDERED
C                     ARGUMENT VALUES (DIMENSION NDIM).
C           VAL     - THE RESULTING VECTOR OF SELECTED FUNCTION VALUES
C                     (DIMENSION NDIM) IN CASE ICOL=1. IN CASE ICOL=2,
C                     VAL IS THE VECTOR OF FUNCTION AND DERIVATIVE VALUES
C                     (DIMENSION 2*NDIM) WHICH ARE STORED IN PAIRS (I.E.
C                     EACH FUNCTION VALUE IS FOLLOWED BY ITS DERIVATIVE
C                     VALUE).
C           NDIM    - THE NUMBER OF POINTS WHICH MUST BE SELECTED OUT OF
C                     THE GIVEN TABLE (Z,F).
C
C        REMARKS
C           NO ACTION IN CASE IROW LESS THAN 1.
C           IF INPUT VALUE NDIM IS GREATER THAN IROW, THE PROGRAM
C           SELECTS ONLY A MAXIMUM TABLE OF IROW POINTS.  THEREFORE THE
C           USER OUGHT TO CHECK CORRESPONDENCE BETWEEN TABLE (ARG,VAL)
C           AND ITS DIMENSION BY COMPARISON OF NDIM AND IROW, IN ORDER
C           TO GET CORRECT RESULTS IN FURTHER WORK WITH TABLE (ARG,VAL).
C           THIS TEST MAY BE DONE BEFORE OR AFTER CALLING
C           SUBROUTINE ATSG.
C           SUBROUTINE ATSG ESPECIALLY CAN BE USED FOR GENERATING THE
C           TABLE (ARG,VAL) NEEDED IN SUBROUTINES ALI, AHI, AND ACFI.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SELECTION IS DONE BY GENERATING THE VECTOR WORK WITH
C           COMPONENTS WORK(I)=ABS(Z(I)-X) AND AT EACH OF THE NDIM STEPS
C           (OR IROW STEPS IF NDIM IS GREATER THAN IROW)
C           SEARCHING FOR THE SUBSCRIPT OF THE SMALLEST COMPONENT, WHICH
C           IS AFTERWARDS REPLACED BY A NUMBER GREATER THAN
C           MAX(WORK(I)).
```

```
C                                                                       ATSG 580
C     ..................................................................ATSG 590
C                                                                       ATSG 600
      SUBROUTINE ATSG(X,Z,F,WORK,IROW,ICOL,ARG,VAL,NDIM)                ATSG 610
C                                                                       ATSG 620
C                                                                       ATSG 630
      DIMENSION Z(1),F(1),WORK(1),ARG(1),VAL(1)                         ATSG 640
      IF(IROW)11,11,1                                                   ATSG 650
    1 N=NDIM                                                            ATSG 660
C     IF N IS GREATER THAN IROW, N IS SET EQUAL TO IROW.                ATSG 670
      IF(N-IROW)3,3,2                                                   ATSG 680
    2 N=IROW                                                            ATSG 690
C                                                                       ATSG 700
C     GENERATION OF VECTOR WORK AND COMPUTATION OF ITS GREATEST ELEMENT.ATSG 710
    3 B=0.                                                              ATSG 720
      DO 5 I=1,IROW                                                     ATSG 730
      DELTA=ABS(Z(I)-X)                                                 ATSG 740
      IF(DELTA-B)5,5,4                                                  ATSG 750
    4 B=DELTA                                                           ATSG 760
    5 WORK(I)=DELTA                                                     ATSG 770
C                                                                       ATSG 780
C     GENERATION OF TABLE (ARG,VAL)                                     ATSG 790
      B=B+1.                                                            ATSG 800
      DO 10 J=1,N                                                       ATSG 810
      DELTA=B                                                           ATSG 820
      DO 7 I=1,IROW                                                     ATSG 830
      IF(WORK(I)-DELTA)6,7,7                                            ATSG 840
    6 II=I                                                              ATSG 850
      DELTA=WORK(I)                                                     ATSG 860
    7 CONTINUE                                                          ATSG 870
      ARG(J)=Z(II)                                                      ATSG 880
      IF(ICOL-1)8,9,8                                                   ATSG 890
    8 VAL(2*J-1)=F(II)                                                  ATSG 900
      III=II+IROW                                                       ATSG 910
      VAL(2*J)=F(III)                                                   ATSG 920
      GOTO 10                                                           ATSG 930
    9 VAL(J)=F(II)                                                      ATSG 940
   10 WORK(II)=B                                                        ATSG 950
   11 RETURN                                                            ATSG 960
      END                                                               ATSG 970
```

```
C                                                                       DTSG  10
C     ..................................................................DTSG  20
C                                                                       DTSG  30
C     SUBROUTINE DATSG                                                  DTSG  40
C                                                                       DTSG  50
C     PURPOSE                                                           DTSG  60
C        NDIM POINTS OF A GIVEN GENERAL TABLE ARE SELECTED AND          DTSG  70
C        ORDERED SUCH THAT ABS(ARG(I)-X).GE.ABS(ARG(J)-X) IF I.GT.J.    DTSG  80
C                                                                       DTSG  90
C     USAGE                                                             DTSG 100
C        CALL DATSG (X,Z,F,WORK,IROW,ICOL,ARG,VAL,NDIM)                 DTSG 110
C                                                                       DTSG 120
C     DESCRIPTION OF PARAMETERS                                         DTSG 130
C        X      - DOUBLE PRECISION SEARCH ARGUMENT.                    DTSG 140
C        Z      - DOUBLE PRECISION VECTOR OD ARGUMENT VALUES           DTSG 150
C                 (DIMENSION IROW).                                     DTSG 160
C        F      - IN CASE ICOL=1, F IS THE DOUBLE PRECISION VECTOR     DTSG 170
C                 OF FUNCTION VALUES (DIMENSION IROW).                  DTSG 180
C                 IN CASE ICOL=2, F IS A DOUBLE PRECISION IROW BY 2     DTSG 190
C                 MATRIX. THE FIRST COLUMN SPECIFIES VECTOR OF FUNC-    DTSG 200
C                 TION VALUES AND THE SECOND VECTOR OF DERIVATIVES.     DTSG 210
C        WORK   - DOUBLE PRECISION WORKING STORAGE (DIMENSION IROW).    DTSG 220
C        IROW   - THE DIMENSION OF VECTORS Z AND WORK AND OF EACH       DTSG 230
C                 COLUMN IN MATRIX F.                                   DTSG 240
C        ICOL   - THE NUMBER OF COLUMNS IN F (I.E. 1 OR 2).             DTSG 250
C        ARG    - RESULTING DOUBLE PRECISION VECTOR OF SELECTED AND     DTSG 260
C                 ORDERED ARGUMENT VALUES (DIMENSION NDIM).             DTSG 270
C        VAL    - RESULTING DOUBLE PRECISION VECTOR OF SELECTED         DTSG 280
C                 FUNCTION VALUES (DIMENSION NDIM) IN CASE ICOL=1.      DTSG 290
C                 IN CASE ICOL=2, VAL IS THE DOUBLE PRECISION VECTOR    DTSG 300
C                 OF FUNCTION AND DERIVATIVE VALUES (DIMENSION          DTSG 310
C                 2*NDIM) WHICH ARE STORED IN PAIRS (I.E. EACH FUNC-    DTSG 320
C                 TION VALUE IS FOLLOWED BY ITS DERIVATIVE VALUE).      DTSG 330
C        NDIM   - THE NUMBER OF POINTS WHICH MUST BE SELECTED OUT OF    DTSG 340
C                 THE GIVEN TABLE (Z,F).                                DTSG 350
C                                                                       DTSG 360
C     REMARKS                                                           DTSG 370
C        NO ACTION IN CASE IROW LESS THAN 1.                            DTSG 380
C        IF INPUT VALJE NDIM IS GREATER THAN IROW, THE PROGRAM          DTSG 390
C        SELECTS ONLY A MAXIMUM TABLE OF IROW POINTS.  THEREFORE THE    DTSG 400
C        USER OUGHT TO CHECK CORRESPONDENCE BETWEEN TABLE (ARG,VAL)     DTSG 410
C        AND ITS DIMENSION BY COMPARISON OF NDIM AND IROW, IN ORDER     DTSG 420
C        TO GET CORRECT RESULTS IN FURTHER WORK WITH TABLE (ARG,VAL).DTSG 430
C        THIS TEST MAY BE DONE BEFORE OR AFTER CALLING                  DTSG 440
C        SUBROUTINE DATSG.                                              DTSG 450
C        SUBROUTINE DATSG ESPECIALLY CAN BE USED FOR GENERATING THE     DTSG 460
C        TABLE (ARG,VAL) NEEDED IN SUBROUTINES DALI, DAHI, AND DACFI.DTSG 470
C                                                                       DTSG 480
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     DTSG 490
C        NONE                                                           DTSG 500
C                                                                       DTSG 510
C     METHOD                                                            DTSG 520
C        SELECTION IS DONE BY GENERATING THE VECTOR WORK WITH           DTSG 530
C        COMPONENTS WORK(I)=ABS(Z(I)-X) AND AT EACH OF THE NDIM STEPSDTSG 540
C        (OR IROW STEPS IF NDIM IS GREATER THAN IROW)                   DTSG 550
C        SEARCHING FOR THE SUBSCRIPT OF THE SMALLEST COMPONENT, WHICHDTSG 560
C        IS AFTERWARDS REPLACED BY A NUMBER GREATER THAN                DTSG 570
C        MAX(WORK(I)).                                                  DTSG 580
C                                                                       DTSG 590
C     ..................................................................DTSG 600
C                                                                       DTSG 610
      SUBROUTINE DATSG(X,Z,F,WORK,IROW,ICOL,ARG,VAL,NDIM)               DTSG 620
C                                                                       DTSG 630
C                                                                       DTSG 640
      DIMENSION Z(1),F(1),WORK(1),ARG(1),VAL(1)                         DTSG 650
      DOUBLE PRECISION X,Z,F,WORK,ARG,VAL,B,DELTA                       DTSG 660
      IF(IROW)11,11,1                                                   DTSG 670
    1 N=NDIM                                                            DTSG 680
C     IF N IS GREATER THAN IROW, N IS SET EQUAL TO IROW.                DTSG 690
      IF(N-IROW)3,3,2                                                   DTSG 700
    2 N=IROW                                                            DTSG 710
C                                                                       DTSG 720
C     GENERATION OF VECTOR WORK AND COMPUTATION OF ITS GREATEST ELEMENT.DTSG 730
    3 B=0.D0                                                            DTSG 740
      DO 5 I=1,IROW                                                     DTSG 750
      DELTA=DABS(Z(I)-X)                                                DTSG 760
      IF(DELTA-B)5,5,4                                                  DTSG 770
    4 B=DELTA                                                           DTSG 780
    5 WORK(I)=DELTA                                                     DTSG 790
C                                                                       DTSG 800
C     GENERATION OF TABLE (ARG,VAL)                                     DTSG 810
      B=B+1.D0                                                          DTSG 820
      DO 10 J=1,N                                                       DTSG 830
      DELTA=B                                                           DTSG 840
      DO 7 I=1,IROW                                                     DTSG 850
      IF(WORK(I)-DELTA)6,7,7                                            DTSG 860
    6 II=I                                                              DTSG 870
      DELTA=WORK(I)                                                     DTSG 880
    7 CONTINUE                                                          DTSG 890
      ARG(J)=Z(II)                                                      DTSG 900
      IF(ICOL-1)8,9,8                                                   DTSG 910
    8 VAL(2*J-1)=F(II)                                                  DTSG 920
      III=II+IROW                                                       DTSG 930
      VAL(2*J)=F(III)                                                   DTSG 940
      GOTO 10                                                           DTSG 950
    9 VAL(J)=F(II)                                                      DTSG 960
   10 WORK(II)=B                                                        DTSG 970
   11 RETURN                                                            DTSG 980
      END                                                               DTSG 990
```

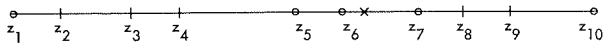
Subroutines ATSM and DATSM

These subroutines perform a table selection out of a given monotonic table. They have the same function as subroutines ATSG and DATSG with the difference that the given table is stored with monotonically increasing or decreasing arguments z_i.

Selection is done in two parts. In the first part, argument z_i next to search argument x is searched for by "binary search". This procedure (in case irow = 10) is shown in Figure 23.

The second part starts at z_{i_1}. At each of the n steps, the procedure decides by comparison of distances if the next step has to go to the right or to the left. Care must be taken that none of the selected subscripts i_j becomes greater than irow or less than one. Thus, in case n = 5, the sequence of argument values becomes $z_{i_1} = z_6$, $z_{i_2} = z_7$, $z_{i_3} = z_5$, $z_{i_4} = z_8$, $z_{i_5} = z_9$. The selected table is stored in the same way as in subroutine ATSG or DATSG.

If (by an error of the user) the dimension n of the table to be selected is greater than the dimension irow of the given table, the procedure selects only a maximum table of irow points. In order to avoid errors in further work with table (ARG, VAL), the user ought to check the correspondence between the selected table and its dimension by comparison of n and irow. This test may be done before or after calling subroutine ATSM or DATSM.



(1) x lies between z_1 and z_{10}.

(2) x lies between z_5 and z_{10}.

(3) x lies between z_5 and z_7.

(4) x lies between z_6 and z_7.

(5) Comparison of the two distances $|z_6 - x|$ and $|z_7 - x|$ shows that z_6 is next to x.

Figure 23. Binary search for argument z_i(irow=10)

```
C                                                                       ATSM  10
C     ...................................................................ATSM  20
C                                                                       ATSM  30
C        SUBROUTINE ATSM                                                ATSM  40
C                                                                       ATSM  50
C        PURPOSE                                                        ATSM  60
C           NDIM POINTS OF A GIVEN TABLE WITH MONOTONIC ARGUMENTS ARE   ATSM  70
C           SELECTED AND ORDERED SUCH THAT                              ATSM  80
C           ABS(ARG(I)-X).GE.ABS(ARG(J)-X) IF I.GT.J.                   ATSM  90
C                                                                       ATSM 100
C        USAGE                                                          ATSM 110
C           CALL ATSM (X,Z,F,IROW,ICOL,ARG,VAL,NDIM)                    ATSM 120
C                                                                       ATSM 130
C        DESCRIPTION OF PARAMETERS                                      ATSM 140
C           X      - THE SEARCH ARGUMENT.                               ATSM 150
C           Z      - THE VECTOR OF ARGUMENT VALUES (DIMENSION IROW).    ATSM 160
C                    THE ARGUMENT VALUES MUST BE STORED IN INCREASING   ATSM 170
C                    OR DECREASING SEQUENCE.                            ATSM 180
C           F      - IN CASE ICOL=1, F IS THE VECTOR OF FUNCTION VALUES ATSM 190
C                    (DIMENSION IROW).                                  ATSM 200
C                    IN CASE ICOL=2, F IS AN IROW BY 2 MATRIX. THE FIRSTATSM 210
C                    COLUMN SPECIFIES THE VECTOR OF FUNCTION VALUES AND ATSM 220
C                    THE SECOND THE VECTOR OF DERIVATIVES.              ATSM 230
C           IROW   - THE DIMENSION OF VECTOR Z AND OF EACH COLUMN       ATSM 240
C                    IN MATRIX F.                                       ATSM 250
C           ICOL   - THE NUMBER OF COLUMNS IN F (I.E. 1 OR 2).          ATSM 260
C           ARG    - THE RESULTING VECTOR OF SELECTED AND ORDERED       ATSM 270
C                    ARGUMENT VALUES (DIMENSION NDIM).                  ATSM 280
C           VAL    - THE RESULTING VECTOR OF SELECTED FUNCTION VALUES   ATSM 290
C                    (DIMENSION NDIM) IN CASE ICOL=1. IN CASE ICOL=2,   ATSM 300
C                    VAL IS THE VECTOR OF FUNCTION AND DERIVATIVE VALUESATSM 310
C                    (DIMENSION 2*NDIM) WHICH ARE STORED IN PAIRS (I.E. ATSM 320
C                    EACH FUNCTION VALUE IS FOLLOWED BY ITS DERIVATIVE  ATSM 330
C                    VALUE).                                            ATSM 340
C           NDIM   - THE NUMBER OF POINTS WHICH MUST BE SELECTED OUT OF ATSM 350
C                    THE GIVEN TABLE (Z,F).                             ATSM 360
C                                                                       ATSM 370
C        REMARKS                                                        ATSM 380
C           NO ACTION IN CASE IROW LESS THAN 1.                         ATSM 390
C           IF INPUT VALUE NDIM IS GREATER THAN IROW, THE PROGRAM       ATSM 400
C           SELECTS ONLY A MAXIMUM TABLE OF IROW POINTS.  THEREFORE THE ATSM 410
C           USER OUGHT TO CHECK CORRESPONDENCE BETWEEN TABLE (ARG,VAL)  ATSM 420
C           AND ITS DIMENSION BY COMPARISON OF NDIM AND IROW, IN ORDER  ATSM 430
C           TO GET CORRECT RESULTS IN FURTHER WORK WITH TABLE (ARG,VAL).ATSM 440
C           THIS TEST MAY BE DONE BEFORE OR AFTER CALLING               ATSM 450
C           SUBROUTINE ATSM.                                            ATSM 460
C           SUBROUTINE ATSM ESPECIALLY CAN BE USED FOR GENERATING THE   ATSM 470
C           TABLE (ARG,VAL) NEEDED IN SUBROUTINES ALI, AHI, AND ACFI.   ATSM 480
C                                                                       ATSM 490
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ATSM 500
C           NONE                                                        ATSM 510
C                                                                       ATSM 520
C        METHOD                                                         ATSM 530
C           SELECTION IS DONE BY SEARCHING THE SUBSCRIPT J OF THAT      ATSM 540
C           ARGUMENT, WHICH IS NEXT TO X (BINARY SEARCH).               ATSM 550
C           AFTERWARDS NEIGHBOURING ARGUMENT VALUES ARE TESTED AND      ATSM 560
C           SELECTED IN THE ABOVE SENSE.                                ATSM 570
C                                                                       ATSM 580
C     ...................................................................ATSM 590
C                                                                       ATSM 600
      SUBROUTINE ATSM(X,Z,F,IROW,ICOL,ARG,VAL,NDIM)                     ATSM 610
C                                                                       ATSM 620
C                                                                       ATSM 630
      DIMENSION Z(1),F(1),ARG(1),VAL(1)                                 ATSM 640
C                                                                       ATSM 650
C     CASE IROW=1 IS CHECKED OUT                                        ATSM 660
      IF(IROW-1)23,21,1                                                 ATSM 670
    1 N=NDIM                                                            ATSM 680
C                                                                       ATSM 690
C     IF N IS GREATER THAN IROW, N IS SET EQUAL TO IROW.                ATSM 700
      IF(N-IROW)3,3,2                                                   ATSM 710
    2 N=IROW                                                            ATSM 720
C                                                                       ATSM 730
C     CASE IROW.GE.2                                                    ATSM 740
C     SEARCHING FOR SUBSCRIPT J SUCH THAT Z(J) IS NEXT TO X.            ATSM 750
    3 IF(Z(IROW)-Z(1))5,4,4                                             ATSM 760
    4 J=IROW                                                            ATSM 770
      I=1                                                               ATSM 780
      GOTO 6                                                            ATSM 790
    5 I=IROW                                                            ATSM 800
      J=1                                                               ATSM 810
    6 K=(J+I)/2                                                         ATSM 820
      IF(X-Z(K))7,7,8                                                   ATSM 830
    7 J=K                                                               ATSM 840
      GOTO 9                                                            ATSM 850
    8 I=K                                                               ATSM 860
    9 IF(IABS(J-I)-1)10,10,6                                            ATSM 870
   10 IF(ABS(Z(J)-X)-ABS(Z(I)-X))12,12,11                               ATSM 880
   11 J=I                                                               ATSM 890
C                                                                       ATSM 900
C     TABLE SELECTION                                                   ATSM 910
   12 K=J                                                               ATSM 920
      JL=0                                                              ATSM 930
      JR=0                                                              ATSM 940
      DO 20 I=1,N                                                       ATSM 950
      ARG(I)=Z(K)                                                       ATSM 960
      IF(ICOL-1)14,14,13                                                ATSM 970
   13 VAL(2*I-1)=F(K)                                                   ATSM 980
      KK=K+IROW                                                         ATSM 990
      VAL(2*I)=F(KK)                                                    ATSM1000
      GOTO 15                                                           ATSM1010
   14 VAL(I)=F(K)                                                       ATSM1020
   15 JJR=J+JR                                                          ATSM1030
      IF(JJR-IROW)16,18,18                                              ATSM1040
   16 JJL=J-JL                                                          ATSM1050
      IF(JJL-1)19,19,17                                                 ATSM1060
   17 IF(ABS(Z(JJR+1)-X)-ABS(Z(JJL-1)-X))19,19,18                       ATSM1070
   18 JL=JL+1                                                           ATSM1080
      K=J-JL                                                            ATSM1090
      GOTO 20                                                           ATSM1100
   19 JR=JR+1                                                           ATSM1110
      K=J+JR                                                            ATSM1120
   20 CONTINUE                                                          ATSM1130
      RETURN                                                            ATSM1140
C                                                                       ATSM1150
C     CASE IROW=1                                                       ATSM1160
   21 ARG(1)=Z(1)                                                       ATSM1170
      VAL(1)=F(1)                                                       ATSM1180
      IF(ICOL-2)23,22,23                                                ATSM1190
   22 VAL(2)=F(2)                                                       ATSM1200
   23 RETURN                                                            ATSM1210
      END                                                               ATSM1220
```

```
C
C     ..................................................................
C
C        SUBROUTINE DATSM
C
C        PURPOSE
C           NDIM POINTS OF A GIVEN TABLE WITH MONOTONIC ARGUMENTS ARE
C           SELECTED AND ORDERED SUCH THAT
C           ABS(ARG(I)-X).GE.ABS(ARG(J)-X) IF I.GT.J.
C
C        USAGE
C           CALL DATSM (X,Z,F,IROW,ICOL,ARG,VAL,NDIM)
C
C        DESCRIPTION OF PARAMETERS
C           X       - DOUBLE PRECISION SEARCH ARGUMENT.
C           Z       - DOUBLE PRECISION VECTOR OF ARGUMENT VALUES (DIMEN-
C                     SION IROW). THE ARGUMENT VALUES MUST BE STORED IN
C                     INCREASING OR DECREASING SEQUENCE.
C           F       - IN CASE ICOL=1, F IS THE DOUBLE PRECISION VECTOR
C                     OF FUNCTION VALUES (DIMENSION IROW).
C                     IN CASE ICOL=2, F IS A DOUBLE PRECISION IROW BY 2
C                     MATRIX. THE FIRST COLUMN SPECIFIES VECTOR OF FUNC-
C                     TION VALUES AND THE SECOND VECTOR OF DERIVATIVES.
C           IROW    - THE DIMENSION OF VECTOR Z AND OF EACH COLUMN
C                     IN MATRIX F.
C           ICOL    - THE NUMBER OF COLUMNS IN F (I.E. 1 OR 2).
C           ARG     - RESULTING DOUBLE PRECISION VECTOR OF SELECTED AND
C                     ORDERED ARGUMENT VALUES (DIMENSION NDIM).
C           VAL     - RESULTING DOUBLE PRECISION VECTOR OF SELECTED
C                     FUNCTION VALUES (DIMENSION NDIM) IN CASE ICOL=1.
C                     IN CASE ICOL=2, VAL IS THE DOUBLE PRECISION VECTOR
C                     OF FUNCTION AND DERIVATIVE VALUES (DIMENSION
C                     2*NDIM) WHICH ARE STORED IN PAIRS (I.E. EACH FUNC-
C                     TION VALUE IS FOLLOWED BY ITS DERIVATIVE VALUE).
C           NDIM    - THE NUMBER OF POINTS WHICH MUST BE SELECTED OUT OF
C                     THE GIVEN TABLE (Z,F).
C
C        REMARKS
C           NO ACTION IN CASE IROW LESS THAN 1.
C           IF INPUT VALUE NDIM IS GREATER THAN IROW, THE PROGRAM
C           SELECTS ONLY A MAXIMUM TABLE OF IROW POINTS.  THEREFORE THE
C           USER OUGHT TO CHECK CORRESPONDENCE BETWEEN TABLE (ARG,VAL)
C           AND ITS DIMENSION BY COMPARISON OF NDIM AND IROW, IN ORDER
C           TO GET CORRECT RESULTS IN FURTHER WORK WITH TABLE (ARG,VAL).
C           THIS TEST MAY BE DONE BEFORE OR AFTER CALLING
C           SUBROUTINE DATSM.
C           SUBROUTINE DATSM ESPECIALLY CAN BE USED FOR GENERATING THE
C           TABLE (ARG,VAL) NEEDED IN SUBROUTINES DALI, DAHI, AND DACFI.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           SELECTION IS DONE BY SEARCHING THE SUBSCRIPT J OF THAT
C           ARGUMENT, WHICH IS NEXT TO X (BINARY SEARCH).
C           AFTERWARDS NEIGHBOURING ARGUMENT VALUES ARE TESTED AND
C           SELECTED IN THE ABOVE SENSE.
C
C     ..................................................................
C
      SUBROUTINE DATSM(X,Z,F,IROW,ICOL,ARG,VAL,NDIM)
C
C
      DIMENSION Z(1),F(1),ARG(1),VAL(1)
      DOUBLE PRECISION X,Z,F,ARG,VAL
C
C     CASE IROW=1 IS CHECKED OUT
      IF(IROW-1)23,21,1
    1 N=NDIM
C
C     IF N IS GREATER THAN IROW, N IS SET EQUAL TO IROW.
      IF(N-IROW)3,3,2
    2 N=IROW
C
C     CASE IROW.GE.2
C     SEARCHING FOR SUBSCRIPT J SUCH THAT Z(J) IS NEXT TO X.
    3 IF(Z(IROW)-Z(1))5,4,4
    4 J=IROW
      I=1
      GOTO 6
    5 I=IROW
      J=1
    6 K=(J+I)/2
      IF(X-Z(K))7,7,8
    7 J=K
      GOTO 9
    8 I=K
    9 IF(IABS(J-I)-1)10,10,6
   10 IF(DABS(Z(J)-X)-DABS(Z(I)-X))12,12,11
   11 J=I
C
C     TABLE SELECTION
   12 K=J
      JL=0
      JR=0
      DO 20 I=1,N
      ARG(I)=Z(K)
      IF(ICOL-1)14,14,13
   13 VAL(2*I-1)=F(K)
      KK=K+IROW
      VAL(2*I)=F(KK)
      GOTO 15
   14 VAL(I)=F(K)
   15 JJR=J+JR
      IF(JJR-IROW)16,18,18
   16 JJL=J-JL
      IF(JJL-1)19,19,17
   17 IF(DABS(Z(JJR+1)-X)-DABS(Z(JJL-1)-X))19,19,18
   18 JL=JL+1
      K=J-JL
      GOTO 20
   19 JR=JR+1
      K=J+JR
   20 CONTINUE
      RETURN
C
C     CASE IROW=1
   21 ARG(1)=Z(1)
      VAL(1)=F(1)
      IF(ICOL-2)23,22,23
   22 VAL(2)=F(2)
   23 RETURN
      END
```

Subroutines ATSE and DATSE

These subroutines perform a table selection out of a given equidistant table. They have the same function as subroutines ATSG and DATSG with the difference that the given table is stored with equidistant arguments. Instead of vector Z of argument values, subroutines ATSE and DATSE need only the starting argument z_s and the increment of arguments dz.

Selection is done in two parts. In the first part, the subscript i_1 of the argument next to search argument x is computed, using the formula

$$i_1 = \left[\frac{x - z_s}{dz} + 1.5 \right]$$

where [a] means "integer part of a".

The second part starts at z_{i_1} and goes one step to the left (right), one to the right (left), and so on until all n arguments are selected. Care must be taken that none of the selected subscripts i_j becomes greater than irow or less than one. In the example shown in Figure 24, the sequence (z_{i_j}) becomes

$z_{i_1} = z_6$, $z_{i_2} = z_5$, $z_{i_3} = z_7$, $z_{i_4} = z_4$, $z_{i_5} = z_8$, $z_{i_6} = z_3$, $z_{i_7} = z_2$.

The selected table is stored in the same way as in subroutine ATSG or DATSG.

If, by a user error, the dimension n of the table to be selected is greater than the dimension irow of the given table, the procedure selects only a maximum table of irow points. In order to avoid errors in further work with table (ARG, VAL), the user ought to check the correspondence between the selected table and its dimension by comparison of n and irow. This test may be done before or after calling subroutine ATSE or DATSE.

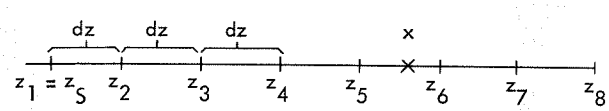


Figure 24. Generation of sequence (z_{i_i}) in case irow=8, n=7

```
C                                                                       ATSE  10
C     ..................................................................ATSE  20
C                                                                       ATSE  30
C        SUBROUTINE ATSE                                                ATSE  40
C                                                                       ATSE  50
C        PURPOSE                                                        ATSE  60
C           NDIM POINTS OF A GIVEN TABLE WITH EQUIDISTANT ARGUMENTS ARE ATSE  70
C           SELECTED AND ORDERED SUCH THAT                              ATSE  80
C           ABS(ARG(I)-X).GE.ABS(ARG(J)-X) IF I.GT.J.                   ATSE  90
C                                                                       ATSE 100
C        USAGE                                                          ATSE 110
C           CALL ATSE (X,ZS,DZ,F,IROW,ICOL,ARG,VAL,NDIM)                ATSE 120
C                                                                       ATSE 130
C        DESCRIPTION OF PARAMETERS                                      ATSE 140
C           X      - THE SEARCH ARGUMENT.                               ATSE 150
C           ZS     - THE STARTING VALUE OF ARGUMENTS.                   ATSE 160
C           DZ     - THE INCREMENT OF ARGUMENT VALUES.                  ATSE 170
C           F      - IN CASE ICOL=1, F IS THE VECTOR OF FUNCTION VALUES ATSE 180
C                    (DIMENSION IROW).                                  ATSE 190
C                    IN CASE ICOL=2, F IS AN IROW BY 2 MATRIX. THE FIRSTATSE 200
C                    COLUMN SPECIFIES THE VECTOR OF FUNCTION VALUES AND ATSE 210
C                    THE SECOND THE VECTOR OF DERIVATIVES.              ATSE 220
C           IROW   - THE DIMENSION OF EACH COLUMN IN MATRIX F.          ATSE 230
C           ICOL   - THE NUMBER OF COLUMNS IN F (I.E. 1 OR 2).          ATSE 240
C           ARG    - THE RESULTING VECTOR OF SELECTED AND ORDERED       ATSE 250
C                    ARGUMENT VALUES (DIMENSION NDIM).                  ATSE 260
C           VAL    - THE RESULTING VECTOR OF SELECTED FUNCTION VALUES   ATSE 270
C                    (DIMENSION NDIM) IN CASE ICOL=1. IN CASE ICOL=2,   ATSE 280
C                    VAL IS THE VECTOR OF FUNCTION AND DERIVATIVE VALUESATSE 290
C                    (DIMENSION 2*NDIM) WHICH ARE STORED IN PAIRS (I.E. ATSE 300
C                    EACH FUNCTION VALUE IS FOLLOWED BY ITS DERIVATIVE  ATSE 310
C                    VALUE).                                            ATSE 320
C           NDIM   - THE NUMBER OF POINTS WHICH MUST BE SELECTED OUT OF ATSE 330
C                    THE GIVEN TABLE.                                   ATSE 340
C                                                                       ATSE 350
C        REMARKS                                                        ATSE 360
C           NO ACTION IN CASE IROW LESS THAN 1.                         ATSE 370
C           IF INPUT VALUE NDIM IS GREATER THAN IROW, THE PROGRAM       ATSE 380
C           SELECTS ONLY A MAXIMUM TABLE OF IROW POINTS.  THEREFORE THE ATSE 390
C           USER OUGHT TO CHECK CORRESPONDENCE BETWEEN TABLE (ARG,VAL)  ATSE 400
C           AND ITS DIMENSION BY COMPARISON OF NDIM AND IROW, IN ORDER  ATSE 410
C           TO GET CORRECT RESULTS IN FURTHER WORK WITH TABLE (ARG,VAL).ATSE 420
C           THIS TEST MAY BE DONE BEFORE OR AFTER CALLING               ATSE 430
C           SUBROUTINE ATSE.                                            ATSE 440
C           SUBROUTINE ATSE ESPECIALLY CAN BE USED FOR GENERATING THE   ATSE 450
C           TABLE (ARG,VAL) NEEDED IN SUBROUTINES ALI, AHI, AND ACFI.   ATSE 460
C                                                                       ATSE 470
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ATSE 480
C           NONE                                                        ATSE 490
C                                                                       ATSE 500
C        METHOD                                                         ATSE 510
C           SELECTION IS DONE BY COMPUTING THE SUBSCRIPT J OF THAT      ATSE 520
C           ARGUMENT, WHICH IS NEXT TO X.                               ATSE 530
C           AFTERWARDS NEIGHBOURING ARGUMENT VALUES ARE TESTED AND      ATSE 540
C           SELECTED IN THE ABOVE SENSE.                                ATSE 550
C                                                                       ATSE 560
C     ..................................................................ATSE 570
C                                                                       ATSE 580
      SUBROUTINE ATSE(X,ZS,DZ,F,IROW,ICOL,ARG,VAL,NDIM)                 ATSE 590
C                                                                       ATSE 600
C                                                                       ATSE 610
      DIMENSION F(1),ARG(1),VAL(1)                                      ATSE 620
      IF(IROW-1)19,17,1                                                 ATSE 630
C                                                                       ATSE 640
C     CASE DZ=0 IS CHECKED OUT                                          ATSE 650
    1 IF(DZ)2,17,2                                                      ATSE 660
    2 N=NDIM                                                            ATSE 670
C                                                                       ATSE 680
C     IF N IS GREATER THAN IROW, N IS SET EQUAL TO IROW.                ATSE 690
      IF(N-IROW)4,4,3                                                   ATSE 700
    3 N=IROW                                                            ATSE 710
C                                                                       ATSE 720
C     COMPUTATION OF STARTING SUBSCRIPT J.                              ATSE 730
    4 J=(X-ZS)/DZ+1.5                                                   ATSE 740
      IF(J)5,5,6                                                        ATSE 750
    5 J=1                                                               ATSE 760
    6 IF(J-IROW)8,8,7                                                   ATSE 770
    7 J=IROW                                                            ATSE 780
C                                                                       ATSE 790
C     GENERATION OF TABLE ARG,VAL IN CASE DZ.NE.0.                      ATSE 800
    8 II=J                                                              ATSE 810
      JL=0                                                              ATSE 820
      JR=0                                                              ATSE 830
      DO 16 I=1,N                                                       ATSE 840
      ARG(I)=ZS+FLOAT(II-1)*DZ                                          ATSE 850
      IF(ICOL-2)9,10,10                                                 ATSE 860
    9 VAL(I)=F(II)                                                      ATSE 870
      GOTO 11                                                           ATSE 880
   10 VAL(2*I-1)=F(II)                                                  ATSE 890
      III=II+IROW                                                       ATSE 900
      VAL(2*I)=F(III)                                                   ATSE 910
   11 IF(J+JR-IROW)12,15,12                                             ATSE 920
   12 IF(J-JL-1)13,14,13                                                ATSE 930
   13 IF((ARG(I)-X)*DZ)14,15,15                                         ATSE 940
   14 JR=JR+1                                                           ATSE 950
      II=J+JR                                                           ATSE 960
      GOTO 16                                                           ATSE 970
   15 JL=JL+1                                                           ATSE 980
      II=J-JL                                                           ATSE 990
   16 CONTINUE                                                          ATSE1000
      RETURN                                                            ATSE1010
C                                                                       ATSE1020
C     CASE DZ=0                                                         ATSE1030
   17 ARG(1)=ZS                                                         ATSE1040
      VAL(1)=F(1)                                                       ATSE1050
      IF(ICOL-2)19,19,18                                                ATSE1060
   18 VAL(2)=F(2)                                                       ATSE1070
   19 RETURN                                                            ATSE1080
      END                                                               ATSE1090
```

```
C                                                                       DTSE  10
C     ..................................................................DTSE  20
C                                                                       DTSE  30
C        SUBROUTINE DATSE                                               DTSE  40
C                                                                       DTSE  50
C        PURPOSE                                                        DTSE  60
C           NDIM POINTS OF A GIVEN TABLE WITH EQUIDISTANT ARGUMENTS ARE DTSE  70
C           SELECTED AND ORDERED SUCH THAT                              DTSE  80
C           ABS(ARG(I)-X).GE.ABS(ARG(J)-X) IF I.GT.J.                   DTSE  90
C                                                                       DTSE 100
C        USAGE                                                          DTSE 110
C           CALL DATSE (X,ZS,DZ,F,IROW,ICOL,ARG,VAL,NDIM)               DTSE 120
C                                                                       DTSE 130
C        DESCRIPTION OF PARAMETERS                                      DTSE 140
C           X      - DOUBLE PRECISION SEARCH ARGUMENT.                  DTSE 150
C           ZS     - DOUBLE PRECISION STARTING VALUE OF ARGUMENTS.      DTSE 160
C           DZ     - DOUBLE PRECISION INCREMENT OF ARGUMENT VALUES.     DTSE 170
C           F      - IN CASE ICOL=1, F IS THE DOUBLE PRECISION VECTOR   DTSE 180
C                    OF FUNCTION VALUES (DIMENSION IROW).               DTSE 190
C                    IN CASE ICOL=2, F IS A DOUBLE PRECISION IROW BY 2  DTSE 200
C                    MATRIX. THE FIRST COLUMN SPECIFIES VECTOR OF FUNC- DTSE 210
C                    TION VALUES AND THE SECOND VECTOR OF DERIVATIVES.  DTSE 220
C           IROW   - THE DIMENSION OF EACH COLUMN IN MATRIX F.          DTSE 230
C           ICOL   - THE NUMBER OF COLUMNS IN F (I.E. 1 OR 2).          DTSE 240
C           ARG    - RESULTING DOUBLE PRECISION VECTOR OF SELECTED AND  DTSE 250
C                    ORDERED ARGUMENT VALUES (DIMENSION NDIM).          DTSE 260
C           VAL    - RESULTING DOUBLE PRECISION VECTOR OF SELECTED      DTSE 270
C                    FUNCTION VALUES (DIMENSION NDIM) IN CASE ICOL=1.   DTSE 280
C                    IN CASE ICOL=2, VAL IS THE DOUBLE PRECISION VECTOR DTSE 290
C                    OF FUNCTION AND DERIVATIVE VALUES (DIMENSION       DTSE 300
C                    2*NDIM) WHICH ARE STORED IN PAIRS (I.E. EACH FUNC- DTSE 310
C                    TION VALUE IS FOLLOWED BY ITS DERIVATIVE VALUE).   DTSE 320
C           NDIM   - THE NUMBER OF POINTS WHICH MUST BE SELECTED OUT OF DTSE 330
C                    THE GIVEN TABLE.                                   DTSE 340
C                                                                       DTSE 350
C        REMARKS                                                        DTSE 360
C           NO ACTION IN CASE IROW LESS THAN 1.                         DTSE 370
C           IF INPUT VALUE NDIM IS GREATER THAN IROW, THE PROGRAM       DTSE 380
C           SELECTS ONLY A MAXIMUM TABLE OF IROW POINTS.  THEREFORE THE DTSE 390
C           USER OUGHT TO CHECK CORRESPONDENCE BETWEEN TABLE (ARG,VAL)  DTSE 400
C           AND ITS DIMENSION BY COMPARISON OF NDIM AND IROW, IN ORDER  DTSE 410
C           TO GET CORRECT RESULTS IN FURTHER WORK WITH TABLE (ARG,VAL).DTSE 420
C           THIS TEST MAY BE DONE BEFORE OR AFTER CALLING               DTSE 430
C           SUBROUTINE DATSE.                                           DTSE 440
C           SUBROUTINE DATSE ESPECIALLY CAN BE USED FOR GENERATING THE  DTSE 450
C           TABLE (ARG,VAL) NEEDED IN SUBROUTINES DALI, DAHI, AND DACFI.DTSE 460
C                                                                       DTSE 470
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTSE 480
C           NONE                                                        DTSE 490
C                                                                       DTSE 500
C        METHOD                                                         DTSE 510
C           SELECTION IS DONE BY COMPUTING THE SUBSCRIPT J OF THAT      DTSE 520
C           ARGUMENT, WHICH IS NEXT TO X.                               DTSE 530
C           AFTERWARDS NEIGHBOURING ARGUMENT VALUES ARE TESTED AND      DTSE 540
C           SELECTED IN THE ABOVE SENSE.                                DTSE 550
C                                                                       DTSE 560
C     ..................................................................DTSE 570
C                                                                       DTSE 580
      SUBROUTINE DATSE(X,ZS,DZ,F,IROW,ICOL,ARG,VAL,NDIM)                DTSE 590
C                                                                       DTSE 600
C                                                                       DTSE 610
      DIMENSION F(1),ARG(1),VAL(1)                                      DTSE 620
      DOUBLE PRECISION X,ZS,DZ,F,ARG,VAL                                DTSE 630
      IF(IROW-1)19,17,1                                                 DTSE 640
C                                                                       DTSE 650
C     CASE DZ=0 IS CHECKED OUT                                          DTSE 660
    1 IF(DZ)2,17,2                                                      DTSE 670
    2 N=NDIM                                                            DTSE 680
C                                                                       DTSE 690
C     IF N IS GREATER THAN IROW, N IS SET EQUAL TO IROW.                DTSE 700
      IF(N-IROW)4,4,3                                                   DTSE 710
    3 N=IROW                                                            DTSE 720
C                                                                       DTSE 730
C     COMPUTATION OF STARTING SUBSCRIPT J.                              DTSE 740
    4 J=(X-ZS)/DZ+1.500                                                 DTSE 750
      IF(J)5,5,6                                                        DTSE 760
    5 J=1                                                               DTSE 770
    6 IF(J-IROW)8,8,7                                                   DTSE 780
    7 J=IROW                                                            DTSE 790
C                                                                       DTSE 800
C     GENERATION OF TABLE ARG,VAL IN CASE DZ.NE.0.                      DTSE 810
    8 II=J                                                              DTSE 820
      JL=0                                                              DTSE 830
      JR=0                                                              DTSE 840
      DO 16 I=1,N                                                       DTSE 850
      ARG(I)=ZS+DFLOAT(II-1)*DZ                                         DTSE 860
      IF(ICOL-2)9,10,10                                                 DTSE 870
    9 VAL(I)=F(II)                                                      DTSE 880
      GOTO 11                                                           DTSE 890
   10 VAL(2*I-1)=F(II)                                                  DTSE 900
      III=II+IROW                                                       DTSE 910
      VAL(2*I)=F(III)                                                   DTSE 920
   11 IF(J+JR-IROW)12,15,12                                             DTSE 930
   12 IF(J-JL-1)13,14,13                                                DTSE 940
   13 IF((ARG(I)-X)*DZ)14,15,15                                         DTSE 950
   14 JR=JR+1                                                           DTSE 960
      II=J+JR                                                           DTSE 970
      GOTO 16                                                           DTSE 980
   15 JL=JL+1                                                           DTSE 990
      II=J-JL                                                           DTSE1000
   16 CONTINUE                                                          DTSE1010
      RETURN                                                            DTSE1020
C                                                                       DTSE1030
C     CASE DZ=0                                                         DTSE1040
   17 ARG(1)=ZS                                                         DTSE1050
      VAL(1)=F(1)                                                       DTSE1060
      IF(ICOL-2)19,19,18                                                DTSE1070
   18 VAL(2)=F(2)                                                       DTSE1080
   19 RETURN                                                            DTSE1090
      END                                                               DTSE1100
```

Subroutine SG13 and DSG13

These subroutines compute a vector $Z = (z_1, \ldots, z_n)$ of smoothed function values, given vectors $X = (x_1, \ldots, x_n)$ of argument values and $Y = (y_1, \ldots, y_n)$ of corresponding function values. Except at the endpoints x_1 and x_n, each smoothed value z_i is obtained by evaluating at x_i the least-squares polynomial of degree 1 relevant to the three successive points (x_{i-1}, y_{i-1}), (x_i, y_i), and (x_{i+1}, y_{i+1}).

1. Mathematical background

For $i = 3, \ldots, n$ we must find m_i and b_i such that

$$w_i(x) = m_i x + b_i \tag{1}$$

gives the least-squares fit to the points (x_{i-1}, y_{i-2}), (x_{i-1}, y_{i-1}), and (x_i, y_i). The problem, then, is to minimize

$$F(m_i, b_i) = \sum_{k=0}^{2} \left[w_i(x_{i-k}) - y_{i-k} \right]^2$$

This minimum will occur when

$$\frac{\partial F}{\partial b_i} = 0 \text{ and } \frac{\partial F}{\partial m_i} = 0 \tag{2}$$

Now

$$\frac{\partial F}{\partial b_i} = 2 \sum_{k=0}^{2} \left[w_i(x_{i-k}) - y_{i-k} \right]$$

and

$$\frac{\partial F}{\partial m_i} = 2 \sum_{k=0}^{2} x_{i-k} \left[w_i(x_{i-k}) - y_{i-k} \right] \tag{3}$$

Solving equations (2) and (3) yields:

$$m_i = \frac{\sum_{k=0}^{2} x_{i-k} y_{i-k} - 1/3 \left(\sum_{k=0}^{2} x_{i-k} \right) \left(\sum_{k=0}^{2} y_{i-k} \right)}{\sum_{k=0}^{2} x_{i-k}^2 - 1/3 \left(\sum_{k=0}^{2} x_{i-k} \right)^2} \tag{4}$$

and

$$b_i = \frac{i}{3} \sum_{k=0}^{2} \left(y_{i-k} - m_i x_{i-k} \right) \tag{5}$$

Letting:

$$\left.\begin{aligned} \bar{y}_i &= \frac{i}{3} \sum_{k=0}^{2} y_{i-k}, \quad \bar{x}_i = \frac{i}{3} \sum_{k=0}^{2} x_{i-k}, \\ t_{i,k} &= x_{i-k} - \bar{x}_i \quad \text{and} \quad v_{i,k} = y_{i-k} - \bar{y}_i \end{aligned}\right\} \tag{6}$$

we may rewrite (4) and (5) as:

$$m_i = \frac{\sum_{k=0}^{2} t_{i,k} v_{i,k}}{\sum_{k=0}^{2} t_{i,k}^2} \tag{7}$$

and

$$b_i = \bar{y}_i - m_i \bar{x}_i \tag{8}$$

Using (8) in (1) gives

$$w_i(x) = m_i(x - \bar{x}_i) + \bar{y}_i$$

where m_i is as in (7).

The desired smoothed values z_i are given by:

$$z_i = \begin{cases} w_3(x_1) = m_3 t_{3,2} + \bar{y}_3 & \text{if } i=1 \\ w_{i+1}(x_i) = m_{i+1} t_{i+1,1} + \bar{y}_{i+1} & \text{if } i=2, \ldots, n-1 \\ w_n(x_n) = m_n t_{n,0} + \bar{y}_n & \text{if } i=n \end{cases} \tag{9}$$

2. Programming considerations

The subroutines compute the z_i in serial order according to (9). If $n<3$, there is no computation, and the error parameter IER is set to -1. Otherwise IER is set to zero at the end of the computation. The output vector Z can have the same storage allocation as X or Y.

For reference see Hildebrand, F. B., Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp 258-311.

```
C
C     ..................................................................
C
C        SUBROUTINE SG13
C
C        PURPOSE
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN
C           VECTORS OF ARGUMENT VALUES AND CORRESPONDING FUNCTION
C           VALUES.
C
C        USAGE
C           CALL SG13(X,Y,Z,NDIM,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      -  GIVEN VECTOR OF ARGUMENT VALUES (DIMENSION NDIM)
C           Y      -  GIVEN VECTOR OF FUNCTION VALUES CORRESPONDING TO X
C                     (DIMENSION NDIM)
C           Z      -  RESULTING VECTOR OF SMOOTHED FUNCTION VALUES
C                     (DIMENSION NDIM)
C           NDIM   -  DIMENSION OF VECTORS X,Y,AND Z
C           IER    -  RESULTING ERROR PARAMETER
C                     IER = -1  - NDIM IS LESS THAN 3
C                     IER =  0  - NO ERROR
C
C        REMARKS
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.
C           (2)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS X OR Y.  IF
C                 X OR Y IS DISTINCT FROM Z, THEN IT IS NOT DESTROYED.
C
C        SUBROUTINES AND SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), EACH SMOOTHED
C           VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE LEAST-
C           SQUARES POLYNOMIAL OF DEGREE 1 RELEVANT TO THE 3 SUCCESSIVE
C           POINTS (X(I+K),Y(I+K)) K = -1,0,1.(SEE HILDEBRAND, F.B..
C           INTRODUCTION  TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/
C           TORONTO/LONDON, 1956, PP.258-311.)
C
C     ..................................................................
C
      SUBROUTINE SG13(X,Y,Z,NDIM,IER)
C
C
      DIMENSION X(1),Y(1),Z(1)
C
C        TEST OF DIMENSION
      IF(NDIM-3)7,1,1
C
C        START LOOP
    1 DO 6 I=3,NDIM
      XM=.3333333*(X(I-2)+X(I-1)+X(I))
      YM=.3333333*(Y(I-2)+Y(I-1)+Y(I))
      T1=X(I-2)-XM
      T2=X(I-1)-XM
      T3=X(I)-XM
      XM=T1*T1+T2*T2+T3*T3
      IF(XM)3,3,2
    2 XM=(T1*(Y(I-2)-YM)+T2*(Y(I-1)-YM)+T3*(Y(I)-YM))/XM
C
C        CHECK FIRST POINT
    3 IF(I-3)4,4,5
    4 H=XM*T1+YM
    5 Z(I-2)=H
    6 H=XM*T2+YM
C        END OF LOOP
C
C        UPDATE LAST TWO COMPONENTS
      Z(NDIM-1)=H
      Z(NDIM)=XM*T3+YM
      IER=0
      RETURN
C
C        ERROR EXIT IN CASE NDIM IS LESS THAN 3
    7 IER=-1
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DSG13
C
C        PURPOSE
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN
C           VECTORS OF ARGUMENT VALUES AND CORRESPONDING FUNCTION
C           VALUES.
C
C        USAGE
C           CALL DSG13(X,Y,Z,NDIM,IER)
C
C        DESCRIPTION OF PARAMETERS
C           X      -  GIVEN VECTOR OF DOUBLE PRECISION ARGUMENT VALUES
C                     (DIMENSION NDIM)
C           Y      -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES
C                     CORRESPONDING TO X (DIMENSION NDIM)
C           Z      -  RESULTING VECTOR OF DOUBLE PRECISION SMOOTHED
C                     FUNCTION VALUES (DIMENSION NDIM)
C           NDIM   -  DIMENSION OF VECTORS X,Y,AND Z
C           IER    -  RESULTING ERROR PARAMETER
C                     IER = -1  - NDIM IS LESS THAN 3
C                     IER =  0  - NO ERROR
C
C        REMARKS
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.
C           (2)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS X OR Y.  IF
C                 X OR Y IS DISTINCT FROM Z, THEN IT IS NOT DESTROYED.
C
C        SUBROUTINES AND SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), EACH SMOOTHED
C           VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE LEAST-
C           SQUARES POLYNOMIAL OF DEGREE 1 RELEVANT TO THE 3 SUCCESSIVE
C           POINTS (X(I+K),Y(I+K)) K = -1,0,1.(SEE HILDEBRAND, F.B..
C           INTRODUCTION  TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/
C           TORONTO/LONDON, 1956, PP.258-311.)
C
C     ..................................................................
C
      SUBROUTINE DSG13(X,Y,Z,NDIM,IER)
C
C
      DIMENSION X(1),Y(1),Z(1)
      DOUBLE PRECISION X,Y,Z,XM,YM,T1,T2,T3,H
C
C        TEST OF DIMENSION
      IF(NDIM-3)7,1,1
C
C        START LOOP
    1 DO 6 I=3,NDIM
      XM=.3333333333333333D0*(X(I-2)+X(I-1)+X(I))
      YM=.3333333333333333D0*(Y(I-2)+Y(I-1)+Y(I))
      T1=X(I-2)-XM
      T2=X(I-1)-XM
      T3=X(I)-XM
      XM=T1*T1+T2*T2+T3*T3
      IF(XM)3,3,2
    2 XM=(T1*(Y(I-2)-YM)+T2*(Y(I-1)-YM)+T3*(Y(I)-YM))/XM
C
C        CHECK FIRST POINT
    3 IF(I-3)4,4,5
    4 H=XM*T1+YM
    5 Z(I-2)=H
    6 H=XM*T2+YM
C        END OF LOOP
C
C        UPDATE LAST TWO COMPONENTS
      Z(NDIM-1)=H
      Z(NDIM)=XM*T3+YM
      IER=0
      RETURN
C
C        ERROR EXIT IN CASE NDIM IS LESS THAN 3
    7 IER=-1
      RETURN
      END
```

Subroutines SE13 and DSE13

These subroutines compute a vector $Z = (z_1, \ldots, z_n)$ of smoothed function values, given a vector $Y = (y_1, \ldots, y_n)$ of function values whose entries y_i correspond to n equidistantly spaced argument values x_i, with $x_i - x_{i-1} = h$ for $i = 2, \ldots, n$. Except at the endpoints x_1 and x_n, each smoothed value z_i is obtained by evaluating at x_i the least-squares polynomial of degree 1 relevant to the three successive points (x_{i+k}, y_{i+k}), $k = -1, 0, 1$.

1. Mathematical background

The procedure is exactly that described for subroutines SG13 and DSG13, but here we have the additional relation $x_i - x_{i-1} = h$, a constant, for $i = 2, \ldots, n$. This leads to the following expressions for the z_i:

$$z_i = \begin{cases} \frac{1}{6}(5y_1 + 2y_2 - y_3) & \text{if } i=1 \\ \frac{1}{3}(y_{i-1} + y_i + y_{i+1}) & \text{if } i=2,\ldots,n-1 \\ \frac{1}{6}(-y_{n-2} + 2y_{n-1} + 5y_n) & \text{if } i=n \end{cases} \quad (1)$$

2. Programming considerations

The subroutines compute the z_i in serial order according to (1). If $n < 3$, there is no computation, and the error parameter IER is set to -1. Otherwise IER is set to zero at the end of the computation. The output vector Z can have the same storage allocation as Y.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp 295-302.

```
C                                                                       SE13  10
C     ..................................................................SE13  20
C                                                                       SE13  30
C        SUBROUTINE SE13                                                SE13  40
C                                                                       SE13  50
C        PURPOSE                                                        SE13  60
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN A     SE13  70
C           VECTOR OF FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO       SE13  80
C           EQUIDISTANTLY SPACED ARGUMENT VALUES.                       SE13  90
C                                                                       SE13 100
C        USAGE                                                          SE13 110
C           CALL SE13(Y,Z,NDIM,IER)                                     SE13 120
C                                                                       SE13 130
C        DESCRIPTION OF PARAMETERS                                      SE13 140
C           Y     -  GIVEN VECTOR OF FUNCTION VALUES (DIMENSION NDIM)   SE13 150
C           Z     -  RESULTING VECTOR OF SMOOTHED FUNCTION VALUES       SE13 160
C                    (DIMENSION NDIM)                                   SE13 170
C           NDIM  -  DIMENSION OF VECTORS Y AND Z                       SE13 180
C           IER   -  RESULTING ERROR PARAMETER                          SE13 190
C                    IER = -1  - NDIM IS LESS THAN 3                    SE13 200
C                    IER =  0  - NO ERROR                               SE13 210
C                                                                       SE13 220
C        REMARKS                                                        SE13 230
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.               SE13 240
C           (2)  Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y     SE13 250
C                IS DISTINCT FROM Z, THEN IT IS NOT DESTROYED.          SE13 260
C                                                                       SE13 270
C        SUBROUTINES AND SUBPROGRAMS REQUIRED                           SE13 280
C           NONE                                                        SE13 290
C                                                                       SE13 300
C        METHOD                                                         SE13 310
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    SE13 320
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), EACH SMOOTHED     SE13 330
C           VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE LEAST-     SE13 340
C           SQUARES POLYNOMIAL OF DEGREE 1 RELEVANT TO THE 3 SUCCESSIVE SE13 350
C           POINTS (X(I+K),Y(I+K)) K = -1,0,1.  (SEE HILDEBRAND, F.B.,  SE13 360
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ SE13 370
C           TORONTO/LONDON, 1956, PP. 295-302.)                         SE13 380
C                                                                       SE13 390
C     ..................................................................SE13 400
C                                                                       SE13 410
      SUBROUTINE SE13(Y,Z,NDIM,IER)                                     SE13 420
C                                                                       SE13 430
C                                                                       SE13 440
      DIMENSION Y(1),Z(1)                                               SE13 450
C                                                                       SE13 460
C        TEST OF DIMENSION                                              SE13 470
      IF(NDIM-3)3,1,1                                                   SE13 480
C                                                                       SE13 490
C        PREPARE LOOP                                                   SE13 500
    1 B=.1666667*(5.*Y(1)+Y(2)-Y(3))                                    SE13 510
      C=.1666667*(5.*Y(NDIM)+Y(NDIM-1)+Y(NDIM-1)-Y(NDIM-2))             SE13 520
C                                                                       SE13 530
C        START LOOP                                                     SE13 540
      DO 2 I=3,NDIM                                                     SE13 550
      A=B                                                               SE13 560
      B=.3333333*(Y(I-2)+Y(I-1)+Y(I))                                   SE13 570
    2 Z(I-2)=A                                                          SE13 580
C        END OF LOOP                                                    SE13 590
C                                                                       SE13 600
C        UPDATE LAST TWO COMPONENTS                                     SE13 610
      Z(NDIM-1)=B                                                       SE13 620
      Z(NDIM)=C                                                         SE13 630
      IER=0                                                             SE13 640
      RETURN                                                            SE13 650
C                                                                       SE13 660
C        ERROR EXIT IN CASE NDIM IS LESS THAN 3                         SE13 670
    3 IER=-1                                                            SE13 680
      RETURN                                                            SE13 690
      END                                                               SE13 700
```

```
C                                                                       DE13  10
C     ..................................................................DE13  20
C                                                                       DE13  30
C        SUBROUTINE DSE13                                               DE13  40
C                                                                       DE13  50
C        PURPOSE                                                        DE13  60
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN A     DE13  70
C           VECTOR OF FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO       DE13  80
C           EQUIDISTANTLY SPACED ARGUMENT VALUES.                       DE13  90
C                                                                       DE13 100
C        USAGE                                                          DE13 110
C           CALL DSE13(Y,Z,NDIM,IER)                                    DE13 120
C                                                                       DE13 130
C        DESCRIPTION OF PARAMETERS                                      DE13 140
C           Y     -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES   DE13 150
C                    (DIMENSION NDIM)                                   DE13 160
C           Z     -  RESULTING VECTOR OF DOUBLE PRECISION SMOOTHED      DE13 170
C                    FUNCTION VALUES (DIMENSION NDIM)                   DE13 180
C           NDIM  -  DIMENSION OF VECTORS Y AND Z                       DE13 190
C           IER   -  RESULTING ERROR PARAMETER                          DE13 200
C                    IER = -1  - NDIM IS LESS THAN 3                    DE13 210
C                    IER =  0  - NO ERROR                               DE13 220
C                                                                       DE13 230
C        REMARKS                                                        DE13 240
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.               DE13 250
C           (2)  Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y     DE13 260
C                IS DISTINCT FROM Z, THEN IT IS NOT DESTROYED.          DE13 270
C                                                                       DE13 280
C        SUBROUTINES AND SUBPROGRAMS REQUIRED                           DE13 290
C           NONE                                                        DE13 300
C                                                                       DE13 310
C        METHOD                                                         DE13 320
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    DE13 330
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), EACH SMOOTHED     DE13 340
C           VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE LEAST-     DE13 350
C           SQUARES POLYNOMIAL OF DEGREE 1 RELEVANT TO THE 3 SUCCESSIVE DE13 360
C           POINTS (X(I+K),Y(I+K)) K = -1,0,1.  (SEE HILDEBRAND, F.B.,  DE13 370
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ DE13 380
C           TORONTO/LONDON, 1956, PP. 295-302.)                         DE13 390
C                                                                       DE13 400
C     ..................................................................DE13 410
C                                                                       DE13 420
      SUBROUTINE DSE13(Y,Z,NDIM,IER)                                    DE13 430
C                                                                       DE13 440
      DIMENSION Y(1),Z(1)                                               DE13 450
      DOUBLE PRECISION Y,Z,A,B,C                                        DE13 460
C                                                                       DE13 470
C        TEST OF DIMENSION                                              DE13 480
      IF(NDIM-3)3,1,1                                                   DE13 490
C                                                                       DE13 500
C        PREPARE LOOP                                                   DE13 510
    1 B=.1666666666666667D0*(5.D0*Y(1)+Y(2)+Y(3))                       DE13 520
      C=.1666666666666667*(5.D0*Y(NDIM)+Y(NDIM-1)+Y(NDIM-1)-Y(NDIM-2))  DE13 530
C                                                                       DE13 540
C        START LOOP                                                     DE13 550
      DO 2 I=3,NDIM                                                     DE13 560
      A=B                                                               DE13 570
      B=.3333333333333333D0*(Y(I-2)+Y(I-1)+Y(I))                        DE13 580
    2 Z(I-2)=A                                                          DE13 590
C        END OF LOOP                                                    DE13 600
C                                                                       DE13 610
C        UPDATE LAST TWO COMPONENTS                                     DE13 620
      Z(NDIM-1)=B                                                       DE13 630
      Z(NDIM)=C                                                         DE13 640
      IER=0                                                             DE13 650
      RETURN                                                            DE13 660
C                                                                       DE13 670
C        ERROR EXIT IN CASE NDIM IS LESS THAN 3                         DE13 680
    3 IER=-1                                                            DE13 690
      RETURN                                                            DE13 700
      END                                                               DE13 710
```

Subroutines SE15 and DSE15

These subroutines compute a vector $Z = (z_1, \ldots, z_n)$ of smoothed function values, given a vector $Y = (y_1, \ldots, y_n)$ of function values whose entries y_i correspond to n equidistantly spaced argument values x_i with $x_i - x_{i-1} = h$ for $i = 2, \ldots, n$. Except at the points x_1, x_2, x_{n-1}, and x_n, each smoothed value z_i is obtained by evaluating at x_i the least-squares polynomial of degree 1 relevant to the five successive points (x_{i+k}, y_{i+k}), $k = -2, -1, \ldots, 2$.

1. Mathematical background

For $i = 5, \ldots, n$ we find m_i and b_i such that

$$w_i(x) = m_i x + b_i \tag{1}$$

gives the least-squares fit to the points (x_{i-k}, y_{i-k}), $k=0, \ldots, 4$. The problem, then, is to minimize

$$F(m_i, b_i) = \sum_{k=0}^{4} \left[w_i(x_{i-k}) - y_{i-k} \right]^2$$

This minimum will occur when

$$\frac{\partial F}{\partial b_i} = 0 \text{ and } \frac{\partial F}{\partial m_i} = 0 \tag{2}$$

Now

$$\frac{\partial F}{\partial b_i} = 2 \sum_{k=0}^{4} \left[w_i(x_{i-k}) - y_{i-k} \right]$$

and

$$\frac{\partial F}{\partial m_i} = 2 \sum_{k=0}^{4} x_{i-k} \left[w_i(x_{i-k}) - y_{i-k} \right] \tag{3}$$

Solving equations (2) and (3) yields:

$$m_i = \frac{\sum_{k=0}^{4} x_{i-k} y_{i-k} - \frac{1}{5}\left(\sum_{k=0}^{4} x_{i-k}\right)\left(\sum_{k=0}^{4} y_{i-k}\right)}{\sum_{k=0}^{4} x_{i-k}^2 - \frac{1}{5}\left(\sum_{k=0}^{4} x_{i-k}\right)^2} \tag{4}$$

and

$$b_i = \frac{1}{5} \sum_{k=0}^{4} \left[y_{i-k} - m_i x_{i-k} \right] \tag{5}$$

Using the fact that $x_j - x_{j-1} = h$, a constant, for $j = 2, \ldots, n$, (4) and (5) may be rewritten as

$$m_i = \frac{1}{10h} (2y_i + y_{i-1} - y_{i-3} - 2y_{i-4}) \tag{6}$$

and

$$b_i = \frac{1}{5} \sum_{k=0}^{4} y_{i-k} - m_i x_{i-2} \tag{7}$$

Using (7) in (1) yields

$$w_i(x) = m_i(x - x_{i-2}) + \frac{1}{5}(y_{i-4} + \ldots + y_i)$$

The desired smoothed function values z_i are given by:

$$z_i = \begin{cases} w_5(x_1) = \frac{1}{5}(3y_1 + 2y_2 + y_3 - y_5) & i=1 \\ w_5(x_2) = \frac{1}{10}(4y_1 + 3y_2 + 2y_3 + y_4) & i=2 \\ w_{i+2}(x_i) = \frac{1}{5}(y_{i-2} + y_{i-1} + y_i + y_{i+1} + y_{i+2}) & i=3, \ldots, n-2 \\ w_n(x_{n-1}) = \frac{1}{10}(y_{n-3} + 2y_{n-2} + 3y_{n-1} + 4y_n) & i=n-1 \\ w_n(x_n) = \frac{1}{5}(-y_{n-4} + y_{n-2} + 2y_{n-1} + 3y_n) & i=n \end{cases} \tag{8}$$

2. Programming considerations

The subroutines compute the z_i in serial order according to (8). If $n < 5$, there is no computation, and the error parameter IER is set to -1. Otherwise IER is set to zero at the end of the computation. The output vector Z can have the same storage allocation as Y.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp 295-302.

```
C
C     ..................................................................
C
C        SUBROUTINE SE15
C
C        PURPOSE
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN A
C           VECTOR OF FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO
C           EQUIDISTANTLY SPACED ARGUMENT VALUES.
C
C        USAGE
C           CALL SE15(Y,Z,NDIM,IER)
C
C        DESCRIPTION OF PARAMETERS
C           Y     -  GIVEN VECTOR OF FUNCTION VALUES (DIMENSION NDIM)
C           Z     -  RESULTING VECTOR OF SMOOTHED FUNCTION VALUES
C                    (DIMENSION NDIM)
C           NDIM  -  DIMENSION OF VECTORS Y AND Z
C           IER   -  RESULTING ERROR PARAMETER
C                    IER = -1  - NDIM IS LESS THAN 5
C                    IER =  0  - NO ERROR
C
C        REMARKS
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.
C           (2)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y IS
C                 DISTINCT FROM Z, THEN IT IS NOT DESTROYED.
C
C        SUBROUTINE AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN
C           EXCEPT AT THE POINTS X(1),X(2),X(NDIM-1) AND X(NDIM), EACH
C           SMOOTHED VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE
C           LEAST-SQUARES POLYNOMIAL OF DEGREE 1 RELEVANT TO THE 5
C           SUCCESSIVE POINTS (X(I+K),Y(I+K)) K = -2,-1,...,2.  (SEE
C           HILDEBRAND, F.B., INTRODUCTION TO NUMERICAL ANALYSIS,
C           MC GRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP. 295-302.)
C
C     ..................................................................
C
      SUBROUTINE SE15(Y,Z,NDIM,IER)
C
C
      DIMENSION Y(1),Z(1)
C
C        TEST OF DIMENSION
      IF(NDIM-5)3,1,1
C
C        PREPARE LOOP
    1 A=Y(1)+Y(1)
      C=Y(2)+Y(2)
      B=.2*(A+Y(1)+C+Y(3)-Y(5))
      C=.1*(A+A+C+Y(2)+Y(3)+Y(3)+Y(4))
C
C        START LOOP
      DO 2 I=5,NDIM
      A=B
      B=C
      C=.2*(Y(I-4)+Y(I-3)+Y(I-2)+Y(I-1)+Y(I))
    2 Z(I-4)=A
C        END OF LOOP
C
C        UPDATE LAST FOUR COMPONENTS
      A=Y(NDIM)+Y(NDIM)
      B=.1*(A+A+Y(NDIM-1)+Y(NDIM-1)+Y(NDIM-1)+Y(NDIM-2)+Y(NDIM-2)
     1      +Y(NDIM-3))
      Z(NDIM-3)=B
      Z(NDIM-2)=C
      Z(NDIM-1)=A
      Z(NDIM)=A+A-C
      IER=0
      RETURN
C
C        ERROR EXIT IN CASE NDIM IS LESS THAN 5
    3 IER=-1
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DSE15
C
C        PURPOSE
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN A
C           VECTOR OF FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO
C           EQUIDISTANTLY SPACED ARGUMENT VALUES.
C
C        USAGE
C           CALL DSE15(Y,Z,NDIM,IER)
C
C        DESCRIPTION OF PARAMETERS
C           Y     -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES
C                    (DIMENSION NDIM)
C           Z     -  RESULTING VECTOR OF DOUBLE PRECISION SMOOTHED
C                    FUNCTION VALUES (DIMENSION NDIM)
C           NDIM  -  DIMENSION OF VECTORS Y AND Z
C           IER   -  RESULTING ERROR PARAMETER
C                    IER = -1  - NDIM IS LESS THAN 5
C                    IER =  0  - NO ERROR
C
C        REMARKS
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.
C           (2)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y IS
C                 DISTINCT FROM Z, THEN IT IS NOT DESTROYED.
C
C        SUBROUTINE AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN
C           EXCEPT AT THE POINTS X(1),X(2),X(NDIM-1) AND X(NDIM), EACH
C           SMOOTHED VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE
C           LEAST-SQUARES POLYNOMIAL OF DEGREE 1 RELEVANT TO THE 5
C           SUCCESSIVE POINTS (X(I+K),Y(I+K)) K = -2,-1,...,2.  (SEE
C           HILDEBRAND, F.B., INTRODUCTION TO NUMERICAL ANALYSIS,
C           MC GRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP. 295-302.)
C
C     ..................................................................
C
      SUBROUTINE DSE15(Y,Z,NDIM,IER)
C
C
      DIMENSION Y(1),Z(1)
      DOUBLE PRECISION Y,Z,A,B,C
C
C        TEST OF DIMENSION
      IF(NDIM-5)3,1,1
C
C        PREPARE LOOP
    1 A=Y(1)+Y(1)
      C=Y(2)+Y(2)
      B=.2D0*(A+Y(1)+C+Y(3)-Y(5))
      C=.1D0*(A+A+C+Y(2)+Y(3)+Y(3)+Y(4))
C
C        START LOOP
      DO 2 I=5,NDIM
      A=B
      B=C
      C=.2D0*(Y(I-4)+Y(I-3)+Y(I-2)+Y(I-1)+Y(I))
    2 Z(I-4)=A
C        END OF LOOP
C
C        UPDATE LAST FOUR COMPONENTS
      A=Y(NDIM)+Y(NDIM)
      B=.1D0*(A+A+Y(NDIM-1)+Y(NDIM-1)+Y(NDIM-1)+Y(NDIM-2)+Y(NDIM-2)
     1      +Y(NDIM-3))
      Z(NDIM-3)=B
      Z(NDIM-2)=C
      Z(NDIM-1)=A
      Z(NDIM)=A+A-C
      IER=0
      RETURN
C
C        ERROR EXIT IN CASE NDIM IS LESS THAN 5
    3 IER=-1
      RETURN
      END
```

Subroutine SE35 and DSE35

These subroutines compute a vector $Z = (z_1, \ldots, z_n)$ of smoothed function values, given a vector $Y = (y_1, \ldots, y_n)$ of function values whose entries y_i correspond to n equidistantly spaced argument values x_i, with $x_i - x_{i-1} = h$ for $i = 2, \ldots, n$. Except at the points x_1, x_2, x_{n-1}, and x_n, each smoothed value z_i is obtained by evaluating at x_i the least-squares polynomial of degree 3 relevant to the five successive points (x_{i+k}, y_{i+k}), $k = -2, -1, \ldots, 2$.

1. Mathematical background

For $i = 5, \ldots, n$ we must find a_i, b_i, c_i, and d_i such that

$$w_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \tag{1}$$

gives the least-squares fit to the points (x_{i-k}, y_{i-k}), $k = 0, \ldots, 4$.

The problem, thus, is to minimize

$$F(a_i, b_i, c_i, d_i) = \sum_{k=0}^{4} \left[w_i(x_{i-k}) - y_{i-k} \right]^2 \tag{2}$$

The minimum will occur when

$$\frac{\partial F}{\partial a_i} = \frac{\partial F}{\partial b_i} = \frac{\partial F}{\partial c_i} = \frac{\partial F}{\partial d_i} = 0$$

Now:

$$\left.\begin{aligned}
\frac{\partial F}{\partial a_i} &= 2 \sum_{k=0}^{4} x_{i-k}^3 \left[w_i(x_{i-k}) - y_{i-k} \right] \\
\frac{\partial F}{\partial b_i} &= 2 \sum_{k=0}^{4} x_{i-k}^2 \left[w_i(x_{i-k}) - y_{i-k} \right] \\
\frac{\partial F}{\partial c_i} &= 2 \sum_{k=0}^{4} x_{i-k} \left[w_i(x_{i-k}) - y_{i-k} \right] \\
\frac{\partial F}{\partial d_i} &= 2 \sum_{k=0}^{4} \left[w_i(x_{i-k}) - y_{i-k} \right]
\end{aligned}\right\} \tag{3}$$

Solving (2) and (3) for a_i, b_i, c_i, and d_i, with $x_i - x_{i-1} = h$, we get:

$$a_i = A_i$$

$$b_i = -3 A_i x_{i-2} + B_i$$

$$c_i = 3 A_i x_{i-2}^2 - 2 B_i x_{i-2} + C_i$$

$$d_i = -A_i x_{i-2}^3 + B_i x_{i-2}^2 - C_i x_{i-2} + D_i + \overline{y}_i$$

where:

$$\overline{y_i} = \frac{1}{5} \sum_{k=0}^{4} y_{i-k}$$

$$A_i = -\frac{1}{12 h^3} (y_{i-4} - 2y_{i-3} + 2y_{i-1} - y_i)$$

$$B_i = \frac{1}{14 h^2} (4y_{i-4} + y_{i-3} + y_{i-1} + 4y_i - 10\,\overline{y}_i)$$

$$C_i = \frac{1}{12h} (y_{i-4} - 8y_{i-3} + 8y_{i-1} - y_i)$$

$$D_i = -\frac{1}{7} (4y_{i-4} + y_{i-3} + y_{i-1} + 4y_i - 10\,\overline{y}_i)$$

Finally, the desired smoothed values z_i are given by:

$$z_i = \left\{\begin{aligned}
w_5(x_1) &= y_1 - \frac{1}{70} \delta^4 y_3 \text{ if } i=1 \\
w_5(x_2) &= y_2 + \frac{2}{35} \delta^4 y_3 \text{ if } i=2 \\
w_{i+2}(x_i) &= y_i - \frac{3}{35} \delta^4 y_i \text{ if } i=3, \ldots, n-2 \\
w_n(x_{n-1}) &= y_{n-1} + \frac{2}{35} \delta^4 y_{n-2} \text{ if } i=n-1 \\
w_n(x_n) &= y_n - \frac{1}{70} \delta^4 y_{n-2} \text{ if } i=n
\end{aligned}\right\} \tag{4}$$

where

$$\delta^4 y_i = y_{i-2} - 4 y_{i-1} + 6y_i - 4y_{i+1} + y_{i+2}$$

for $i=3, \ldots, n-2$

2. Programming considerations

These subroutines compute the z_i in serial order according to (4). If $n < 5$, there is no computation, and the error parameter IER is set to -1. Otherwise IER is set to zero at the end of the computation. The output vector Z can have the same storage allocation as Y.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 295-302.

```
C                                                                       SE35  10
C     ..................................................................SE35  20
C                                                                       SE35  30
C        SUBROUTINE SE35                                                SE35  40
C                                                                       SE35  50
C        PURPOSE                                                        SE35  60
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN A     SE35  70
C           VECTOR OF FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO       SE35  80
C           EQUIDISTANTLY SPACED ARGUMENT VALUES.                       SE35  90
C                                                                       SE35 100
C        USAGE                                                          SE35 110
C           CALL SE35(Y,Z,NDIM,IER)                                     SE35 120
C                                                                       SE35 130
C        DESCRIPTION OF PARAMETERS                                      SE35 140
C           Y     -  GIVEN VECTOR OF FUNCTION VALUES (DIMENSION NDIM)   SE35 150
C           Z     -  RESULTING VECTOR OF SMOOTHED FUNCTION VALUES       SE35 160
C                    (DIMENSION NDIM)                                   SE35 170
C           NDIM  -  DIMENSION OF VECTORS Y AND Z                       SE35 180
C           IER   -  RESULTING ERROR PARAMETER                          SE35 190
C                    IER = -1  - NDIM IS LESS THAN 5                    SE35 200
C                    IER =  0  - NO ERROR                               SE35 210
C                                                                       SE35 220
C        REMARKS                                                        SE35 230
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.               SE35 240
C           (2)  Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y IS  SE35 250
C                DISTINCT FROM Z, THEN IT IS NOT DESTROYED.             SE35 260
C                                                                       SE35 270
C        SUBROUTINE AND FUNCTION SUBPROGRAMS REQUIRED                   SE35 280
C           NONE                                                        SE35 290
C                                                                       SE35 300
C        METHOD                                                         SE35 310
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    SE35 320
C           EXCEPT AT THE POINTS X(1),X(2),X(NDIM-1) AND X(NDIM), EACH  SE35 330
C           SMOOTHED VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE   SE35 340
C           LEAST-SQUARES POLYNOMIAL OF DEGREE 3 RELEVANT TO THE 5      SE35 350
C           SUCCESSIVE POINTS (X(I+K),Y(I+K)) K = -2,-1,...,2.  (SEE    SE35 360
C           HILDEBRAND, F.B., INTRODUCTION TO NUMERICAL ANALYSIS,       SE35 370
C           MC GRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP. 295-302.)  SE35 380
C                                                                       SE35 390
C     ..................................................................SE35 400
C                                                                       SE35 410
      SUBROUTINE SE35(Y,Z,NDIM,IER)                                     SE35 420
C                                                                       SE35 430
C                                                                       SE35 440
      DIMENSION Y(1),Z(1)                                               SE35 450
C                                                                       SE35 460
C        TEST OF DIMENSION                                              SE35 470
      IF(NDIM-5)4,1,1                                                   SE35 480
C                                                                       SE35 490
C        PREPARE LOOP                                                   SE35 500
    1 B=Y(1)                                                            SE35 510
      C=Y(2)                                                            SE35 520
C                                                                       SE35 530
C        START LOOP                                                     SE35 540
      DO 3 I=5,NDIM                                                     SE35 550
      A=B                                                               SE35 560
      B=C                                                               SE35 570
      C=Y(I-2)                                                          SE35 580
C                                                                       SE35 590
C        GENERATE FOURTH CENTRAL DIFFERENCE                             SE35 600
      D=C-B-Y(I-1)                                                      SE35 610
      C=D+D+C                                                           SE35 620
      D=D+D+A+Y(I)                                                      SE35 630
C                                                                       SE35 640
C        CHECK FIRST TWO COMPONENTS                                     SE35 650
      IF(I-5)2,2,3                                                      SE35 660
    2 Z(1)=A-.01428571*D                                                SE35 670
      Z(2)=B+.05714286*D                                                SE35 680
    3 Z(I-2)=C-.09571429*D                                              SE35 690
C        END OF LOOP                                                    SE35 700
C                                                                       SE35 710
C        UPDATE LAST TWO COMPONENTS                                     SE35 720
      Z(NDIM-1)=Y(NDIM-1)+.05714286*D                                   SE35 730
      Z(NDIM)=Y(NDIM)-.01428571*D                                       SE35 740
      IER=0                                                             SE35 750
      RETURN                                                            SE35 760
C                                                                       SE35 770
C        ERROR EXIT IN CASE NDIM IS LESS THAN 5                         SE35 780
    4 IER=-1                                                            SE35 790
      RETURN                                                            SE35 800
      END                                                               SE35 810
```

```
C                                                                       DSE3  10
C     ..................................................................DSE3  20
C                                                                       DSE3  30
C        SUBROUTINE DSE35                                               DSE3  40
C                                                                       DSE3  50
C        PURPOSE                                                        DSE3  60
C           TO COMPUTE A VECTOR OF SMOOTHED FUNCTION VALUES GIVEN A     DSE3  70
C           VECTOR OF FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO       DSE3  80
C           EQUIDISTANTLY SPACED ARGUMENT VALUES.                       DSE3  90
C                                                                       DSE3 100
C        USAGE                                                          DSE3 110
C           CALL DSE35(Y,Z,NDIM,IER)                                    DSE3 120
C                                                                       DSE3 130
C        DESCRIPTION OF PARAMETERS                                      DSE3 140
C           Y     -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES   DSE3 150
C                    (DIMENSION NDIM)                                   DSE3 160
C           Z     -  RESULTING VECTOR OF DOUBLE PRECISION SMOOTHED      DSE3 170
C                    FUNCTION VALUES (DIMENSION NDIM)                   DSE3 180
C           NDIM  -  DIMENSION OF VECTORS Y AND Z                       DSE3 190
C           IER   -  RESULTING ERROR PARAMETER                          DSE3 200
C                    IER = -1  - NDIM IS LESS THAN 5                    DSE3 210
C                    IER =  0  - NO ERROR                               DSE3 220
C                                                                       DSE3 230
C        REMARKS                                                        DSE3 240
C           (1)  IF IER=-1 THERE HAS BEEN NO COMPUTATION.               DSE3 250
C           (2)  Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y IS  DSE3 260
C                DISTINCT FROM Z, THEN IT IS NOT DESTROYED.             DSE3 270
C                                                                       DSE3 280
C        SUBROUTINE AND FUNCTION SUBPROGRAMS REQUIRED                   DSE3 290
C           NONE                                                        DSE3 300
C                                                                       DSE3 310
C        METHOD                                                         DSE3 320
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    DSE3 330
C           EXCEPT AT THE POINTS X(1),X(2),X(NDIM-1) AND X(NDIM), EACH  DSE3 340
C           SMOOTHED VALUE Z(I) IS OBTAINED BY EVALUATING AT X(I) THE   DSE3 350
C           LEAST-SQUARES POLYNOMIAL OF DEGREE 3 RELEVANT TO THE 5      DSE3 360
C           SUCCESSIVE POINTS (X(I+K),Y(I+K)) K = -2,-1,...,2.  (SEE    DSE3 370
C           HILDEBRAND, F.B., INTRODUCTION TO NUMERICAL ANALYSIS,       DSE3 380
C           MC GRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP. 295-302.)  DSE3 390
C                                                                       DSE3 400
C     ..................................................................DSE3 410
C                                                                       DSE3 420
      SUBROUTINE DSE35(Y,Z,NDIM,IER)                                    DSE3 430
C                                                                       DSE3 440
C                                                                       DSE3 450
      DIMENSION Y(1),Z(1)                                               DSE3 460
      DOUBLE PRECISION Y,Z,A,B,C,D                                      DSE3 470
C                                                                       DSE3 480
C        TEST OF DIMENSION                                              DSE3 490
      IF(NDIM-5)4,1,1                                                   DSE3 500
C                                                                       DSE3 510
C        PREPARE LOOP                                                   DSE3 520
    1 B=Y(1)                                                            DSE3 530
      C=Y(2)                                                            DSE3 540
C                                                                       DSE3 550
C        START LOOP                                                     DSE3 560
      DO 3 I=5,NDIM                                                     DSE3 570
      A=B                                                               DSE3 580
      B=C                                                               DSE3 590
      C=Y(I-2)                                                          DSE3 600
C                                                                       DSE3 610
C        GENERATE FOURTH CENTRAL DIFFERENCE                             DSE3 620
      D=C-B-Y(I-1)                                                      DSE3 630
      C=D+D+C                                                           DSE3 640
      D=D+D+A+Y(I)                                                      DSE3 650
C                                                                       DSE3 660
C        CHECK FIRST TWO COMPONENTS                                     DSE3 670
      IF(I-5)2,2,3                                                      DSE3 680
    2 Z(1)=A-.01428571428571428D0*D                                     DSE3 690
      Z(2)=B+.05714285714285714 3D0*D                                   DSE3 700
    3 Z(I-2)=C-.09571428571428571D0*D                                   DSE3 710
C        END OF LOOP                                                    DSE3 720
C                                                                       DSE3 730
C        UPDATE LAST TWO COMPONENTS                                     DSE3 740
      Z(NDIM-1)=Y(NDIM-1)+.05714285714285714 3D0*D                      DSE3 750
      Z(NDIM)=Y(NDIM)-.01428571428571428D0*D                            DSE3 760
      IER=0                                                             DSE3 770
      RETURN                                                            DSE3 780
C                                                                       DSE3 790
C        ERROR EXIT IN CASE NDIM IS LESS THAN 5                         DSE3 800
    4 IER=-1                                                            DSE3 810
      RETURN                                                            DSE3 820
      END                                                               DSE3 830
```

Subroutine APFS and DAPFS

These subroutines compute the solution of the normal equations set up by subroutines APLL and DAPLL. This is the second step in the procedure for determining the least-squares fit of a given discrete function.

1. Mathematical background

Let $f(x)$, $g_i(x)$, $i=1, 2, \ldots, m$, and $w(x) > 0$ be functions defined for $x=x_1, x_2, \ldots, x_n$.

The problem is to determine the coefficients c_i of the linear combination

$$p(x) = \sum_{i=1}^{m} c_i g_i(x)$$

such that

$$e_m = \sum_{k=1}^{n} w(x_k)\,(f(x_k) - p(x_k))^2 = \text{minimum} \qquad (1)$$

The necessary conditions

$$\frac{\partial e_m}{\partial c_i} = 0, \quad i=1, 2, \ldots, m \qquad (2)$$

form a system of m linear equations in m unknowns c_i.

To simplify the notation we introduce the following matrices:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}, \quad F = \begin{bmatrix} f(x_1) \\ \\ \cdot \\ \cdot \\ \cdot \\ f(x_n) \end{bmatrix}, \quad W = \begin{bmatrix} w(x_1) & & & & \\ & w(x_2) & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & w(x_n) \end{bmatrix},$$

$$c = \begin{bmatrix} c_1 \\ \cdot \\ \cdot \\ \cdot \\ c_m \end{bmatrix}, \quad G = \begin{bmatrix} g_1(x_1) & \cdot\ \cdot & g_1(x_n) \\ \cdot & & \cdot \\ \cdot & & \cdot \\ g_m(x_1) & \cdot\ \cdot & g_m(x_n) \end{bmatrix}$$

Then we have

$$e_m = (F^T - C^T G)\, W\, (F - G^T C)$$

or, with $e_o = F^T W F$,

$$e_m = e_o - 2C^T G W F + C^T G W G^T C \qquad (1')$$

Using (1'), equations (2) may be written as

$$G W G^T C = G W F \qquad (2')$$

Combining (1') and (2') gives

$$e_o - e_m = C^T G W G^T C \qquad (3)$$

The normal equations (2') for the unknown vector C may be solved using Cholesky's method, since the coefficient matrix $A = GWG^T$ is obviously symmetric and is positive definite if all the fundamental functions $g_i(x)$ are linearly independent for the arguments x_i; that is, if the rows of G are linearly independent. Let $R = GWF$. Using Cholesky's method, A and R are replaced without additional storage requirements by T and $(T^T)^{-1}R$, where $A = T^T T$, and T is upper triangular.

Using these facts, an easy calculation shows

$$e_o - e_m = \left\| (T^T)^{-1} R \right\|^2$$

Introducing additional fundamental functions in the linear combination $p(x)$ will not affect the first m rows and columns of A or the first m elements of R. Therefore, Cholesky's method gives a decomposition of $e_o - e_m$ into the separate components corresponding to individual degrees of freedom.

2. Programming considerations

All least-squares fits of dimension 1, 2, ..., m may be computed from the reduced normal equations $TC = (T^T)^{-1} R$. If the solutions are generated in the storage locations of T, there is no additional storage requirement.

Using the decomposition of $e_o - e_m$, the factorization may be terminated with dimension k if $e_k < \eta e_o$, giving the least-squares fit of dimension k which satisfies the user-specified precision (relative tolerance η). Due to rounding errors this will work only if η is approximately between 1 and 10^{-6} in single precision, and between 1 and 10^{-15} in double precision. Nevertheless, the square sum of residuals corresponding to a least-squares fit calculated in single or double precision may be as small as $e_o\, 10^{-12}$ or $e_o\, 10^{-30}$, respectively.

Due to rounding errors the square root method may break down if very small or negative pivot elements indicate a loss of significance. Therefore, all pivot elements are tested against the absolute value of EPS multiplied by the first diagonal element of A. If the kth pivot element is not greater than this internal test value, the normal equations are treated as if they had dimension k-1 only.

Let N1, N2 denote the following numbers:

N1 = the largest number less than or equal to m for which there is no indication of loss of significance

N2 = the smallest number less than or equal to m for which

$$e_{N2} \leq \left| \eta \, e_o \right|$$

Depending on the input parameter IOP, which specifies the operations to be performed, the dimension IRES (that is, the dimension of the subsystem of the normal equations used for calculation) is set to N1 if IOP is positive or zero, and to the smaller of the two numbers N1, N2 if IOP is negative.

The operations performed by APFS are:

IOP = 0 Compute the triangular factorization $A = T^T T$, $TC = (T^T)^{-1}R$ and e_{IRES} only.

IOP = ±1 In addition to the above, the least-squares fit of dimension IRES is computed and stored in column IRES of the triangular array WORK.

IOP = ±2 In addition to IOP = 0 calculations, all least-square fits of dimension 1 up to IRES are computed and stored in columns 1 up to IRES of the array WORK.

a. Storage allocation

Initially the symmetric coefficient matrix A is stored columnwise in compressed form in the triangular array WORK. It is followed immediately by the right-hand side vector R and a single location containing e_o initially and e_{IRES} on return. This storage scheme is generated by subroutine APLL.

b. Errors

The error parameter IER is set to -1 if a nonpositive dimension m has been specified.

If IOP is positive or zero, IER is set to 1 in case of an indication of a loss of significance.

If IOP is negative, IER is set to 1 if the specified accuracy could not be reached, even with a fit of dimension IRES.

For reference see A. T. Berztiss, "Least-squares fitting of polynomials to irregularly spaced data", SIAM Review (1964), vol. 6, no. 3, pp 203-227.

```
C                                                                       APFS  10
C     ..................................................................APFS  20
C                                                                       APFS  30
C        SUBROUTINE APFS                                                APFS  40
C                                                                       APFS  50
C        PURPOSE                                                        APFS  60
C           PERFORM SYMMETRIC FACTORIZATION OF THE MATRIX OF THE NORMAL APFS  70
C           EQUATIONS FOLLOWED BY CALCULATION OF THE LEAST SQUARES FIT  APFS  80
C           OPTIONALLY                                                  APFS  90
C                                                                       APFS 100
C        USAGE                                                          APFS 110
C           CALL APFS(WORK,IP,IRES,IOP,EPS,ETA,IER)                     APFS 120
C                                                                       APFS 130
C        DESCRIPTION OF PARAMETERS                                      APFS 140
C           WORK   - GIVEN SYMMETRIC COEFFICIENT MATRIX, STORED         APFS 150
C                    COMPRESSED, I.E UPPER TRIANGULAR PART COLUMNWISE.  APFS 160
C                    THE GIVEN RIGHT HAND SIDE OCCUPIES THE NEXT IP     APFS 170
C                    LOCATIONS IN WORK. THE VERY LAST COMPONENT OF WORK APFS 180
C                    CONTAINS THE SQUARE SUM OF FUNCTION VALUES EO      APFS 190
C                    THIS SCHEME OF STORAGE ALLOCATION IS PRODUCED E.G. APFS 200
C                    BY SUBROUTINE APLL.                                APFS 210
C                    THE GIVEN MATRIX IS FACTORED IN THE FORM           APFS 220
C                    TRANSPOSE(T)*T AND THE GIVEN RIGHT HAND SIDE IS    APFS 230
C                    DIVIDED BY TRANSPOSE(T).                           APFS 240
C                    THE UPPER TRIANGULAR FACTOR T IS RETURNED IN WORK IFAPFS 250
C                    IOP EQUALS ZERO.                                   APFS 260
C                    IN CASE OF NONZERO IOP THE CALCULATED SOLUTIONS ARE APFS 270
C                    STORED IN THE COLUMNS OF TRIANGULAR APPAY WORK OF  APFS 280
C                    CORRESPONDING DIMENSION AND EO  IS REPLACED BY THE APFS 290
C                    SQUARE SUM OF THE ERRORS FOR FIT OF DIMENSION IRES. APFS 300
C                    THE TOTAL DIMENSION OF WORK IS (IP+1)*(IP+2)/2     APFS 310
C           IP     - NUMBER OF FUNDAMENTAL FUNCTIONS USED FOR LEAST    APFS 320
C                    SQUARES FIT                                        APFS 330
C           IRES   - DIMENSION OF CALCULATED LEAST SQUARES FIT.         APFS 340
C                    LET N1, N2, DENOTE THE FOLLOWING NUMBERS           APFS 350
C                    N1 = MAXIMAL DIMENSION FOR WHICH NO LOSS OF        APFS 360
C                         SIGNIFICANCE WAS INDICATED DURING FACTORIZATIONAPFS 370
C                    N2 = SMALLEST DIMENSION FOR WHICH THE SQUARE SUM OF APFS 380
C                         THE ERRORS DOES NOT EXCEED TEST=ABS(ETA*FSQ)  APFS 390
C                    THEN IRES=MINO(IP,N1) IF IOP IS NONNEGATIVE        APFS 400
C                    AND  IRES=MINU(IP,N1,N2) IF IOP IS NEGATIVE        APFS 410
C           IOP    - INPUT PARAMETER FOR SELECTION OF OPERATION        APFS 420
C                    IOP = 0 MEANS TRIANGULAR FACTORIZATION, DIVISION OF APFS 430
C                            THE RIGHT HAND SIDE BY TRANSPOSE(T) AND    APFS 440
C                            CALCULATION OF THE SQUARE SUM OF ERRORS IS APFS 450
C                            PERFORMED ONLY                             APFS 460
C                    IOP = +1 OR -1 MEANS THE SOLUTION OF DIMENSION IRES APFS 470
C                            IS CALCULATED ADDITIONALLY                 APFS 480
C                    IOP = +2 OR -2 MEANS ALL SOLUTIONS FOR DIMENSION ONEAPFS 490
C                            UP TO IRES ARE CALCULATED ADDITIONALLY     APFS 500
C           EPS    - RELATIVE TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.APFS 510
C                    A SENSIBLE VALUE IS BETWEEN 1.E-3 AND 1.E-6        APFS 520
C           ETA    - RELATIVE TOLERANCE FOR TOLERATED SQUARE SUM OF    APFS 530
C                    ERRORS. A REALISTIC VALUE IS BETWEEN 1.EO AND 1.E-6 APFS 540
C           IER    - RESULTANT ERROR PARAMETER                         APFS 550
C                    IER =-1 MEANS NONPOSITIVE IP                       APFS 560
C                    IER = 0 MEANS NO LOSS OF SIGNIFICANCE DETECTED     APFS 570
C                            AND SPECIFIED TOLERANCE OF ERRORS REACHED  APFS 580
C                    IER = 1 MEANS LOSS OF SIGNIFICANCE DETECTED OR     APFS 590
C                            SPECIFIED TOLERANCE OF ERRORS NOT REACHED  APFS 600
C                                                                       APFS 610
C        REMARKS                                                        APFS 620
C           THE ABSOLUTE TOLERANCE USED INTERNALLY FOR TEST ON LOSS OF  APFS 630
C           SIGNIFICANCE IS TOL=ABS(EPS*WORK(1)).                       APFS 640
C           THE ABSOLUTE TOLERANCE USED INTERNALLY FOR THE SQUARE SUM OFAPFS 650
C           ERRORS IS ABS(ETA*FSQ).                                     APFS 660
C           IOP GREATER THAN 2 HAS THE SAME EFFECT AS IOP = 2.          APFS 670
C           IOP LESS THAN -2 HAS THE SAME EFFECT AS IOP =-2.            APFS 680
C           IRES = 0 MEANS THE ABSOLUTE VALUE OF EPS IS NOT LESS THAN   APFS 690
C           ONE AND/OR WORK(1) IS NOT POSITIVE AND/OR IP IS NOT POSITIVEAPFS 700
C                                                                       APFS 710
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  APFS 720
C           NONE                                                        APFS 730
C                                                                       APFS 740
C        METHOD                                                         APFS 750
C           CALCULATION OF THE LEAST SQUARES FITS IS DONE USING         APFS 760
C           CHOLESKYS SQUARE ROOT METHOD FOR SYMMETRIC FACTORIZATION.   APFS 770
C           THE INCORPORATED TEST ON LOSS OF SIGNIFICANCE MEANS EACH    APFS 780
C           RADICAND MUST BE GREATER THAN THE INTERNAL ABSOLUTE         APFS 790
C           TOLERANCE TOL=ABS(EPS*WORK(1)).                             APFS 800
C           IN CASE OF LOSS OF SIGNIFICANCE IN THE ABOVE SENSE ONLY A   APFS 810
C           SUBSYSTEM OF THE NORMAL EQUATIONS IS SOLVED.                APFS 820
C           IN CASE OF NEGATIVE IOP THE TRIANGULAR FACTORIZATION IS     APFS 830
C           TERMINATED PREMATURELY EITHER IF THE SQUARE SUM OF THE      APFS 840
C           ERRORS DOES NOT EXCEED ETA*FSQ OR IF THERE IS INDICATION    APFS 850
C           FOR LOSS OF SIGNIFICANCE                                    APFS 860
C                                                                       APFS 870
C     ..................................................................APFS 880
C                                                                       APFS 890
      SUBROUTINE APFS(WORK,IP,IRES,IOP,EPS,ETA,IER)                     APFS 900
C                                                                       APFS 910
C                                                                       APFS 920
C        DIMENSIONED DUMMY VARIABLES                                    APFS 930
      DIMENSION WORK(1)                                                 APFS 940
      IRES=0                                                            APFS 950
C                                                                       APFS 960
C        TEST OF SPECIFIED DIMENSION                                    APFS 970
C                                                                       APFS 980
      IF(IP)1,1,2                                                       APFS 990
C                                                                       APFS1000
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSION                      APFS1010
    1 IER=-1                                                            APFS1020
      RETURN                                                            APFS1030
C                                                                       APFS1040
C        INITIALIZE FACTORIZATION PROCESS                               APFS1050
    2 IPIV=0                                                            APFS1060
      IPP1=IP+1                                                         APFS1070
      IER=1                                                             APFS1080
      ITE=IP*IPP1/2                                                     APFS1090
      IEND=ITE+IPP1                                                     APFS1100
      TOL=ABS(EPS*WORK(1))                                              APFS1110
      TEST=ABS(ETA*WORK(IEND))                                          APFS1120
C                                                                       APFS1130
C        START LOOP OVER ALL ROWS OF WORK                               APFS1140
      DO 11 I=1,IP                                                      APFS1150
      IPIV=IPIV+I                                                       APFS1160
      JA=IPIV-IRES                                                      APFS1170
      JE=IPIV-1                                                         APFS1180
C                                                                       APFS1190
C        FORM SCALAR PRODUCT NEEDED TO MODIFY CURRENT ROW ELEMENTS      APFS1200
      JK=IPIV
```

```
      DO 9 K=I,IPP1                                                     APFS1210
      SUM=0.                                                            APFS1220
      IF(IRES)5,5,3                                                     APFS1230
    3 JK=JK-IRES                                                        APFS1240
      DO 4 J=JA,JE                                                      APFS1250
      SUM=SUM+WORK(J)*WORK(JK)                                          APFS1260
    4 JK=JK+1                                                           APFS1270
    5 IF(JK-IPIV)6,6,8                                                  APFS1280
C                                                                       APFS1290
C        TEST FOR LOSS OF SIGNIFICANCE                                  APFS1300
    6 SUM=WORK(IPIV)-SUM                                                APFS1310
      IF(SUM-TOL)12,12,7                                                APFS1320
    7 SUM=SQRT(SUM)                                                     APFS1330
      WORK(IPIV)=SUM                                                    APFS1340
      PIV=1./SUM                                                        APFS1350
      GOTO 9                                                            APFS1360
C                                                                       APFS1370
C        UPDATE OFF-DIAGONAL TERMS                                      APFS1380
    8 SUM=(WORK(JK)-SUM)*PIV                                            APFS1390
      WORK(JK)=SUM                                                      APFS1400
    9 JK=JK+K                                                           APFS1410
C                                                                       APFS1420
C        UPDATE SQUARE SUM OF ERRORS                                    APFS1430
      WORK(IEND)=WORK(IEND)-SUM*SUM                                     APFS1440
C                                                                       APFS1450
C        RECORD ADDRESS OF LAST PIVOT ELEMENT                           APFS1460
      IRES=IRES+1                                                       APFS1470
      IADR=IPIV                                                         APFS1480
C                                                                       APFS1490
C        TEST FOR TOLERABLE ERROR IF SPECIFIED                          APFS1500
      IF(IOP)10,11,11                                                   APFS1510
   10 IF(WORK(IEND)-TEST)13,13,11                                       APFS1520
   11 CONTINUE                                                          APFS1530
      IF(IOP)12,22,12                                                   APFS1540
C                                                                       APFS1550
C        PERFORM BACK SUBSTITUTION IF SPECIFIED                         APFS1560
   12 IF(IOP)14,23,14                                                   APFS1570
   13 IER=0                                                             APFS1580
   14 IPIV=IRES                                                         APFS1590
   15 IF(IPIV)23,23,16                                                  APFS1600
   16 SUM=0.                                                            APFS1610
      JA=ITE+IPIV                                                       APFS1620
      JJ=IADR                                                           APFS1630
      JK=IADR                                                           APFS1640
      K=IPIV                                                            APFS1650
      DO 19 I=1,IPIV                                                    APFS1660
      WORK(JK)=(WORK(JA)-SUM)/WORK(JJ)                                  APFS1670
      IF(K-1)20,20,17                                                   APFS1680
   17 JE=JJ-1                                                           APFS1690
      SUM=0.                                                            APFS1700
      DO 18 J=K,IPIV                                                    APFS1710
      SUM=SUM+WORK(JK)*WORK(JE)                                         APFS1720
      JK=JK+1                                                           APFS1730
   18 JE=JE+J                                                           APFS1740
      JK=JE-IPIV                                                        APFS1750
      JA=JA-1                                                           APFS1760
      JJ=JJ-K                                                           APFS1770
   19 K=K-1                                                             APFS1780
   20 IF(IOP/2)21,23,21                                                 APFS1790
   21 IADR=IADR-IPIV                                                    APFS1800
      IPIV=IPIV-1                                                       APFS1810
      GOTO 15                                                           APFS1820
C                                                                       APFS1830
C        NORMAL RETURN                                                  APFS1840
   22 IER=0                                                             APFS1850
   23 RETURN                                                            APFS1860
      END                                                               APFS1870
```

```
C                                                                       DAPF  10
C     ******************************************************************DAPF  20
C                                                                       DAPF  30
C        SUBROUTINE DAPFS                                               DAPF  40
C                                                                       DAPF  50
C        PURPOSE                                                        DAPF  60
C           PERFORM SYMMETRIC FACTORIZATION OF THE MATRIX OF THE NORMAL DAPF  70
C           EQUATIONS FOLLOWED BY CALCULATION OF THE LEAST SQUARES FIT  DAPF  80
C           OPTIONALLY                                                  DAPF  90
C                                                                       DAPF 100
C        USAGE                                                          DAPF 110
C           CALL DAPFS(WORK,IP,IRES,IOP,EPS,ETA,IER)                    DAPF 120
C                                                                       DAPF 130
C        DESCRIPTION OF PARAMETERS                                      DAPF 140
C           WORK  - GIVEN SYMMETRIC COEFFICIENT MATRIX, STORED          DAPF 150
C                   COMPRESSED, I.E UPPER TRIANGULAR PART COLUMNWISE.   DAPF 160
C                   THE GIVEN RIGHT HAND SIDE OCCUPIES THE NEXT IP      DAPF 170
C                   LOCATIONS IN WORK. THE VERY LAST COMPONENT OF WORK  DAPF 180
C                   CONTAINS THE SQUARE SUM OF FUNCTION VALUES EO       DAPF 190
C                   THIS SCHEME OF STORAGE ALLOCATION IS PRODUCED E.G.  DAPF 200
C                   BY SUBROUTINE APLL.                                 DAPF 210
C                   THE GIVEN MATRIX IS FACTORED IN THE FORM            DAPF 220
C                   TRANSPOSE(T)*T AND THE GIVEN RIGHT HAND SIDE IS     DAPF 230
C                   DIVIDED BY TRANSPOSE(T).                            DAPF 240
C                   THE UPPER TRIANGULAR FACTOR T IS RETURNED IN WORK IFDAPF 250
C                   IOP EQUALS ZERO.                                    DAPF 260
C                   IN CASE OF NONZERO IOP THE CALCULATED SOLUTIONS ARE DAPF 270
C                   STORED IN THE COLUMNS OF TRIANGULAR ARRAY WORK OF   DAPF 280
C                   CORRESPONDING DIMENSION AND EO  IS REPLACED BY THE  DAPF 290
C                   SQUARE SUM OF THE ERRORS FOR FIT OF DIMENSION IRES. DAPF 300
C                   THE TOTAL DIMENSION OF WORK IS (IP+1)*(IP+2)/2      DAPF 310
C                   WORK MUST BE OF DOUBLE PRECISION                    DAPF 320
C           IP    - NUMBER OF FUNDAMENTAL FUNCTIONS USED FOR LEAST      DAPF 330
C                   SQUARES FIT                                         DAPF 340
C           IRES  - DIMENSION OF CALCULATED LEAST SQUARES FIT.          DAPF 350
C                   LET N1, N2, DENOTE THE FOLLOWING NUMBERS            DAPF 360
C                   N1 = MAXIMAL DIMENSION FOR WHICH NO LOSS OF         DAPF 370
C                        SIGNIFICANCE WAS INDICATED DURING FACTORIZATIONDAPF 380
C                   N2 = SMALLEST DIMENSION FOR WHICH THE SQUARE SUM OF DAPF 390
C                        THE ERRORS DOES NOT EXCEED TEST=ABS(ETA*FSQ)   DAPF 400
C                   THEN IRES=MINO(IP,N1) IF IOP IS NONNEGATIVE         DAPF 410
C                   AND  IRES=MINO(IP,N1,N2) IF IOP IS NEGATIVE         DAPF 420
C           IOP   - INPUT PARAMETER FOR SELECTION OF OPERATION          DAPF 430
C                   IOP = 0 MEANS TRIANGULAR FACTORIZATION, DIVISION OF DAPF 440
C                           THE RIGHT HAND SIDE BY TRANSPOSE(T) AND     DAPF 450
C                           CALCULATION OF THE SQUARE SUM OF ERRORS IS  DAPF 460
C                           PERFORMED ONLY                              DAPF 470
C                   IOP = +1 OR -1 MEANS THE SOLUTION OF DIMENSION IRES DAPF 480
C                           IS CALCULATED ADDITIONALLY                  DAPF 490
C                   IOP = +2 OR -2 MEANS ALL SOLUTIONS FOR DIMENSION ONEDAPF 500
C                           UP TO IRES ARE CALCULATED ADDITIONALLY      DAPF 510
C           EPS   - FELATIVE TOLERANCE FOR TEST ON LOSS OF SIGNIFICANCE.DAPF 520
C                   A SENSIBLE VALUE IS BETWEEN 1.E-10 AND 1.E-15       DAPF 530
C           ETA   - RELATIVE TOLERANCE FOR TOLERATED SQUARE SUM OF      DAPF 540
C                   ERRORS. A REALISTIC VALUE IS BETWEEN 1.E0 AND 1.E-15DAPF 550
C           IER   - RESULTANT ERROR PARAMETER                           DAPF 560
C                   IER =-1 MEANS NONPOSITIVE IP                        DAPF 570
C                   IER = 0 MEANS NO LOSS OF SIGNIFICANCE DETECTED      DAPF 580
C                           AND SPECIFIED TOLERANCE OF ERRORS REACHED   DAPF 590
C                   IER = 1 MEANS LOSS OF SIGNIFICANCE DETECTED OR      DAPF 600
C                           SPECIFIED TOLERANCE OF ERRORS NOT REACHED   DAPF 610
C                                                                       DAPF 620
C        REMARKS                                                        DAPF 630
C           THE ABSOLUTE TOLERANCE USED INTERNALLY FOR TEST ON LOSS OF  DAPF 640
C           SIGNIFICANCE IS TOL=ABS(EPS*SNGL(WORK(1))).                 DAPF 650
C           THE ABSOLUTE TOLERANCE USED INTERNALLY FOR THE SQUARE SUM OFDAPF 660
C           ERRORS IS ABS(ETA*SNGL(FSQ)).                               DAPF 670
C           IOP GREATER THAN 2 HAS THE SAME EFFECT AS IOP = 2.          DAPF 680
C           IOP LESS THAN -2 HAS THE SAME EFFECT AS IOP =-2.            DAPF 690
C           IRES = 0 MEANS THE ABSOLUTE VALUE OF EPS IS NOT LESS THAN   DAPF 700
C           ONE AND/OR WORK(1) IS NOT POSITIVE AND/OR IP IS NOT POSITIVEDAPF 710
C                                                                       DAPF 720
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DAPF 730
C           NONE                                                        DAPF 740
C                                                                       DAPF 750
C        METHOD                                                         DAPF 760
C           CALCULATION OF THE LEAST SQUARES FITS IS DONE USING         DAPF 770
C           CHOLESKYS SQUARE ROOT METHOD FOR SYMMETRIC FACTORIZATION.   DAPF 780
C           THE INCORPORATED TEST ON LOSS OF SIGNIFICANCE MEANS EACH    DAPF 790
C           RADICAND MUST BE GREATER THAN THE INTERNAL ABSOLUTE         DAPF 800
C           TOLERANCE TOL.                                              DAPF 810
C           IN CASE OF LOSS OF SIGNIFICANCE IN THE ABOVE SENSE ONLY A   DAPF 820
C           SUBSYSTEM OF THE NORMAL EQUATIONS IS SOLVED.                DAPF 830
C           IN CASE OF NEGATIVE IOP THE TRIANGULAR FACTORIZATION IS     DAPF 840
C           TERMINATED PREMATURELY EITHER IF THE SQUARE SUM OF THE      DAPF 850
C           ERRORS DOES NOT EXCEED ETA*FSQ OR IF THERE IS INDICATION    DAPF 860
C           FOR LOSS OF SIGNIFICANCE                                    DAPF 870
C                                                                       DAPF 880
C     ******************************************************************DAPF 890
C                                                                       DAPF 900
      SUBROUTINE DAPFS(WORK,IP,IRES,IOP,EPS,ETA,IER)                    DAPF 910
C                                                                       DAPF 920
C                                                                       DAPF 930
C        DIMENSIONED DUMMY VARIABLES                                    DAPF 940
      DIMENSION WORK(1)                                                 DAPF 950
      DOUBLE PRECISION WORK,SUM,PIV                                     DAPF 960
      IRES=0                                                            DAPF 970
C                                                                       DAPF 980
C        TEST OF SPECIFIED DIMENSION                                    DAPF 990
      IF(IP)1,1,2                                                       DAPF1000
C                                                                       DAPF1010
C        ERROR RETURN IN CASE OF ILLEGAL DIMENSION                      DAPF1020
    1 IER=-1                                                            DAPF1030
      RETURN                                                            DAPF1040
C                                                                       DAPF1050
C        INITIALIZE FACTORIZATION PROCESS                               DAPF1060
    2 IPIV=0                                                            DAPF1070
      IPP1=IP+1                                                         DAPF1080
      IER=1                                                             DAPF1090
      ITE=IP*IPP1/2                                                     DAPF1100
      IEND=ITE+IPP1                                                     DAPF1110
      TOL=ABS(EPS*SNGL(WORK(1)))                                        DAPF1120
      TEST=ABS(ETA*SNGL(WORK(IEND)))                                    DAPF1130
C                                                                       DAPF1140
C        START LOOP OVER ALL ROWS OF WORK                               DAPF1150
      DO 11 I=1,IP                                                      DAPF1160
      IPIV=IPIV+I                                                       DAPF1170
      JA=IPIV-IRES                                                      DAPF1180
      JE=IPIV-1                                                         DAPF1190
C                                                                       DAPF1200
C        FORM SCALAR PRODUCT NEEDED TO MODIFY CURRENT ROW ELEMENTS      DAPF1210
      JK=IPIV                                                           DAPF1220
      DO 9 K=I,IPP1                                                     DAPF1230
      SUM=0.D0                                                          DAPF1240
      IF(IRES)5,5,3                                                     DAPF1250
    3 JK=JK-IRES                                                        DAPF1260
      DO 4 J=JA,JE                                                      DAPF1270
      SUM=SUM+WORK(J)*WORK(JK)                                          DAPF1280
    4 JK=JK+1                                                           DAPF1290
    5 IF(JK-IPIV)6,6,8                                                  DAPF1300
C                                                                       DAPF1310
C        TEST FOR LOSS OF SIGNIFICANCE                                  DAPF1320
    6 SUM=WORK(IPIV)-SUM                                                DAPF1330
      IF(SNGL(SUM)-TOL)12,12,7                                          DAPF1340
    7 SUM=DSQRT(SUM)                                                    DAPF1350
      WORK(IPIV)=SUM                                                    DAPF1360
      PIV=1.D0/SUM                                                      DAPF1370
      GOTO 9                                                            DAPF1380
C                                                                       DAPF1390
C        UPDATE OFF-DIAGONAL TERMS                                      DAPF1400
    8 SUM=(WORK(JK)-SUM)*PIV                                            DAPF1410
      WORK(JK)=SUM                                                      DAPF1420
    9 JK=JK+K                                                           DAPF1430
C                                                                       DAPF1440
C        UPDATE SQUARE SUM OF ERRORS                                    DAPF1450
      WORK(IEND)=WORK(IEND)-SUM*SUM                                     DAPF1460
C                                                                       DAPF1470
C        RECORD ADDRESS OF LAST PIVOT ELEMENT                           DAPF1480
      IRES=IRES+1                                                       DAPF1490
      IADR=IPIV                                                         DAPF1500
C                                                                       DAPF1510
C        TEST FOR TOLERABLE ERROR IF SPECIFIED                          DAPF1520
      IF(IOP)10,11,11                                                   DAPF1530
   10 IF(SNGL(WORK(IEND))-TEST)13,13,11                                 DAPF1540
   11 CONTINUE                                                          DAPF1550
      IF(IOP)12,22,12                                                   DAPF1560
C                                                                       DAPF1570
C        PERFORM BACK SUBSTITUTION IF SPECIFIED                         DAPF1580
   12 IF(IOP)14,23,14                                                   DAPF1590
   13 IER=0                                                             DAPF1600
   14 IPIV=IRES                                                         DAPF1610
   15 IF(IPIV)23,23,16                                                  DAPF1620
   16 SUM=0.D0                                                          DAPF1630
      JA=ITE+IPIV                                                       DAPF1640
      JJ=IADR                                                           DAPF1650
      JK=IADR                                                           DAPF1660
      K=IPIV                                                            DAPF1670
      DO 19 I=1,IPIV                                                    DAPF1680
      WORK(JK)=(WORK(JA)-SUM)/WORK(JJ)                                  DAPF1690
      IF(K-1)20,20,17                                                   DAPF1700
   17 JE=JJ-1                                                           DAPF1710
      SUM=0.D0                                                          DAPF1720
      DO 18 J=K,IPIV                                                    DAPF1730
      SUM=SUM+WORK(JK)*WORK(JE)                                         DAPF1740
      JK=JK+1                                                           DAPF1750
   18 JE=JE+J                                                           DAPF1760
      JK=JE-IPIV                                                        DAPF1770
      JA=JA-1                                                           DAPF1780
      JJ=JJ-K                                                           DAPF1790
   19 K=K-1                                                             DAPF1800
   20 IF(IOP/2)21,23,21                                                 DAPF1810
   21 IADR=IADR-IPIV                                                    DAPF1820
      IPIV=IPIV-1                                                       DAPF1830
      GOTO 15                                                           DAPF1840
C                                                                       DAPF1850
C        NORMAL RETURN                                                  DAPF1860
   22 IER=0                                                             DAPF1870
   23 RETURN                                                            DAPF1880
      END                                                               DAPF1890
```

Subroutine APCH and DAPCH

These subroutines set up the normal equations for a polynomial least-squares fit to a given discrete function. The solution of these equations is determined by subroutine APFS. The best fitting polynomial is computed in terms of its Chebyshev expansion.

1. Mathematical background

Let f(x) be a function defined for $x = x_1, x_2, \ldots x_n$.

The problem is to determine the coefficients of the polynomial

$$p(x) = \sum_{i=1}^{m} c_i x^{i-1}, \text{ such that}$$

$$\sum_{i=1}^{n} (f(x_i) - p(x_i))^2 = \text{minimum}$$

This problem leads to a system of linear equations AC = R where C is the vector of unknown coefficients, A is the m by m symmetric positive definite matrix with elements

$$a_{jk} = \sum_{i=1}^{n} x_i^{j-1} x_i^{k-1}$$

and R is the vector with elements

$$r_j = \sum_{i=1}^{n} f(x_i) x_i^{j-1}$$

If there is a weight w_i corresponding to each point $(x_i, f(x_i))$, the ith term of the sums for the a_{jk} and r_j must be multiplied by w_i.

In practice the positive definite matrix A is badly ill-conditioned. Therefore, the straightforward method described above is feasible only for polynomials of low degree, about 3 or 4.

Use of Chebyshev polynomials instead of monomials results in a remarkable improvement of the condition of the normal equations, provided the arguments have a sensible distribution (for example, equidistant).

Let x_L and x_R denote the leftmost and rightmost arguments. By means of the linear transformation

$$t(x) = \frac{2x - (x_L + x_R)}{x_R - x_L} = x_D \cdot x + x_0$$

the argument range $x_L \le x \le x_R$ is reduced to the argument range $-1 \le t \le +1$.

The polynomial p(x) is calculated in the form of its Chebyshev expansion $p(x) = b_1 T_0(t) + b_2 T_1(t) + \ldots + b_m T_{m-1}(t)$, where $T_k(t)$ is the Chebyshev polynomial of degree k.

Then the vector B of the unknown coefficients b_i is a solution of the matrix equation AB = R, where A is an m by m symmetric positive definite matrix with elements

$$a_{jk} = \sum_{i=1}^{n} T_{j-1}(t(x_i)) T_{k-1}(t(x_i))$$

and R is a vector with elements

$$r_j = \sum_{i=1}^{n} T_{j-1}(t(x_i)) f(x_i)$$

The values of the Chebyshev polynomials for the argument t are calculated by means of the three-term recurrence equation:

$$T_k(t) = 2t \cdot T_{k-1}(t) - T_{k-2}(t); \; k \ge 2$$

with starting values $T_0(t) = 1$, $T_1(t) = t$.

In setting up the matrix A, time is saved by using the identity $2T_j \cdot T_k = T_{j+k} + T_{j-k}$.

a. Remarks

The Chebyshev expansion of the polynomial p(x) gives a much better indication of the accuracy of the approximation than the coefficient vector of the polynomial itself. If the specified degree of the polynomial is too high, the last terms of the Chebyshev expansion will be uniformly small compared to the coefficients in front. The degree may be reduced by the number of small trailing coefficients without unduly enlarging the overall error.

An upper bound for the error introduced by neglecting the last terms of the Chebyshev expansion is given by the sum of the absolute values of these terms. Normally, transformation of the Chebyshev expansions in t(x) to ordinary polynomials in x results in severe loss of accuracy. Therefore, no attempt is made to return the polynomial expansions, which is no disadvantage since the Chebyshev expansion may be evaluated effectively for a specified argument x using subroutines CNPS and DCNPS with argument $t = x \cdot x_D + x_0$ and the calculated coefficient vector of the Chebyshev expansion.

The transformation of the calculated Chebyshev expansion to an ordinary polynomial may be accomplished using subroutines TCNP and DTCNP.

. Programming considerations

ıput arrays passed to subroutines APCH and)APCH are DATI and WORK. DATI contains the iven data set to be approximated, consisting of n rgument values $x_1, \ldots, x_n$ followed by correspond-ıg function values $f_1, \ldots, f_n$ and by corresponding 'eight values (if any). Constant weighting (value ne) is implied if DATI (2n + 1) contains a nonposi-ve value.

Instead of arguments x_i, we use $t_i = x_i \cdot x_D + x_0$ ɔr setting up the normal equations. The constants $_D$ and x_0 are returned as result values.

The normal equations are generated in the tri-ngular array WORK. Only the upper triangular art of the symmetric coefficient matrix is gener-ted and stored columnwise, followed immediately y the right-hand side and by the weighted square um of function values.

This storage allocation scheme is required by ubroutine APFS and DAPFS, which may be used ɔr calculation of the solution of the normal equa-ons. Exceptions are indicated by the error param-ter IER, which is set to -1 if n and m are not both ositive or if n is less than m, and to +1 if the argu-ıents are coinciding; otherwise to zero, indicating uccessful operation.

```
                                                                        APCH  10
*****************************************************************APCH  20
                                                                        APCH  30
   SUBROUTINE APCH                                                      APCH  40
                                                                        APCH  50
   PURPOSE                                                              APCH  60
      SET UP NORMAL EQUATIONS OF LEAST SQUARES FIT IN TERMS OF          APCH  70
      CHEBYSHEV POLYNOMIALS FOR A GIVEN DISCRETE FUNCTION               APCH  80
                                                                        APCH  90
   USAGE                                                                APCH 100
      CALL APCH(DATI,N,IP,XO,XC,WORK,IER)                               APCH 110
                                                                        APCH 120
   DESCRIPTION OF PARAMETERS                                            APCH 130
      DATI  - VECTOR OF DIMENSION 3*N (OR DIMENSION 2*N+1)              APCH 140
              CONTAINING THE GIVEN ARGUMENTS, FOLLOWED BY THE           APCH 150
              FUNCTION VALUES AND N (RESPECTIVELY 1) WEIGHT             APCH 160
              VALUES. THE CONTENT OF VECTOR DATI REMAINS                APCH 170
              UNCHANGED.                                                APCH 180
      N     - NUMBER OF GIVEN POINTS                                    APCH 190
      IP    - DIMENSION OF LEAST SQUARES FIT, I.E. NUMBER OF            APCH 200
              CHEBYSHEV POLYNOMIALS USED AS FUNDAMENTAL FUNCTIONS APCH 210
              IP SHOULD NOT EXCEED N                                    APCH 220
      XO    - RESULTANT MULTIPLICATIVE CONSTANT FOR LINEAR              APCH 230
              TRANSFORMATION OF ARGUMENT RANGE                          APCH 240
      XC    - RESULTANT ADDITIVE CONSTANT FOR LINEAR                    APCH 250
              TRANSFORMATION OF ARGUMENT RANGE                          APCH 260
      WORK  - WORKING STORAGE OF DIMENSION (IP+1)*(IP+2)/2              APCH 270
              ON RETURN WORK CONTAINS THE SYMMETRIC COEFFICIENT         APCH 280
              MATRIX OF THE NORMAL EQUATIONS IN COMPRESSED FORM         APCH 290
              FOLLOWED IMMEDIATELY BY RIGHT HAND SIDE                   APCH 300
              AND SQUARE SUM OF FUNCTION VALUES                         APCH 310
      IER   - RESULTING ERROR PARAMETER                                 APCH 320
              IER =-1 MEANS FORMAL ERRORS IN DIMENSION                  APCH 330
              IER = 0 MEANS NO ERRORS                                   APCH 340
              IER = 1 MEANS COINCIDING ARGUMENTS                        APCH 350
                                                                        APCH 360
   REMARKS                                                              APCH 370
      NO WEIGHTS ARE USED IF THE VALUE OF DATI(2*N+1) IS                APCH 380
      NOT POSITIVE.                                                     APCH 390
      EXECUTION OF SUBROUTINE APCH IS A PREPARATORY STEP FOR            APCH 400
      CALCULATION OF LEAST SQUARES FITS IN CHEBYSHEV POLYNOMIALS        APCH 410
      IT SHOULD BE FOLLOWED BY EXECUTION OF SUBROUTINE APFS             APCH 420
                                                                        APCH 430
   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                        APCH 440
      NONE                                                              APCH 450
                                                                        APCH 460
   METHOD                                                               APCH 470
      THE LEAST SQUARE FIT IS DETERMINED USING CHEBYSHEV                APCH 480
      POLYNOMIALS AS FUNDAMENTAL FUNCTION SYSTEM.                       APCH 490
      THE METHOD IS DISCUSSED IN THE ARTICLE                            APCH 500
      A.T.BERZTISS, LEAST SQUARES FITTING TO IRREGULARLY SPACED         APCH 510
      DATA, SIAM REVIEW, VOL.6, ISS.3, 1964, PP. 203-227.               APCH 520
                                                                        APCH 530
*****************************************************************APCH 540
                                                                        APCH 550
   SUBROUTINE APCH(DATI,N,IP,XO,XC,WORK,IER)                            APCH 560
                                                                        APCH 570
                                                                        APCH 580
   DIMENSIONED DUMMY VARIABLES                                          APCH 590
   DIMENSION DATI(1),WORK(1)                                            APCH 600
C
C        CHECK FOR FORMAL ERRORS IN SPECIFIED DIMENSIONS                APCH 610
      IF(N-1)15,20,1                                                    APCH 620
    1 IF(IP)15,19,2                                                     APCH 630
C                                                                       APCH 640
C        SEARCH SMALLEST AND LARGEST ARGUMENT                           APCH 650
    2 IF(IP-N)3,3,19                                                    APCH 660
    3 XA=DATI(1)                                                        APCH 670
      XC=XA                                                             APCH 680
      XE=0.                                                             APCH 690
      DO 7 I=1,N                                                        APCH 700
      XP=DATI(I)                                                        APCH 710
      IF(XA-XP)5,5,4                                                    APCH 720
    4 XA=XP                                                             APCH 730
    5 IF(XC-XP)6,7,7                                                    APCH 740
    6 XC=XP                                                             APCH 750
    7 CONTINUE                                                          APCH 760
C                                                                       APCH 770
C        INITIALIZE CALCULATION OF NORMAL EQUATIONS                     APCH 780
      XC=XC-XA                                                          APCH 790
      M=(IP*(IP+1))/2                                                   APCH 800
      IENDM=M+IP+1                                                      APCH 810
      MT2=IP+IP                                                         APCH 820
      MT2M=MT2-1                                                        APCH 830
C                                                                       APCH 840
C        SET WORKING STORAGE AND RIGHT HAND SIDE TO ZERO                APCH 850
      DO 8 I=1,IP                                                       APCH 860
      J=MT2-I                                                           APCH 870
      WORK(J)=0.                                                        APCH 880
      WORK(I)=0.                                                        APCH 890
      K=M+I                                                             APCH 900
    8 WORK(K)=0.                                                        APCH 910
C                                                                       APCH 920
C        CHECK FOR DEGENERATE ARGUMENT RANGE                            APCH 930
      IF(XC)20,20,9                                                     APCH 940
C                                                                       APCH 950
C        CALCULATE CONSTANTS FOR RECTICTION OF ARGUMENTS                APCH 960
    9 XC=-(XC+XA)/XC                                                    APCH 970
      XC=2./XC                                                          APCH 980
      SUM=0.                                                            APCH 990
C                                                                       APCH1000
C        START GREAT LOOP OVER ALL GIVEN POINTS                         APCH1010
      DO 15 I=1,N                                                       APCH1020
      T=DATI(I)*XC+XC                                                   APCH1030
      J=I+N                                                             APCH1040
      DF=DATI(J)                                                        APCH1050
C                                                                       APCH1060
C        CALCULATE AND STORE VALUES OF CHEBYSHEV POLYNOMIALS            APCH1070
C        FOR ARGUMENT T                                                 APCH1080
      XA=1.                                                             APCH1090
      XP=T                                                              APCH1100
      IF(DATI(2*N+1))11,11,10                                           APCH1110
   10 J=J+N                                                             APCH1120
      XA=DATI(J)                                                        APCH1130
      XP=T*XA                                                           APCH1140
   11 T=T+T                                                             APCH1150
      SUM=SUM+DF*DF*XA                                                  APCH1160
      DF=DF+DF                                                          APCH1170
      J=1                                                               APCH1180
   12 K=M+J                                                             APCH1190
      WORK(K)=WORK(K)+DF*XA                                             APCH1200
   13 WORK(J)=WORK(J)+XA                                                APCH1210
      IF(J-MT2M)14,15,15                                                APCH1220
   14 J=J+1                                                             APCH1230
      XE=T*XP-XA                                                        APCH1240
      XA=XP                                                             APCH1250
      XP=XE                                                             APCH1260
      IF(J-IP)12,12,13                                                  APCH1270
   15 CONTINUE                                                          APCH1280
      WORK(IENDM)=SUM+SUM                                               APCH1290
C                                                                       APCH1300
C        CALCULATE MATRIX OF NORMAL EQUATIONS                           APCH1310
      LL=M                                                              APCH1320
      KK=MT2M                                                           APCH1330
      JJ=1                                                              APCH1340
      K=KK                                                              APCH1350
      DO 18 J=1,M                                                       APCH1360
      WORK(LL)=WORK(K)+WORK(JJ)                                         APCH1370
      LL=LL-1                                                           APCH1380
      IF(K-JJ)16,16,17                                                  APCH1390
   16 KK=KK-2                                                           APCH1400
      K=KK                                                              APCH1410
      JJ=1                                                              APCH1420
      GOTO 18                                                           APCH1430
   17 JJ=JJ+1                                                           APCH1440
      K=K-1                                                             APCH1450
   18 CONTINUE                                                          APCH1460
      IER=0                                                             APCH1470
      RETURN                                                            APCH1480
C                                                                       APCH1490
C        ERROR RETURN IN CASE OF FORMAL ERRORS                          APCH1500
   15 IER=-1                                                            APCH1510
      RETURN                                                            APCH1520
C                                                                       APCH1530
C        ERROR RETURN IN CASE OF COINCIDING ARGUMENTS                   APCH1540
   20 IER=1                                                             APCH1550
      RETURN                                                            APCH1560
      END                                                               APCH1570
                                                                        APCH1580
```

```
C                                                                       DAPC  10
C     ****************************************************************DAPC  20
C                                                                       DAPC  30
C        SUBROUTINE DAPCH                                               DAPC  40
C                                                                       DAPC  50
C        PURPOSE                                                        DAPC  60
C           SET UP NORMAL EQUATIONS OF LEAST SQUARES FIT IN TERMS OF    DAPC  70
C           CHEBYSHEV POLYNOMIALS FOR A GIVEN DISCRETE FUNCTION         DAPC  80
C                                                                       DAPC  90
C        USAGE                                                          DAPC 100
C           CALL DAPCH(DATI,N,IP,XO,XO,WORK,IER)                        DAPC 110
C                                                                       DAPC 120
C        DESCRIPTION OF PARAMETERS                                      DAPC 130
C           DATI  - VECTOR OF DIMENSION 3*N (OR DIMENSION 2*N+1)        DAPC 140
C                   CONTAINING THE GIVEN ARGUMENTS, FOLLOWED BY THE     DAPC 150
C                   FUNCTION VALUES AND N (RESPECTIVELY 1) WEIGHT       DAPC 160
C                   VALUES. THE CONTENT OF VECTOR DATI REMAINS          DAPC 170
C                   UNCHANGED.                                          DAPC 180
C                   DATI MUST BE OF DOUBLE PRECISION                    DAPC 190
C           N     - NUMBER OF GIVEN POINTS                              DAPC 200
C           IP    - DIMENSION OF LEAST SQUARES FIT, I.E. NUMBER OF      DAPC 210
C                   CHEBYSHEV POLYNOMIALS USED AS FUNDAMENTAL FUNCTIONS DAPC 220
C                   IP SHOULD NOT EXCEED N                              DAPC 230
C           XO    - RESULTANT MULTIPLICATIVE CONSTANT FOR LINEAR        DAPC 240
C                   TRANSFORMATION OF ARGUMENT RANGE                    DAPC 250
C                   XO MUST BE DOUBLE PRECISION                         DAPC 260
C           XO    - RESULTANT ADDITIVE CONSTANT FOR LINEAR              DAPC 270
C                   TRANSFORMATION OF ARGUMENT RANGE                    DAPC 280
C                   XO MUST BE DOUBLE PRECISION                         DAPC 290
C           WORK  - WORKING STORAGE OF DIMENSION (IP+1)*(IP+2)/2        DAPC 300
```

```
C                 ON RETURN WORK CONTAINS THE SYMMETRIC COEFFICIENT
C                 MATRIX OF THE NORMAL EQUATIONS IN COMPRESSED FORM
C                 FOLLOWED IMMEDIATELY BY RIGHT HAND SIDE
C                 AND SQUARE SUM OF FUNCTION VALUES
C                 WORK MUST BE OF DOUBLE PRECISION
C         IER   - RESULTING ERROR PARAMETER
C                 IER =-1 MEANS FORMAL ERRORS IN DIMENSION
C                 IER = 0 MEANS NO ERRORS
C                 IER = 1 MEANS COINCIDING ARGUMENTS
C
C       REMARKS
C          NO WEIGHTS ARE USED IF THE VALUE OF DATI(2*N+1) IS
C          NOT POSITIVE.
C          EXECUTION OF SUBROUTINE DAPCH IS A PREPARATORY STEP FOR
C          CALCULATION OF LEAST SQUARES FITS IN CHEBYSHEV POLYNOMIALS
C          IT SHOULD BE FOLLOWED BY EXECUTION OF SUBROUTINE DAPFS
C
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C          NONE
C
C       METHOD
C          THE LEAST SQUARE FIT IS DETERMINED USING CHEBYSHEV
C          POLYNOMIALS AS FUNDAMENTAL FUNCTION SYSTEM.
C          THE METHOD IS DISCUSSED IN THE ARTICLE
C          A.T.BERZTISS, LEAST SQUARES FITTING TO IRREGULARLY SPACED
C          DATA, SIAM REVIEW, VOL.6, ISS.3, 1964, PP. 203-227.
C
C     ..................................................................
C
      SUBROUTINE DAPCH(DATI,N,IP,XO,XO,WORK,IER)
C
C
C       DIMENSIONED DUMMY VARIABLES
      DIMENSION DATI(1),WORK(1)
      DOUBLE PRECISION DATI,WORK,XO,XO,XA,XE,XM,DF,T,SUM
C
C        CHECK FOR FORMAL ERRORS IN SPECIFIED DIMENSIONS
      IF(N-1)19,20,1
    1 IF(IP)19,19,2
C
C        SEARCH SMALLEST AND LARGEST ARGUMENT
    2 IF(IP-N)3,3,19
    3 XA=DATI(1)
      XO=XA
      XE=0.D0
      DO 7 I=1,N
      XM=DATI(I)
      IF(XA-XM)5,5,4
    4 XA=XM
    5 IF(XO-XM)6,7,7
    6 XO=XM
    7 CONTINUE
C
C        INITIALIZE CALCULATION OF NORMAL EQUATIONS
      XD=XO-XA
      M=(IP*(IP+1))/2
      IEND=M+IP+1
      MT2=IP+IP
      MT2M=MT2-1
C
C        SET WORKING STORAGE AND RIGHT HAND SIDE TO ZERO
      DO 8 I=1,IP
      J=MT2-I
      WORK(J)=0.D0
      WORK(I)=0.D0
      K=M+I
    8 WORK(K)=0.D0
C
C        CHECK FOR DEGENERATE ARGUMENT RANGE
      IF(XD)20,20,9
C
C        CALCULATE CONSTANTS FOR REDUCTION OF ARGUMENTS
    9 XO=-(XO+XA)/XD
      XD=2.D0/XD
      SUM=0.D0
C
C        START GREAT LOOP OVER ALL GIVEN POINTS
      DO 15 I=1,N
      T=DATI(I)*XD+XO
      J=I+N
      DF=DATI(J)
C
C        CALCULATE AND STORE VALUES OF CHEBYSHEV POLYNOMIALS
C        FOR ARGUMENT T
      XA=1.D0
      XM=T
      IF(DATI(2*N+1))11,11,10
   10 J=J+N
      XA=DATI(J)
      XM=T*XA
   11 T=T+T
      SUM=SUM+DF*DF*XA
      DF=DF+DF
      J=1
   12 K=M+J
      WORK(K)=WORK(K)+DF*XA
   13 WORK(J)=WORK(J)+XA
      IF(J-MT2M)14,15,15
   14 J=J+1
      XE=T*XM-XA
      XA=XM
      XM=XE
      IF(J-IP)12,12,13
   15 CONTINUE
      WORK(IEND)=SUM+SUM
C
C        CALCULATE MATRIX OF NORMAL EQUATIONS
      LL=M
      KK=MT2M
      JJ=1
      K=KK
      DO 18 J=1,M
      WORK(LL)=WORK(K)+WORK(JJ)
      LL=LL-1
      IF(K-JJ)16,16,17
   16 KK=KK-2
      K=KK
      JJ=1
      GOTO 18
   17 JJ=JJ+1
      K=K-1
   18 CONTINUE
      IER=0
      RETURN
C
C        ERROR RETURN IN CASE OF FORMAL ERRORS
   19 IER=-1
      RETURN
C
C        ERROR RETURN IN CASE OF COINCIDING ARGUMENTS
   20 IER=1
      RETURN
      END
```

Subroutines ARAT and DARAT

These subroutines compute the best rational approximation $P(x)/Q(x)$ to a discrete function, $f(x)$, in the least-squares sense. The argument range should be the interval $[-1, +1]$.

Subroutines FRAT and DFRAT are an integral part of the procedure and have no independent meaning.

1. Mathematical background

Given positive integers ip, iq, and n, arguments x_i, function values f_i, and positive weights w_i, $i=1, 2, \ldots, n$, the problem is to determine the coefficients of the polynomials

$$P(x) = p_1 + p_2 x + \ldots + p_{ip} x^{ip-1}$$

$$Q(x) = q_1 + q_2 x + \ldots + q_{iq} x^{iq-1}$$

such that

$$F = \sum_{i=1}^{n} w_i \left[f_i - \frac{P(x_i)}{Q(x_i)} \right]^2 = \text{minimum} \qquad (1)$$

A direct approach to the problem (1) leads to a system of nonlinear equations. Therefore the problem is reformulated in the following way.

Let $P_o(x)$ and $Q_o(x)$ be an approximate solution to (1). The problem is then to determine increments $\Delta P_o(x)$ and $\Delta Q_o(x)$ such that

$$F = \sum_{i=1}^{n} w_i \left[f_i - \frac{P_o(x_i) + \Delta P_o(x_i)}{Q_o(x_i) + \Delta Q_o(x_i)} \right]^2 = \text{minimum} \qquad (1')$$

Without loss of generality it may be assumed that the constant term of Q_o remains unchanged, that is that the constant term of ΔQ_o is zero, since numerator and denominator of the rational approximation are determined only up to a common factor.

Linearizing $\dfrac{P_o + \Delta P_o}{Q_o + \Delta Q_o}$ with respect to ΔP_o and ΔQ_o,

$$\frac{P_o + \Delta P_o}{Q_o + \Delta Q_o} \approx \frac{P_o}{Q_o} \left[1 + \frac{\Delta P_o}{P_o} - \frac{\Delta Q_o}{Q_o} \right] \qquad (2)$$

and using (2) in (1') gives the approximation $F \approx F_o$ where

$$F_o = \sum_{i=1}^{n} \frac{w_i}{Q_o(x_i)^2} \left[f_i Q_o(x_i) - P_o(x_i) - \Delta P_o(x_i) + \frac{P_o(x_i)}{Q_o(x_i)} \Delta Q_o(x_i) \right]^2$$

Defining

$$f_i^* = f_i Q_o(x_i) - P_o(x_i)$$

$$w_i^* = w_i / Q_o(x_i)^2$$

and a set of functions

$$g_k(x) = -\frac{P_o(x)}{Q_o(x)} x^k \qquad k = 1, 2, \ldots, iq-1$$

$$g_k(x) = x^{k-iq} \qquad k = iq, iq+1, \ldots, iq+ip-1$$

we may write

$$F_o = \sum_{i=1}^{n} w_i^* \left[f_i^* - g(x_i) \right]^2$$

where

$$g(x) = \sum_{i=1}^{m} c_i g_i(x) \qquad m = iq + ip - 1$$

and the c_i are the coefficients of $\Delta Q_o(x)$ and $\Delta P_o(x)$:

$$\Delta Q_o(x) = \sum_{i=1}^{iq-1} c_i x^i$$

$$\Delta P_o(x) = \sum_{i=iq}^{m} c_i x^{i-iq}$$

Now the coefficients c_i are determined so that F_o = minimum. This is a problem of the type discussed in APLL with function values f_i^*, weights w_i^* and fundamental functions $g_i(x)$, and leads to a system of linear equations

$$AC = R$$

with symmetric positive semi-definite coefficient matrix A (see APLL writeup for a desciption of A and R).

In subroutine ARAT, APLL is used to set up the normal equations AC = R, and APFS is used to solve the system for the coefficient vector C.

Given a solution ΔP_o and ΔQ_o to the problem F_o = minimum, a new approximation is formed using

$$P_1 = P_o + \Delta P_o$$

$$Q_1 = Q_o + \Delta Q_o$$

In general, given approximations P_k and Q_k, increments ΔP_k and ΔQ_k are determined as indicated above, and the next approximations are

$$P_{k+1} = P_k + \Delta P_k$$

$$Q_{k+1} = Q_k + \Delta Q_k \qquad (3)$$

This process is started with some initial approximation (usually with $P_o(x) \equiv 0$, $Q_o(x) \equiv 1$) and is stopped when there is indication of convergence (see below).

a. Refinement of the calculation procedure

The above approach has several defects:

(1) There is no guarantee for improvement of the rational fit obtained with successive iterations if the initial approximation is not close enough to a true solution.

(2) Zeros may come into the denominator, causing a complete breakdown of the procedure.

(3) The normal equations of the linearized problem are badly ill-conditioned if monomials are used as fundamental functions.

The first difficulty is resolved using an approach suggested by D. Braess in reference (1). Let

$$S_k = \sum_{i=1}^{n} w_i \left[f_i - \frac{P_k(x_i)}{Q_k(x_i)} \right]^2$$

Then instead of using the relations (3) we use

$$P_{k+1} = P_k + r \Delta P_k$$

$$Q_{k+1} = Q_k + r \Delta Q_k$$

where the relaxation factor r is chosen so that S_{k+1} is close to a minimum when regarded as a function of r, and $S_{k+1} \leq S_k$. If this cannot be done, the error parameter IER is set to IER = 1, and the process is stopped. Choosing r sufficiently small also prevents zeros from entering the denominator.

The condition of the normal equations is improved remarkably by using Chebyshev polynomials $T_k(x)$ instead of monomials x^k as fundamental functions; that is, the numerator and denominator are calculated in terms of their Chebyshev expansions. The argument range is assumed to be (-1, +1).

If it turns out that the normal equations are still ill-conditioned, the diagonal elements of the coefficient matrix are slightly increased. This modification was suggested by D. W. Marquardt in reference (2). In this case again the calculated increment is modified by a relaxation factor r; but now r may take on values even greater than 1 if this does not imply zeros coming into the denominator.

b. Convergence

Let T_k denote the sum of the absolute values of the coefficients of $P_k(x)$ and $Q_k(x)$. Then, assuming that the Marquardt modification was not needed, the process is stopped if either

$$\frac{S_k - S_{k+1}}{S_{k+1}} \leq r\,\epsilon$$

or

$$\frac{|T_k - T_{k+1}|}{T_{k+1}} \leq r\,\epsilon$$

or

$$S_{k+1} \leq \eta\, T_{k+1}$$

Here $\epsilon = 10^{-5}$ and $\eta = 10^{-11}$. If the diagonal terms of the coefficient matrix of the normal equations were increased, only the last test is applied. If none of these conditions is met, the process is repeated until the maximum number of iterations k = LIMIT is reached (LIMIT is set to 20 in ARAT).

c. Remarks

(1) The user must specify the dimension of the denominator and of the numerator of the rational approximation he wants to calculate. The calculation procedure is somewhat sensitive to getting adequate dimensions. If the specified dimensions are unrealistically high, the calculation procedure may break down due to a very poor convergence. In this case it is best to repeat the calculation with decreased dimensions.

(2) The initial approximation which the user has the option to specify must be such that the denominator is free of zeros over the range of approximation (-1, +1). The standard initial approximation is $P(x) \equiv 0$ and $Q(x) \equiv 1$.

(3) The arguments of the given discrete functions should be reduced to the interval (-1, +1). Arguments absolutely greater than 1 cause error propagation in the calculation of Chebyshev polynomials and may produce dubious results.

(4) Numerical experience has shown that the dimensions used in single precision should not exceed 5, and in double precision should not be above 10.

(5) Transformation of Chebyshev expansions to ordinary polynomials normally results in severe loss of accuracy. Therefore, no attempt is made to return the polynomial coefficients. This transformation may be easily accomplished through subroutines TCNP and DTCNP. The calculated Chebyshev expansions may be evaluated for a specified argument using subroutines CNPS and DCNPS.

2. Programming considerations

Arrays passed to subroutines ARAT and DARAT are DATI, WORK, and P. DATI contains the given data set to be approximated, consisting of n arguments $x_1, \ldots, x_n$ followed by the corresponding function values $f_1, \ldots, f_n$ and by weights $w_1, \ldots, w_n$. Constant weighting (value one) is implied if DATI (2n + 1) contains a nonpositive value.

WORK is used as working storage. On return it contains the values of the numerator in WORK (n + 1) up to WORK (2n) and the values of the denominator in WORK (2n + 1) up to WORK (3n). The storage positions WORK (1) up to WORK (n) and WORK (3n + 1) up to WORK (4n) are used to record the current nonmoderated values of change of numerator and denominator respectively. The remaining (ip + iq)(ip + iq + 1) + 1 storage locations of WORK are used to store the normal equations in compressed form twice, in a working area as well as in a save area. This is necessary to avoid recalculation of the normal equations in case of an ill-conditioned coefficient matrix.

P is a vector of dimension ip + iq containing the calculated rational approximation on return. The coefficients of the Chebyshev expansion of the denominator are stored first, immediately followed by the coefficients of the Chebyshev expansion of the numerator. Coefficients are ordered from low to

high. P may also be used as an input array containing the initial approximation in terms of its Chebyshev expansions (with same ordering) optionally.

The normal equations are generated internally by means of subroutines APLL and DAPLL in the storage allocation scheme which is required by subroutines APFS and DAPFS. Subroutines FRAT and DFRAT are simply the interface between subroutines ARAT, DARAT and APLL, DAPLL used for handling the given data set and the specified fundamental functions.

The error parameter IER gives an indication of success or failure of the procedure.

- IER = 0 — The process converged in the sense mentioned above.
- IER = -1 — Formal error occurred in specified dimensions or initial approximations such as violation of any of the restrictions $n > 0$, $ip > 0$, $iq > 0$, $n \geq ip + iq$ or $Q_0(x) \neq 0$ on $(-1, 1)$. The latter could occur only if an optional initial approximation was used.
- IER = 1 — The Marquardt modification was tried LIMIT times without success, or a suitable relaxation factor could not be found.
- IER = 2 — The process has not converged after LIMIT steps.

In all cases where IER ≥ 0 there is a calculated approximation available, but only in the case IER = 0 is there strong indication that this approximation is close to the best rational fit in the least-squares sense.

For reference see:

(1) D. Braess, "Ueber Daempfung bei Minimalisierungsverfahren", Computing (1966) vol. 1, ed. 3, pp 264 - 272.

(2) D. W. Marquardt, "An algorithm for least-squares estimation of non-linear parameters", JSIAM (1963), vol. II, ed. 2, pp. 431— 441.


```
C
C     ..................................................................
C
C        SUBROUTINE ARAT
C
C        PURPOSE
C           CALCULATE BEST RATIONAL APPROXIMATION OF A DISCRETE
C           FUNCTION IN THE LEAST SQUARES SENSE
C
C        USAGE
C           CALL ARAT(DATI,N,WORK,P,IP,IQ,IER)
C
C        DESCRIPTION OF PARAMETERS
C           DATI  - TWODIMENSIONAL ARRAY WITH 3 COLUMNS AND N ROWS
C                   THE FIRST COLUMN MUST CONTAIN THE GIVEN ARGUMENTS,
C                   THE SECOND COLUMN THE GIVEN FUNCTION VALUES AND
C                   THE THIRD COLUMN THE GIVEN WEIGHTS IF ANY.
C                   IF NO WEIGHTS ARE TO BE USED THEN THE THIRD
C                   COLUMN MAY BE DROPPED , EXCEPT THE FIRST ELEMENT
C                   WHICH MUST CONTAIN A NONPOSITIVE VALUE
C           N     - NUMBER OF NODES OF THE GIVEN DISCRETE FUNCTION
C           WORK  - WORKING STORAGE WHICH IS OF DIMENSION
C                   (IP+IQ)*(IP+IQ+1)+4*N+1 AT LEAST.
C                   ON RETURN THE VALUES OF THE NUMERATOR ARE CONTAINED
C                   IN WORK(N+1) UP TO WORK(2*N), WHILE THE VALUES OF
C                   THE DENOMINATOR ARE STORED IN WORK(2*N+1) UP TO
C                   WORK(3*N)
C           P     - RESULTANT COEFFICIENT VECTOR OF DENOMINATOR AND
C                   NUMERATOR. THE DENOMINATOR IS STORED IN FIRST IQ
C                   LOCATIONS, THE NUMERATOR IN THE FOLLOWING IP
C                   LOCATIONS.
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH.
C           IP    - DIMENSION OF THE NUMERATOR   (INPUT VALUE)
C           IQ    - DIMENSION OF THE DENOMINATOR (INPUT VALUE)
C           IER   - RESULTANT ERROR PARAMETER
C                   IER =-1 MEANS FORMAL ERRORS
C                   IER = 0 MEANS NO ERRORS
C                   IER = 1,2 MEANS POOR CONVERGENCE OF ITERATION
C                   IER IS ALSO USED AS INPUT VALUE
C                   A NONZERO INPUT VALUE INDICATES AVAILABILITY OF AN
C                   INITIAL APPROXIMATION STORED IN P
C
C        REMARKS
C           THE COEFFICIENT VECTORS OF THE DENOMINATOR AND NUMERATOR
C           OF THE RATIONAL APPROXIMATION ARE BOTH STORED IN P
C           STARTING WITH LOW POWERS (DENOMINATOR FIRST).
C           IP+IQ MUST NOT EXCEED N, ALL THREE VALUES MUST BE POSITIVE.
C           SINCE CHEBYSHEV POLYNOMIALS ARE USED AS FUNDAMENTAL
C           FUNCTIONS, THE ARGUMENTS SHOULD BE REDUCED TO THE INTERVAL
C           (-1,1). THIS CAN ALWAYS BE ACCOMPLISHED BY MEANS OF A LINEAR
C           TRANSFORMATION OF THE ORIGINALLY GIVEN ARGUMENTS.
C           IF A FIT IN OTHER FUNCTIONS IS REQUIRED, CNP AND CNPS MUST
C           BE REPLACED BY SUBROUTINES WHICH ARE OF ANALOGOUS DESIGN.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           APLL, APFS, FRAT, CNPS, CNP
C           CNP IS REQUIRED WITHIN FRAT
C
C        METHOD
C           THE ITERATIVE SCHEME USED FOR CALCULATION OF THE
C           APPROXIMATION IS REPEATED SOLUTION OF THE NORMAL EQUATIONS
C           WHICH ARE OBTAINED BY LINEARIZATION.
C           A REFINED TECHNIQUE OF THIS LINEAR LEAST SQUARES APPROACH
C           IS USED WHICH GUARANTEES THAT THE DENOMINATOR IS FREE OF
C           ZEROES WITHIN THE APPROXIMATION INTERVAL.
C           FOR REFERENCE SEE
C           D.BRAESS, UEBER DAEMPFUNG BEI MINIMALISIERUNGSVERFAHREN,
C           COMPUTING(1966), VOL.1, ED.3, PP.264-272.
C           D.W.MARQUARDT, AN ALGORITHM FOR LEAST-SQUARES ESTIMATION
C           OF NONLINEAR PARAMETERS,
C           JSIAM(1963), VOL.11, ED.2, PP.431-441.
C
C     ..................................................................
      SUBROUTINE ARAT(DATI,N,WORK,P,IP,IQ,IER)
C
C
      EXTERNAL FRAT
C
C        DIMENSIONED LOCAL VARIABLE
      DIMENSION IERV(3)
C
C        DIMENSIONED DUMMY VARIABLES
      DIMENSION DATI(1),WORK(1),P(1)
C
C        INITIALIZE TESTVALUES
      LIMIT=20
      ETA =1.E-11
      EPS=1.E-5
C
C        CHECK FOR FORMAL ERRORS
      IF(N)4,4,1
    1 IF(IP)4,4,2
    2 IF(IQ)4,4,3
    3 IPQ=IP+IQ
      IF(N-IPQ)4,5,5
C
C        ERROR RETURN IN CASE OF FORMAL ERRORS
    4 IER=-1
      RETURN
C
C        INITIALIZE ITERATION PROCESS
    5 KOUNT=0
      IERV(2)=IP
      IERV(3)=IQ
      NDP=N+N+1
      NNE=NDP+NDP
      IX=IPQ-1
      IQP1=IQ+1
      IRHS=NNE+IPQ*IX/2
      IEND=IRHS+IX
C
C        TEST FOR AVAILABILITY OF AN INITIAL APPROXIMATION
      IF(IER)8,6,8
C
C        INITIALIZE NUMERATOR AND DENOMINATOR
    6 DO 7 I=2,IPQ
    7 P(I)=0.
      P(1)=1.
C
C        CALCULATE VALUES OF NUMERATOR AND DENOMINATOR FOR INITIAL
C           APPROXIMATION
    8 DO 9 J=1,N
      T=DATI(J)
      I=J+N
      CALL CNPS(WORK(I),T,P(IQP1),IP)
      K=I+N
    9 CALL CNPS(WORK(K),T,P,IQ)
C
C        SET UP NORMAL EQUATIONS (MAIN LOOP OF ITERATION)
   10 CALL APLL(FRAT,N,IX,WORK,WORK(IEND+1),DATI,IERV)
C
C        CHECK FOR ZERO DENOMINATOR
      IF(IERV(1))4,11,4
   11 INCR=0
      RELAX=2.
C
C        RESTORE MATRIX IN WORKING STORAGE
   12 J=IEND
      DO 13 I=NNE,IEND
      J=J+1
   13 WORK(I)=WORK(J)
      IF(KOUNT)14,14,15
C
C        SAVE SQUARE SUM OF ERRORS
   14 OSUM=WORK(IEND)
      DIAG=OSUM*EPS
      K=IQ
C
C        ADD CONSTANT TO DIAGONAL
      IF(WORK(NNE))17,17,19
   15 IF(INCR)19,19,16
   16 K=IPQ
   17 J=NNE-1
      DO 18 I=1,K
      WORK(J)=WORK(J)+DIAG
   18 J=J+I
C
C        SOLVE NORMAL EQUATIONS
   19 CALL APFS(WORK(NNE),IX,IPES,1,EPS,ETA,IER)
C
C        CHECK FOR FAILURE OF EQUATION SOLVER
```

```
      IF(IRES)4,4,20
C
C        TEST FOR DEFECTIVE NORMALEQUATIONS
   20 IF(IRES-IX)21,24,24
   21 IF(INCR)22,22,23
   22 DIAG=DIAG*0.125
   23 DIAG=DIAG+DIAG
      INCR=INCR+1
C
C        START WITH OVER RELAXATION
      RELAX=8.
      IF(INCR-LIMIT)12,45,45
C
C        CALCULATE VALUES OF CHANGE OF NUMERATOR AND DENOMINATOR
   24 L=NDP
      J=NNE+IRES*(IRES-1)/2-1
      K=J+IQ
      WORK(J)=0.
      IRQ=IQ
      IRP=IRES-IQ+1
      IF(IRP)25,26,26
   25 IRQ=IRES+1
   26 DO 29 I=1,N
      T=DATI(I)
      WORK(I)=0.
      CALL CNPS(WORK(I),T,WORK(K),IRP)
      M=L+N
      CALL CNPS(WORK(M),T,WORK(J),IRQ)
      IF(WORK(M)*WORK(L))27,29,29
   27 SUM=WORK(L)/WORK(M)
      IF(RELAX+SUM)29,29,28
   28 RELAX=-SUM
   29 L=L+1
C
C        MODIFY RELAXATION FACTOR IF NECESSARY
      SSOE=OSUM
      ITER=LIMIT
   30 SUM=0.
      RELAX=RELAX*0.5
      DO 32 I=1,N
      M=I+N
      K=M+N
      L=K+N
      SAVE=DATI(M)-(WORK(M)+RELAX*WORK(I))/(WORK(K)+RELAX*WORK(L))
      SAVE=SAVE*SAVE
      IF(DATI(NDP))32,32,31
   31 SAVE=SAVE*DATI(K)
   32 SUM=SUM+SAVE
      IF(ITER)45,33,33
   33 ITER=ITER-1
      IF(SUM-OSUM)34,37,35
   34 OSUM=SUM
      GOTO 30
C
C        TEST FOR IMPROVEMENT
   35 IF(OSUM-SSOE)36,30,30
   36 RELAX=RELAX+RELAX
   37 T=0.
      SAVE=0.
      K=IRES+1
      DO 38 I=2,K
      J=J+1
      T=T+ABS(P(I))
      P(I)=P(I)+RELAX*WORK(J)
   38 SAVE=SAVE+ABS(P(I))
C
C        UPDATE CURRENT VALUES OF NUMERATOR AND DENOMINATOR
      DO 39 I=1,N
      J=I+N
      K=J+N
      L=K+N
      WORK(J)=WORK(J)+RELAX*WORK(I)
   39 WORK(K)=WORK(K)+RELAX*WORK(L)
C
C        TEST FOR CONVERGENCE
      IF(INCR)40,40,42
   40 IF(SSOE-OSUM-RELAX*EPS*OSUM)46,46,41
   41 IF(ABS(T-SAVE)-RELAX*EPS*SAVE)46,46,42
   42 IF(OSUM-ETA*SAVE)46,46,43
   43 KOUNT=KOUNT+1
      IF(KOUNT-LIMIT)10,44,44
C
C        ERROR RETURN IN CASE OF POOR CONVERGENCE
   44 IER=2
      RETURN
   45 IER=1
      RETURN
C
C        NORMAL RETURN
   46 IER=0
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE FRAT
C
C        PURPOSE
C           FRAT IS USED FOR HANDLING OF DATA AND FUNDAMENTAL FUNCTIONS
C           WITH RATIONAL APPROXIMATION. IT IS A SUBSTANTIAL PART OF
C           RATIONAL APPROXIMATION AND HAS NO MEANING INDEPENDENTLY
C
C        USAGE
C           CALL FRAT(I,N,M,P,DATI,WGT,IER)
C
C        DESCRIPTION OF PARAMETERS
C           I      - SUBSCRIPT OF CURRENT DATA POINT
C           N      - NUMBER OF ALL DATA POINTS
C           M      - NUMBER OF FUNDAMENTAL FUNCTIONS USED
C           P      - ARRAY OF DIMENSION M+1 AT LEAST, WHICH CONTAINS
C                    ON RETURN THE VALUES OF THE M FUNDAMENTAL
C                    FUNCTIONS, FOLLOWED BY CURRENT FUNCTION VALUE
C           DATI   - ARRAY CONTAINING GIVEN N ARGUMENTS, FOLLOWED
C                    BY N FUNCTION VALUES AND FINALLY BY I RESPECTIVELY
C                    N WEIGHT VALUES
C           WGT    - RESULTANT WEIGHT FACTOR USED FOR I-TH TERM
C           IER    - RESULTANT ERROR PARAMETER, COMBINED WITH INPUT
C                    VALUES FOR CONTROL
C                    IER(2) MEANS DIMENSION OF NUMERATOR
C                    IER(3) MEANS DIMENSION OF DENOMINATOR
C                    IER(1) IS USED AS RESULTANT ERROR PARAMETER,
C                    IER(1) = 0 IN CASE OF NO ERRORS
C                    IER(1) = 1 OTHERWISE (ZERO DENOMINATOR)
C
C        REMARKS
C           VECTOR IER IS USED FOR COMMUNICATION BETWEEN FRAT AND FRAT
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           CNP
C
C        METHOD
C           CF. MATHEMATICAL DESCRIPTION OF SUBROUTINE FRAT
C
C     ..................................................................
C
      SUBROUTINE FRAT(I,N,M,P,DATI,WGT,IER)
C
C        DIMENSIONED DUMMY VARIABLES
      DIMENSION P(1),DATI(1),IER(1)
C
C        INITIALIZATION
      IP=IER(2)
      IQ=IER(3)
      IQM1=IQ-1
      IPQ=IP+IQ
C
C        LOOK UP ARGUMENT, FUNCTION VALUE AND WEIGHT
C        LOOK UP NUMERATOR AND DENOMINATOR
      T=DATI(I)
      J=I+N
      F=DATI(J)
      FNUM=P(J)
      J=J+N
      WGT=1.
      IF(DATI(2*N+1))2,2,1
    1 WGT=DATI(J)
    2 FDEN=P(J)
C
C        CALCULATE FUNCTION VALUE USED
      F=F*FDEN-FNUM
C
C        CHECK FOR ZERO DENOMINATOR
      IF(FDEN)4,3,4
C
C        ERROR RETURN IN CASE OF ZERO DENOMINATOR
    3 IER(1)=1
      RETURN
C
C        CALCULATE WEIGHT FACTORS USED
    4 WGT=WGT/(FDEN*FDEN)
      FNUM=-FNUM/FDEN
C
C        CALCULATE FUNDAMENTAL FUNCTIONS
      J=IQM1
      IF(IP-IQ)6,6,5
    5 J=IP-1
    6 CALL CNP(P(IQ),T,J)
C
C        STORE VALUES OF DENOMINATOR FUNDAMENTAL FUNCTIONS
    7 IF(IQM1)10,10,8
    8 DO 9 II=1,IQM1
      J=II+IQ
    9 P(II)=P(J)*FNUM
C
C        STORE FUNCTION VALUE
   10 P(IPQ)=F
C
C        NORMAL RETURN
      IER(1)=0
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DARAT
C
C        PURPOSE
C           CALCULATE BEST RATIONAL APPROXIMATION OF A DISCRETE
C           FUNCTION IN THE LEAST SQUARES SENSE
C
C        USAGE
C           CALL DARAT(DATI,N,WORK,P,IP,IQ,IER)
C
C        DESCRIPTION OF PARAMETERS
C           DATI  - TWODIMENSIONAL ARRAY WITH 3 COLUMNS AND N ROWS
C                   THE FIRST COLUMN MUST CONTAIN THE GIVEN ARGUMENTS,
C                   THE SECOND COLUMN THE GIVEN FUNCTION VALUES AND
C                   THE THIRD COLUMN THE GIVEN WEIGHTS IF ANY.
C                   IF NO WEIGHTS ARE TO BE USED THEN THE THIRD
C                   COLUMN MAY BE DROPPED , EXCEPT THE FIRST ELEMENT
C                   WHICH MUST CONTAIN A NONPOSITIVE VALUE
C                   DATI MUST BE OF DOUBLE PRECISION
C           N     - NUMBER OF NODES OF THE GIVEN DISCRETE FUNCTION
C           WORK  - WORKING STORAGE WHICH IS OF DIMENSION
C                   (IP+IQ)*(IP+IQ+1)+4*N+1 AT LEAST.
C                   ON RETURN THE VALUES OF THE NUMERATOR ARE CONTAINED
C                   IN WORK(N+1) UP TO WORK(2*N), WHILE THE VALUES OF
C                   THE DENOMINATOR ARE STORED IN WORK(2*N+1) UP TO
C                   WORK(3*N)
C                   WORK MUST BE OF DOUBLE PRECISION
C           P     - RESULTANT COEFFICIENT VECTOR OF DENOMINATOR AND
C                   NUMERATOR. THE DENOMINATOR IS STORED IN FIRST IQ
C                   LOCATIONS, THE NUMERATOR IN THE FOLLOWING IP
C                   LOCATIONS.
C                   COEFFICIENTS ARE ORDERED FROM LOW TO HIGH.
C                   P MUST BE OF DOUBLE PRECISION
C           IP    - DIMENSION OF THE NUMERATOR   (INPUT VALUE)
C           IQ    - DIMENSION OF THE DENOMINATOR (INPUT VALUE)
C           IER   - RESULTANT ERROR PARAMETER
C                   IER =-1 MEANS FORMAL ERRORS
C                   IER = 0 MEANS NO ERRORS
C                   IER = 1,2 MEANS POOR CONVERGENCE OF ITERATION
C                   IER IS ALSO USED AS INPUT VALUE
C                   A NONZERO INPUT VALUE INDICATES AVAILABILITY OF AN
C                   INITIAL APPROXIMATION STORED IN P
C
C        REMARKS
C           THE COEFFICIENT VECTORS OF THE DENOMINATOR AND NUMERATOR
C           OF THE RATIONAL APPROXIMATION ARE BOTH STORED IN P
C           STARTING WITH LOW POWERS (DENOMINATOR FIRST).
C           IP+IQ MUST NOT EXCEED N, ALL THREE VALUES MUST BE POSITIVE.
C           SINCE CHEBYSHEV POLYNOMIALS ARE USED AS FUNDAMENTAL
C           FUNCTIONS, THE ARGUMENTS SHOULD BE REDUCED TO THE INTERVAL
C           (-1,1). THIS CAN ALWAYS BE ACCOMPLISHED BY MEANS OF A LINEAR
C           TRANSFORMATION OF THE ORIGINALLY GIVEN ARGUMENTS.
C           IF A FIT IN OTHER FUNCTIONS IS REQUIRED, DCNP AND DCNPS MUST
C           BE REPLACED BY SUBROUTINES WHICH ARE OF ANALOGOUS DESIGN.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           DAPLL, DAPFS, DFRAT, DCNPS, DCNP
C           DCNP IS REQUIRED WITHIN DFRAT
C
C        METHOD
C           THE ITERATIVE SCHEME USED FOR CALCULATION OF THE
C           APPROXIMATION IS REPEATED SOLUTION OF THE NORMAL EQUATIONS
C           WHICH ARE OBTAINED BY LINEARIZATION.
C           A REFINED TECHNIQUE OF THIS LINEAR LEAST SQUARES APPROACH
C           IS USED WHICH GUARANTEES THAT THE DENOMINATOR IS FREE OF
C           ZEROES WITHIN THE APPROXIMATION INTERVAL.
C           FOR REFERENCE SEE
C           D.BRAESS, UEBER DAEMPFUNG BEI MINIMALISIERUNGSVERFAHREN,
C           COMPUTING(1966), VOL.1, ED.3, PP.264-272.
C           D.W.MARQUARDT, AN ALGORITHM FOR LEAST-SQUARES ESTIMATION
C           OF NONLINEAR PARAMETERS,
C           JSIAM(1963), VOL.11, ED.2, PP.431-441.
C
C     ..................................................................
C
      SUBROUTINE DARAT(DATI,N,WORK,P,IP,IQ,IER)
C
C
      EXTERNAL DFRAT
C
C        DIMENSIONED LOCAL VARIABLE
      DIMENSION IERV(3)
C
C        DIMENSIONED DUMMY VARIABLES
      DIMENSION DATI(1),WORK(1),P(1)
      DOUBLE PRECISION DATI,WORK,P,T,OSUM,DIAG,RELAX,SUM,SSOE,SAVE
C
C        INITIALIZE TESTVALUES
      LIMIT=20
      ETA=1.E-29
      EPS=1.E-14
C
C        CHECK FOR FORMAL ERRORS
      IF(N)4,4,1
    1 IF(IP)4,4,2
    2 IF(IQ)4,4,3
    3 IPQ=IP+IQ
      IF(N-IPQ)4,5,5
C
C        ERROR RETURN IN CASE OF FORMAL ERRORS
    4 IER=-1
      RETURN
C
C        INITIALIZE ITERATION PROCESS
    5 KOUNT=0
      IERV(2)=IP
      IERV(3)=IQ
      NDP=N+N+1
      NNE=NDP+NDP
      IX=IPQ-1
      IQP1=IQ+1
      IRHS=NNE+IPQ*IX/2
      IEND=IRHS+IX
C
C        TEST FOR AVAILABILITY OF AN INITIAL APPROXIMATION
      IF(IER)8,6,8
C
C        INITIALIZE NUMERATOR AND DENOMINATOR
    6 DO 7 I=2,IPQ
    7 P(I)=0.D0
      P(1)=1.D0
C
C        CALCULATE VALUES OF NUMERATOR AND DENOMINATOR FOR INITIAL
C        APPROXIMATION
    8 DO 9 J=1,N
      T=DATI(J)
      I=J+N
      CALL DCNPS(WORK(I),T,P(IQP1),IP)
      K=I+N
    9 CALL DCNPS(WORK(K),T,P,IQ)
C
C        SET UP NORMAL EQUATIONS (MAIN LOOP OF ITERATION)
   10 CALL DAPLL(DFRAT,N,IX,WORK,WORK(IEND+1),DATI,IERV)
C
C        CHECK FOR ZERO DENOMINATOR
      IF(IERV(1))4,11,4
   11 INCR=0
      RELAX=2.D0
C
C        RESTORE MATRIX IN WORKING STORAGE
   12 J=IEND
      DO 13 I=NNE,IEND
      J=J+1
   13 WORK(I)=WORK(J)
      IF(KOUNT)14,14,15
C
C        SAVE SQUARE SUM OF ERRORS
   14 OSUM=WORK(IEND)
      DIAG=OSUM*EPS
      K=IQ
C
C        ADD CONSTANT TO DIAGONAL
      IF(WORK(NNE))17,17,19
   15 IF(INCR)19,19,16
   16 K=IPQ
   17 J=NNE-1
      DO 18 I=1,K
      WORK(J)=WORK(J)+DIAG
   18 J=J+I
C
C        SOLVE NORMAL EQUATIONS
   19 CALL DAPFS(WORK(NNE),IX,IRES,1,EPS,ETA,IER)
C
C        CHECK FOR FAILURE OF EQUATION SOLVER
      IF(IRES)4,4,20
C
C        TEST FOR DEFECTIVE NORMALEQUATIONS
   20 IF(IRES-IX)21,24,24
   21 IF(INCR)22,22,23
   22 DIAG=DIAG*0.125D0
   23 DIAG=DIAG+DIAG
      INCR=INCR+1
C
C        START WITH OVER RELAXATION
      RELAX=8.D0
      IF(INCR-LIMIT)12,45,45
C
C        CALCULATE VALUES OF CHANGE OF NUMERATOR AND DENOMINATOR
   24 L=NDP
      J=NNE+IRES*(IRES-1)/2-1
      K=J+IQ
      WORK(J)=0.D0
      IRQ=IQ
      IRP=IRES-IQ+1
      IF(IRP)25,26,26
   25 IRQ=IRES+1
   26 DO 29 I=1,N
      T=DATI(I)
      WORK(I)=0.D0
      CALL DCNPS(WORK(I),T,WORK(K),IRP)
      M=L+N
      CALL DCNPS(WORK(M),T,WORK(J),IRQ)
      IF(WORK(M)*WORK(L))27,29,29
   27 SUM=WORK(L)/WORK(M)
      IF(RELAX+SUM)29,29,28
   28 RELAX=-SUM
   29 L=L+1
C
C        MODIFY RELAXATION FACTOR IF NECESSARY
      SSOE=OSUM
      ITER=LIMIT
   30 SUM=0.D0
      RELAX=RELAX*0.500
      DO 32 I=1,N
      M=I+N
      K=M+N
      L=K+N
      SAVE=DATI(M)-(WORK(M)+RELAX*WORK(I))/(WORK(K)+RELAX*WORK(L))
      SAVE=SAVE*SAVE
      IF(DATI(NDP))32,32,31
   31 SAVE=SAVE*DATI(K)
   32 SUM=SUM+SAVE
      IF(ITER)45,33,33
   33 ITER=ITER-1
      IF(SUM-OSUM)34,37,35
   34 OSUM=SUM
      GOTO 30
C
C        TEST FOR IMPROVEMENT
   35 IF(OSUM-SSOE)36,30,30
   36 RELAX=RELAX+RELAX
   37 T=0.
      SAVE=0.D0
      K=IRES+1
      DO 38 I=2,K
      J=J+1
      T=T+DABS(P(I))
      P(I)=P(I)+RELAX*WORK(J)
   38 SAVE=SAVE+DABS(P(I))
C
C        UPDATE CURRENT VALUES OF NUMERATOR AND DENOMINATOR
      DO 39 I=1,N
      J=I+N
      K=J+N
      L=K+N
      WORK(J)=WORK(J)+RELAX*WORK(I)
   39 WORK(K)=WORK(K)+RELAX*WORK(L)
C
C        TEST FOR CONVERGENCE
      IF(INCR)40,40,42
   40 IF(SSOE-OSUM-RELAX*OSUM*DBLE(EPS))46,46,41
   41 IF(DABS(T-SAVE)-RELAX*SAVE*DBLE(EPS))46,46,42
   42 IF(OSUM-SAVE*DBLE(ETA))46,46,43
   43 KOUNT=KOUNT+1
      IF(KOUNT-LIMIT)10,44,44
C
C        ERROR RETURN IN CASE OF POOR CONVERGENCE
   44 IER=2
      RETURN
   45 IER=1
      RETURN
C
C        NORMAL RETURN
   46 IER=0
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DFRAT
C
C        PURPOSE
C           DFRAT IS USED FOR HANDLING OF DATA AND FUNDAMENTAL FUNCTIONS
C           WITH RATIONAL APPROXIMATION. IT IS A SUBSTANTIAL PART OF
C           RATIONAL APPROXIMATION AND HAS NO MEANING INDEPENDENTLY
C
C        USAGE
C           CALL DFRAT(I,N,M,P,DATI,WGT,IER)
C
C        DESCRIPTION OF PARAMETERS
C           I      - SUBSCRIPT OF CURRENT DATA POINT
C           N      - NUMBER OF ALL DATA POINTS
C           M      - NUMBER OF FUNDAMENTAL FUNCTIONS USED
C           P      - ARRAY OF DIMENSION M+1 AT LEAST, WHICH CONTAINS
C                    ON RETURN THE VALUES OF THE M FUNDAMENTAL
C                    FUNCTIONS, FOLLOWED BY CURRENT FUNCTION VALUE
C                    P MUST BE OF DOUBLE PRECISION
C           DATI   - ARRAY CONTAINING GIVEN N ARGUMENTS, FOLLOWED
C                    BY N FUNCTION VALUES AND FINALLY BY 1 RESPECTIVELY
C                    N WEIGHT VALUES
C                    DATI MUST BE OF DOUBLE PRECISION
C           WGT    - RESULTANT WEIGHT FACTOR USED FOR I-TH TERM
C                    WGT MUST BE OF DOUBLE PRECISION
C           IER    - RESULTANT ERROR PARAMETER, COMBINED WITH INPUT
C                    VALUES FOR CONTROL
C                    IER(2) MEANS DIMENSION OF NUMERATOR
C                    IER(3) MEANS DIMENSION OF DENOMINATOR
C                    IER(1) IS USED AS RESULTANT ERROR PARAMETER,
C                    IER(1) = 0 IN CASE OF NO ERRORS
C                    IER(1) = 1 OTHERWISE (ZERO DENOMINATOR)
C
C        REMARKS
C           VECTOR IER IS USED FOR COMMUNICATION BETWEEN DFRAT AND DFRAT
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           DCNP
C
C        METHOD
C           CF. MATHEMATICAL DESCRIPTION OF SUBROUTINE ARAT
C
C     ..................................................................
C
      SUBROUTINE DFRAT(I,N,M,P,DATI,WGT,IER)
C
C
C        DIMENSIONED DUMMY VARIABLES
      DIMENSION P(1),DATI(1),IER(1)
      DOUBLE PRECISION P,DATI,WGT,T,F,FNUM,FDEN
C
C        INITIALIZATION
      IP=IER(2)
      IQ=IER(3)
      IQM1=IQ-1
      IPQ=IP+IQ
C
C        LOOK UP ARGUMENT, FUNCTION VALUE AND WEIGHT
C        LOOK UP NUMERATOR AND DENOMINATOR
      T=DATI(I)
      J=I+N
      F=DATI(J)
      FNUM=P(J)
      J=J+N
      WGT=1.D0
      IF(DATI(2*N+1))2,2,1
    1 WGT=DATI(J)
    2 FDEN=P(J)
C
C        CALCULATE FUNCTION VALUE USED
      F=F*FDEN-FNUM
C
C        CHECK FOR ZERO DENOMINATOR
      IF(FDEN)4,3,4
C
C        ERROR RETURN IN CASE OF ZERO DENOMINATOR
    3 IER(1)=1
      RETURN
C
C        CALCULATE WEIGHT FACTORS USED
    4 WGT=WGT/(FDEN*FDEN)
      FNUM=-FNUM/FDEN
C
C        CALCULATE FUNDAMENTAL FUNCTIONS
      J=IQM1
      IF(IP-IQ)6,6,5
    5 J=IP-1
    6 CALL DCNP(P(IQ),T,J)
C
C        STORE VALUES OF DENOMINATOR FUNDAMENTAL FUNCTIONS
    7 IF(IQM1)10,10,8
    8 DO 9 II=1,IQM1
      J=II+IQ
    9 P(II)=P(J)*FNUM
C
C        STORE FUNCTION VALUE
   10 P(IPQ)=F
C
C        NORMAL RETURN
      IER(1)=0
      RETURN
      END
```

Subroutines APLL and DAPLL

These subroutines set up the normal equations for a linear least-squares fit to a given discrete function. APLL is a preparatory step for the calculation of the least-squares fit. Normally it is followed by subroutine APFS. P → 260

A user-coded external subroutine must be supplied to compute the values of the fundamental functions.

1. Mathematical background

Let $f(x)$, $g_i(x)$, $i = 1, 2, \ldots, m$, and $w(x) > 0$ be functions defined for $x = x_1, x_2, \ldots, x_n$ (the x_i may be vectors as well as scalars). The problem is to determine the coefficients c_i of the linear combination

$$p(x) = \sum_{i=1}^{m} c_i g_i(x)$$

such that

$$\sum_{k=1}^{n} w(x_k)\,(f(x_k) - p(x_k))^2 = \text{minimum}$$

This problem leads to a system of linear equations $AC = R$, where C is the vector of unknown coefficients, A is the m by m symmetric positive definite matrix with elements

$$a_{jk} = \sum_{i=1}^{n} w(x_i)\, g_j(x_i)\, g_k(x_i)$$

and R is an m-dimensional vector with elements

$$r_j = \sum_{i=1}^{n} w(x_i)\, f(x_i)\, g_j(x_i)$$

(See APFS for details.)

Some remarks regarding polynomial approximation are in order. Use of monomials $g_i(x) = x^{i-1}$ as fundamental functions results in a badly ill-conditioned coefficient matrix A. If Chebyshev or Legendre polynomials are used instead, the condition of the normal equations is improved remarkably, provided the arguments have a sensible distribution (for example, are equidistant in the interval -1 to +1).

2. Programming considerations

To allow for full flexibility in data handling, the user must provide an external subroutine that returns the values $g_1(x_i)$, $g_2(x_i), \ldots, g_m(x_i)$, $f(x_i)$ stored in a vector P together with the weight value $w(x_i)$. Input values to this subroutine are the current subscript value i, a number of points n, and -- for ease of communication between main line and external -- a dummy array named DATI.

Coefficient matrix A and right-hand side R are allocated adjacently. A is stored columnwise in compressed form, and R is augmented by one element ff in which the weighted square sum of function values is returned. The coefficient matrix A, right-hand side R, and square sum of function values ff are generated in a loop over i from 1 to n. Initially they are set to zero.

Storage allocation scheme:

$$\begin{pmatrix} a_{11} a_{12} \cdots a_{1m} r_1 \\ a_{22} \cdots a_{2m} r_2 \\ \vdots \\ a_{mm} r_m \\ ff \end{pmatrix} \quad \begin{pmatrix} g_1(x_i) \\ g_2(x_i) \\ \vdots \\ g_m(x_i) \\ f(x_i) \end{pmatrix} \quad w(x_i)$$

Triangular array WORK of dimension m+1 by m+1 — Vector P of dimension m+1 — Weight WGT

For updating of the jth column of the triangular array WORK, the jth element of vector P is multiplied by the weight WGT, giving AUX and then P(i). AUX is added to the ith element in the jth column of WORK for i equal 1 up to j.

In this way the coefficient matrix A, the right-hand side R, and the square sum of function values are generated quite naturally by one and the same scheme.

a. External subroutine for data handling

The error parameter IER may be declared as scalar (that is, vector with one element) or vector in the user's main line. Within APLL it is treated as a vector with one element.

IER(1) is used for control purposes. If IER(1) is set to a nonzero value within the external subroutine, APLL returns to the calling program (main line), with an unchanged value of IER(1). APLL sets the error parameter IER(1) to zero at the beginning. An error return with IER(1) = -1 is activated if n and m are not both positive or if n is less than m.

The user has full flexibility for handling of the data x_i, $f(x_i)$, $w(x_i)$, $g_1(x_i), \ldots, g_m(x_i)$.

(1) If he wishes to allocate x_i, $f(x_i)$, $w(x_i)$, $g_1(x_i), \ldots, g_m(x_i)$ in main storage, he may use an array the name of which replaces the dummy DATI.

Integer values needed for addressing may be inserted as additional components in the vector IER.

(2) Use of COMMON is another possibility for data exchange between the main line and the data handling external subroutine.

(3) Calculation of some or all of the required quantities as functions of the subscript or as functions of the argument x_i is another convenient choice.

(4) The needed data may be read in sequentially from one or more external devices, for example, x_i, $f(x_i)$, $w(x_i)$ from card unit, and/or $g_1(x_i), \ldots, g_m(x_i)$ from tape unit.

The four cases listed above may occur in any sensible combination.

```
C                                                                       APLL  10
C     ..................................................................APLL  20
C                                                                       APLL  30
C        SUBROUTINE APLL                                                APLL  40
C                                                                       APLL  50
C        PURPOSE                                                        APLL  60
C           SET UP NORMAL EQUATIONS FOR A LINEAR LEAST SQUARES FIT      APLL  70
C           TO A GIVEN DISCRETE FUNCTION                                APLL  80
C                                                                       APLL  90
C        USAGE                                                          APLL 100
C           CALL APLL(FFCT,N,IP,P,WORK,DATI,IER)                        APLL 110
C           SUBROUTINE FFCT REQUIRES AN EXTERNAL STATEMENT              APLL 120
C                                                                       APLL 130
C        DESCRIPTION OF PARAMETERS                                      APLL 140
C           FFCT  - USER CODED SUBROUTINE WHICH MUST BE DECLARED        APLL 150
C                   EXTERNAL IN THE MAIN PROGRAM. IT IS CALLED          APLL 160
C                   CALL FFCT(I,N,IP,P,DATI,WGT,IER) AND RETURNS        APLL 170
C                   THE VALUES OF THE FUNDAMENTAL FUNCTIONS FOR         APLL 180
C                   THE I-TH ARGUMENT IN P(1) UP TO P(IP)               APLL 190
C                   FOLLOWED BY THE I-TH FUNCTION VALUE IN P(IP+1)      APLL 200
C                   N IS THE NUMBER OF ALL POINTS                       APLL 210
C                   DATI IS A DUMMY PARAMETER WHICH IS USED AS ARRAY    APLL 220
C                   NAME. THE GIVEN DATA SET MAY BE ALLOCATED IN DATI   APLL 230
C                   WGT IS THE WEIGHT FACTOR FOR THE I-TH POINT         APLL 240
C                   IER IS USED AS RESULTANT ERROR PARAMETER IN FFCT    APLL 250
C           N     - NUMBER OF GIVEN POINTS                              APLL 260
C           IP    - NUMBER OF FUNDAMENTAL FUNCTIONS USED FOR LEAST      APLL 270
C                   SQUARES FIT                                         APLL 280
C                   IP SHOULD NOT EXCEED N                              APLL 290
C           P     - WORKING STORAGE OF DIMENSION IP+1, WHICH            APLL 300
C                   IS USED AS INTERFACE BETWEEN APLL AND THE USER      APLL 310
C                   CODED SUBROUTINE FFCT                               APLL 320
C           WORK  - WORKING STORAGE OF DIMENSION (IP+1)*(IP+2)/2.       APLL 330
C                   ON RETURN WORK CONTAINS THE SYMMETRIC COEFFICIENT   APLL 340
C                   MATRIX OF THE NORMAL EQUATIONS IN COMPRESSED FORM,  APLL 350
C                   I.E. UPPER TRINGULAR PART ONLY STORED COLUMNWISE.   APLL 360
C                   THE FOLLOWING IP POSITIONS CONTAIN THE RIGHT        APLL 370
C                   HAND SIDE AND WORK((IP+1)*(IP+2)/2) CONTAINS        APLL 380
C                   THE WEIGHTED SQUARE SUM OF THE FUNCTION VALUES      APLL 390
C           DATI  - DUMMY ENTRY TO COMMUNICATE AN ARRAY NAME BETWEEN    APLL 400
C                   MAIN LINE AND SUBROUTINE FFCT.                      APLL 410
C           IER   - RESULTING ERROR PARAMETER                           APLL 420
C                   IER =-1 MEANS FORMAL ERRORS IN SPECIFIED DIMENSIONS APLL 430
C                   IER = 0 MEANS NO ERRORS                             APLL 440
C                   IER = 1 MEANS ERROR IN EXTERNAL SUBROUTINE FFCT     APLL 450
C                                                                       APLL 460
C        REMARKS                                                        APLL 470
C           TO ALLOW FOR EASY COMMUNICATION OF INTEGER VALUES           APLL 480
C           BETWEEN MAINLINE AND EXTERNAL SUBROUTINE FFCT, THE ERROR    APLL 490
C           PARAMETER IER IS TREATED AS A VECTOR OF DIMENSION 1 WITHIN  APLL 500
C           SUBROUTINE APLL. ADDITIONAL COMPONENTS OF IER MAY BE        APLL 510
C           INTRODUCED BY THE USER FOR COMMUNICATION BACK AND FORTH.    APLL 520
C           IN THIS CASE, HOWEVER, THE USER MUST SPECIFY IER AS A       APLL 530
C           VECTOR IN HIS MAINLINE.                                     APLL 540
C           EXECUTION OF SUBROUTINE APLL IS A PREPARATORY STEP FOR      APLL 550
C           CALCULATION OF THE LINEAR LEAST SQUARES FIT.                APLL 560
C           NORMALLY IT IS FOLLOWED BY EXECUTION OF SUBROUTINE APFS     APLL 570
C                                                                       APLL 580
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  APLL 590
C           THE EXTERNAL SUBROUTINE FFCT MUST BE FURNISHED BY THE USER  APLL 600
C                                                                       APLL 610
C        METHOD                                                         APLL 620
C           HANDLING OF THE GIVEN DATA SET (ARGUMENTS,FUNCTION VALUES   APLL 630
```

```
C          AND WEIGHTS) IS COMPLETELY LEFT TO THE USER                  APLL 640
C          ESSENTIALLY HE HAS THREE CHOICES                             APLL 650
C          (1) THE I-TH VALUES OF ARGUMENT, FUNCTION VALUE AND WEIGHT   APLL 660
C              ARE CALCULATED WITHIN SUBROUTINE FFCT FOR GIVEN I.       APLL 670
C          (2) THE I-TH VALUES OF ARGUMENT, FUNCTION VALUE AND WEIGHT   APLL 680
C              ARE DETERMINED BY TABLE LOOK UP. THE STORAGE LOCATIONS   APLL 690
C              REQUIRED ARE ALLOCATED WITHIN THE DUMMY ARRAY DATI       APLL 700
C              (POSSIBLY IN P TOO, IN EXCESS OF THE SPECIFIED IP + 1    APLL 710
C              LOCATIONS).                                              APLL 720
C              ANOTHER POSSIBILITY WOULD BE TO USE COMMON AS INTERFACE  APLL 730
C              BETWEEN MAIN LINE AND SUBROUTINE FFCT AND TO ALLOCATE    APLL 740
C              STORAGE FOR THE DATA SET IN COMMON.                      APLL 750
C          (3) THE I-TH VALUES OF ARGUMENT, FUNCTION VALUE AND WEIGHT   APLL 760
C              ARE READ IN FROM AN EXTERNAL DEVICE. THIS MAY BE EASILY  APLL 770
C              ACCOMPLISHED SINCE I IS USED STRICTLY INCREASING FROM    APLL 780
C              ONE UP TO N WITHIN APLL                                  APLL 790
C                                                                       APLL 800
C   ...................................................................APLL 810
C                                                                       APLL 820
      SUBROUTINE APLL(FFCT,N,IP,P,WORK,DATI,IER)                        APLL 830
C                                                                       APLL 840
C                                                                       APLL 850
C        DIMENSIONED DUMMY VARIABLES                                    APLL 860
      DIMENSION P(1),WORK(1),DATI(1),IER(1)                             APLL 870
C                                                                       APLL 880
C        CHECK FOR FORMAL ERRORS IN SPECIFIED DIMENSIONS                APLL 890
      IF(N)10,10,1                                                      APLL 900
    1 IF(IP)10,10,2                                                     APLL 910
    2 IF(N-IP)10,3,3                                                    APLL 920
C                                                                       APLL 930
C        SET WORKING STORAGE AND RIGHT HAND SIDE TO ZERO                APLL 940
    3 IPP1=IP+1                                                         APLL 950
      M=IPP1*(IP+2)/2                                                   APLL 960
      IER(1)=0                                                          APLL 970
      DO 4 I=1,M                                                        APLL 980
    4 WORK(I)=0.                                                        APLL 990
C                                                                       APLL1000
C        START GREAT LOOP OVER ALL GIVEN POINTS                         APLL1010
      DO 8 I=1,N                                                        APLL1020
      CALL FFCT(I,N,IP,P,DATI,WGT,IER)                                  APLL1030
      IF(IER(1))9,5,9                                                   APLL1040
    5 J=0                                                               APLL1050
      DO 7 K=1,IPP1                                                     APLL1060
      AUX=P(K)*WGT                                                      APLL1070
      DO 6 L=1,K                                                        APLL1080
      J=J+1                                                             APLL1090
    6 WORK(J)=WORK(J)+P(L)*AUX                                          APLL1100
    7 CONTINUE                                                          APLL1110
    8 CONTINUE                                                          APLL1120
C                                                                       APLL1130
C        NORMAL RETURN                                                  APLL1140
    9 RETURN                                                            APLL1150
C                                                                       APLL1160
C        ERROR RETURN IN CASE OF FORMAL ERRORS                          APLL1170
   10 IER(1)=-1                                                         APLL1180
      RETURN                                                            APLL1190
      END                                                               APLL1200
```

```
C                                                                       DAPL 10
C   ...................................................................DAPL 20
C                                                                       DAPL 30
C        SUBROUTINE DAPLL                                               DAPL 40
C        PURPOSE                                                        DAPL 50
C           SET UP NORMAL EQUATIONS FOR A LINEAR LEAST SQUARES FIT      DAPL 60
C           TO A GIVEN DISCRETE FUNCTION                                DAPL 70
C                                                                       DAPL 80
C        USAGE                                                          DAPL 90
C           CALL DAPLL(FFCT,N,IP,P,WORK,DATI,IER)                       DAPL 100
C           SUBROUTINE FFCT REQUIRES AN EXTERNAL STATEMENT              DAPL 110
C                                                                       DAPL 120
C        DESCRIPTION OF PARAMETERS                                      DAPL 130
C           FFCT  - USER CODED SUBROUTINE WHICH MUST BE DECLARED        DAPL 140
C                   EXTERNAL IN THE MAIN PROGRAM. IT IS CALLED          DAPL 150
C                   CALL FFCT(I,N,IP,P,DATI,WGT,IER) AND RETURNS        DAPL 160
C                   THE VALUES OF THE FUNDAMENTAL FUNCTIONS FOR         DAPL 170
C                   THE I-TH ARGUMENT IN P(1) UP TO P(IP)               DAPL 180
C                   FOLLOWED BY THE I-TH FUNCTION VALUE IN P(IP+1)      DAPL 190
C                   N IS THE NUMBER OF ALL POINTS                       DAPL 200
C                   P,DATI,WGT MUST BE OF DOUBLE PRECISION.             DAPL 210
C                   DATI IS A DUMMY PARAMETER WHICH IS USED AS ARRAY    DAPL 220
C                   NAME. THE GIVEN DATA SET MAY BE ALLOCATED IN DATI   DAPL 230
C                   WGT IS THE WEIGHT FACTOR FOR THE I-TH POINT         DAPL 240
C                   IER IS USED AS RESULTANT ERROR PARAMETER IN FFCT    DAPL 250
C           N     - NUMBER OF GIVEN POINTS                              DAPL 260
C           IP    - NUMBER OF FUNDAMENTAL FUNCTIONS USED FOR LEAST      DAPL 270
C                   SQUARES FIT                                         DAPL 280
C                   IP SHOULD NOT EXCEED N                              DAPL 290
C           P     - WORKING STORAGE OF DIMENSION IP+1, WHICH            DAPL 300
C                   IS USED AS INTERFACE BETWEEN APLL AND THE USER      DAPL 310
C                   CODED SUBROUTINE FFCT                               DAPL 320
C                   P MUST BE OF DOUBLE PRECISION.                      DAPL 330
C           WORK  - WORKING STORAGE OF DIMENSION (IP+1)*(IP+2)/2.       DAPL 340
C                   ON RETURN WORK CONTAINS THE SYMMETRIC COEFFICIENT   DAPL 350
C                   MATRIX OF THE NORMAL EQUATIONS IN COMPRESSED FORM,  DAPL 360
C                   I.E. UPPER TRINGULAR PART ONLY STORED COLUMNWISE.   DAPL 370
C                   THE FOLLOWING IP POSITIONS CONTAIN THE RIGHT        DAPL 380
C                   HAND SIDE AND WORK((IP+1)*(IP+2)/2) CONTAINS        DAPL 390
C                   THE WEIGHTED SQUARE SUM OF THE FUNCTION VALUES      DAPL 400
C                   WORK MUST BE OF DOUBLE PRECISION.                   DAPL 410
C           DATI  - DUMMY ENTRY TO COMMUNICATE AN ARRAY NAME BETWEEN    DAPL 420
C                   MAIN LINE AND SUBROUTINE FFCT.                      DAPL 430
C                   DATI MUST BE OF DOUBLE PRECISION.                   DAPL 440
C           IER   - RESULTING ERROR PARAMETER                           DAPL 450
C                   IER =-1 MEANS FORMAL ERRORS IN SPECIFIED DIMENSIONS DAPL 460
C                   IER = 0 MEANS NO ERRORS                             DAPL 470
C                   IER = 1 MEANS ERROR IN EXTERNAL SUBROUTINE FFCT     DAPL 480
C                                                                       DAPL 490
C        REMARKS                                                        DAPL 500
C           TO ALLOW FOR EASY COMMUNICATION OF INTEGER VALUES           DAPL 510
C           BETWEEN MAINLINE AND EXTERNAL SUBROUTINE FFCT. THE ERROR    DAPL 520
C           PARAMETER IER IS TREATED AS A VECTOR OF DIMENSION 1 WITHIN  DAPL 530
C           SUBROUTINE DAPLL. ADDITIONAL COMPONENTS OF IER MAY BE       DAPL 540
C           INTRODUCED BY THE USER FOR COMMUNICATION BACK AND FORTH.    DAPL 550
C           IN THIS CASE, HOWEVER, THE USER MUST SPECIFY IER AS A       DAPL 560
C           VECTOR IN HIS MAINLINE.                                     DAPL 570
C           EXECUTION OF SUBROUTINE DAPLL IS A PREPARATORY STEP FOR     DAPL 580
C           CALCULATION OF THE LINEAR LEAST SQUARES FIT.                DAPL 590
C           NORMALLY IT IS FOLLOWED BY EXECUTION OF SUBROUTINE DAPFS    DAPL 600
C                                                                       DAPL 610
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DAPL 620
C           THE EXTERNAL SUBROUTINE FFCT MUST BE FURNISHED BY THE USER  DAPL 630
C                                                                       DAPL 640
C        METHOD                                                         DAPL 650
C           HANDLING OF THE GIVEN DATA SET (ARGUMENTS,FUNCTION VALUES   DAPL 660
C           AND WEIGHTS) IS COMPLETELY LEFT TO THE USER                 DAPL 670
C           ESSENTIALLY HE HAS THREE CHOICES                            DAPL 680
C           (1) THE I-TH VALUES OF ARGUMENT, FUNCTION VALUE AND WEIGHT  DAPL 690
C               ARE CALCULATED WITHIN SUBROUTINE FFCT FOR GIVEN I.      DAPL 700
C           (2) THE I-TH VALUES OF ARGUMENT, FUNCTION VALUE AND WEIGHT  DAPL 710
C               ARE DETERMINED BY TABLE LOOK UP. THE STORAGE LOCATIONS  DAPL 720
C               REQUIRED ARE ALLOCATED WITHIN THE DUMMY ARRAY DATI      DAPL 730
C               (POSSIBLY IN P TOO, IN EXCESS OF THE SPECIFIED IP + 1   DAPL 740
C               LOCATIONS).                                             DAPL 750
C               ANOTHER POSSIBILITY WOULD BE TO USE COMMON AS INTERFACE DAPL 760
C               BETWEEN MAIN LINE AND SUBROUTINE FFCT AND TO ALLOCATE   DAPL 770
C               STORAGE FOR THE DATA SET IN COMMON.                     DAPL 780
C           (3) THE I-TH VALUES OF ARGUMENT, FUNCTION VALUE AND WEIGHT  DAPL 790
C               ARE READ IN FROM AN EXTERNAL DEVICE. THIS MAY BE EASILY DAPL 800
C               ACCOMPLISHED SINCE I IS USED STRICTLY INCREASING FROM   DAPL 810
C               ONE UP TO N WITHIN APLL                                 DAPL 820
C                                                                       DAPL 830
C   ...................................................................DAPL 840
C                                                                       DAPL 850
      SUBROUTINE DAPLL(FFCT,N,IP,P,WORK,DATI,IER)                       DAPL 860
C                                                                       DAPL 870
C                                                                       DAPL 880
C        DIMENSIONED DUMMY VARIABLES                                    DAPL 890
      DIMENSION P(1),WORK(1),DATI(1),IER(1)                             DAPL 900
      DOUBLE PRECISION P,WORK,DATI,WGT,AUX                              DAPL 910
C                                                                       DAPL 920
C        CHECK FOR FORMAL ERRORS IN SPECIFIED DIMENSIONS                DAPL 930
      IF(N)10,10,1                                                      DAPL 940
    1 IF(IP)10,10,2                                                     DAPL 950
    2 IF(N-IP)10,3,3                                                    DAPL 960
C                                                                       DAPL 970
C        SET WORKING STORAGE AND RIGHT HAND SIDE TO ZERO                DAPL 980
    3 IPP1=IP+1                                                         DAPL 990
      M=IPP1*(IP+2)/2                                                   DAPL1000
      IER(1)=0                                                          DAPL1010
      DO 4 I=1,M                                                        DAPL1020
    4 WORK(I)=0.D0                                                      DAPL1030
C                                                                       DAPL1040
C        START GREAT LOOP OVER ALL GIVEN POINTS                         DAPL1050
      DO 8 I=1,N                                                        DAPL1060
      CALL FFCT(I,N,IP,P,DATI,WGT,IER)                                  DAPL1070
      IF(IER(1))9,5,9                                                   DAPL1080
    5 J=0                                                               DAPL1090
      DO 7 K=1,IPP1                                                     DAPL1100
      AUX=P(K)*WGT                                                      DAPL1110
      DO 6 L=1,K                                                        DAPL1120
      J=J+1                                                             DAPL1130
    6 WORK(J)=WORK(J)+P(L)*AUX                                          DAPL1140
    7 CONTINUE                                                          DAPL1150
    8 CONTINUE                                                          DAPL1160
C                                                                       DAPL1170
C        NORMAL RETURN                                                  DAPL1180
    9 RETURN                                                            DAPL1190
C                                                                       DAPL1200
C        ERROR RETURN IN CASE OF FORMAL ERRORS                          DAPL1210
   10 IER(1)=-1                                                         DAPL1220
      RETURN                                                            DAPL1230
      END                                                               DAPL1240
```

Subroutine FORIF

This subroutine produces the Fourier coefficients for a given periodic function.

Given:

1. A function f(x) for values of x between 0 and 2π

2. N - the spacing desired such that the interval is $2\pi/(2N + 1)$

3. M - the desired order of the Fourier coefficients, $0 \leq M \leq N$

the coefficients of the Fourier series that approximate the given function are calculated as follows:

$$C_1 = \cos\left(\frac{2\pi}{2N+1}\right) \tag{1}$$

$$S_1 = \sin\left(\frac{2\pi}{2N+1}\right) \tag{2}$$

$$U_2 = 0$$

$$U_1 = 0$$

$$C = 1$$

$$S = 0$$

$$J = 1$$

The following recursive sequence is used to compute U_0, U_1, and U_2:

$$U_0 = f\left(\frac{2m\pi}{2N+1}\right) + 2\,C\,U_1 - U_2 \tag{3}$$

$$U_2 = U_1$$

$$U_1 = U_0$$

for values of $m = 2N, 2N-1, \ldots, 1$

The coefficients are then:

$$A_J = \frac{2}{2N+1}\left(f(0) + C\,U_1 - U_2\right) \tag{4}$$

$$B_J = \frac{2}{2N+1}\,S\,U_1 \tag{5}$$

The values of C and S are updated to:

$$Q = C_1\,C - S_1\,S$$

$$S = C_1\,S + S_1\,C$$

$$C = Q$$

J is stepped by 1 and the sequence starting at equation (3) is now repeated until M+1 pairs of coefficients have been computed.

```
C                                                                       FRIF  10
C     ...................................................................FRIF  20
C                                                                       FRIF  30
C        SUBROUTINE FORIF                                               FRIF  40
C                                                                       FRIF  50
C        PURPOSE                                                        FRIF  60
C           FOURIER ANALYSIS OF A GIVEN PERIODIC FUNCTION IN THE        FRIF  70
C           RANGE 0-2PI                                                 FRIF  80
C           COMPUTES THE COEFFICIENTS OF THE DESIRED NUMBER OF TERMS    FRIF  90
C           IN THE FOURIER SERIES F(X)=A(0)+SUM(A(K)COS KX+B(K)SIN KX)  FRIF 100
C           WHERE K=1,2,....,M TO APPROXIMATE THE COMPUTED VALUES OF A  FRIF 110
C           GIVEN FUNCTION SUBPROGRAM                                   FRIF 120
C                                                                       FRIF 130
C        USAGE                                                          FRIF 140
C           CALL FORIF(FUN,N,M,A,B,IER)                                 FRIF 150
C                                                                       FRIF 160
C        DESCRIPTION OF PARAMETERS                                      FRIF 170
C           FUN-NAME OF FUNCTION SUBPROGRAM TO BE USED FOR COMPUTING    FRIF 180
C               DATA POINTS                                             FRIF 190
C           N  -DEFINES THE INTERVAL SUCH THAT 2N+1 POINTS ARE TAKEN    FRIF 200
C               OVER THE INTERVAL (0,2PI). THE SPACING IS THUS 2PI/2N+1 FRIF 210
C           M  -THE MAXIMUM ORDER OF THE HARMONICS TO BE FITTED         FRIF 220
C           A  -RESULTANT VECTOR OF FOURIER COSINE COEFFICIENTS OF      FRIF 230
C               LENGTH M+1                                              FRIF 240
C               A SUB 0, A SUB 1,..., A SUB M                           FRIF 250
C           B  -RESULTANT VECTOR OF FOURIER SINE COEFFICIENTS OF        FRIF 260
C               LENGTH M+1                                              FRIF 270
C               B SUB 0, B SUB 1,..., B SUB M                           FRIF 280
C           IER-RESULTANT ERROR CODE WHERE                              FRIF 290
C               IER=0  NO ERROR                                         FRIF 300
C               IER=1  N NOT GREATER OR EQUAL TO M                      FRIF 310
C               IER=2  M LESS THAN 0                                    FRIF 320
C                                                                       FRIF 330
C        REMARKS                                                        FRIF 340
C           M MUST BE GREATER THAN OR EQUAL TO ZERO                     FRIF 350
C           N MUST BE GREATER THAN OR EQUAL TO M                        FRIF 360
C           THE FIRST ELEMENT IN VECTOR B IS ZERO IN ALL CASES          FRIF 370
C                                                                       FRIF 380
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  FRIF 390
C           FUN-NAME OF USER FUNCTION SUBPROGRAM USED FOR COMPUTING     FRIF 400
C               DATA POINTS                                             FRIF 410
C           CALLING PROGRAM MUST HAVE FORTRAN EXTERNAL STATEMENT        FRIF 420
C           CONTAINING NAMES OF FUNCTION SUBPROGRAMS LISTED IN CALL TO  FRIF 430
C           FORIF                                                       FRIF 440
C                                                                       FRIF 450
C        METHOD                                                         FRIF 460
C           USES RECURSIVE TECHNIQUE DESCRIBED IN A. RALSTON, H. WILF,  FRIF 470
C           'MATHEMATICAL METHODS FOR DIGITAL COMPUTERS', JOHN WILEY    FRIF 480
C           AND SONS, NEW YORK, 1965, CHAPTER 24. THE METHOD OF         FRIF 490
C           INDEXING THROUGH THE PROCEDURE HAS BEEN MODIFIED TO         FRIF 500
C           SIMPLIFY THE COMPUTATION.                                   FRIF 510
C                                                                       FRIF 520
C     ...................................................................FRIF 530
C                                                                       FRIF 540
      SUBROUTINE FORIF(FUN,N,M,A,B,IER)                                 FRIF 550
      DIMENSION A(1),B(1)                                               FRIF 560
C                                                                       FRIF 570
C        CHECK FOR PARAMETER ERRORS                                     FRIF 580
C                                                                       FRIF 590
      IER=0                                                             FRIF 600
   20 IF(M) 30,40,40                                                    FRIF 610
   30 IER=2                                                             FRIF 620
      RETURN                                                            FRIF 630
   40 IF(M-N) 60,60,50                                                  FRIF 640
   50 IER=1                                                             FRIF 650
      RETURN                                                            FRIF 660
C                                                                       FRIF 670
C        COMPUTE AND PRESET CONSTANTS                                   FRIF 680
C                                                                       FRIF 690
   60 AN=N                                                              FRIF 700
      COEF=2.0/(2.0*AN+1.0)                                             FRIF 710
      CONST=3.141593*COEF                                               FRIF 720
      S1=SIN(CONST)                                                     FRIF 730
      C1=COS(CONST)                                                     FRIF 740
      C=1.0                                                             FRIF 750
      S=0.0                                                             FRIF 760
      J=1                                                               FRIF 770
      FUNZ=FUN(0.0)                                                     FRIF 780
   70 U2=0.0                                                            FRIF 790
      U1=0.0                                                            FRIF 800
      AI=2*N                                                            FRIF 810
C                                                                       FRIF 820
C        FORM FOURIER COEFFICIENTS RECURSIVELY                          FRIF 830
C                                                                       FRIF 840
   75 X=AI*CONST                                                        FRIF 850
      U0=FUN(X)+2.0*C*U1-U2                                             FRIF 860
      U2=U1                                                             FRIF 870
      U1=U0                                                             FRIF 880
      AI=AI-1.0                                                         FRIF 890
      IF(AI) 80,80,75                                                   FRIF 900
   80 A(J)=COEF*(FUNZ+C*U1-U2)                                          FRIF 910
      B(J)=COEF*S*U1                                                    FRIF 920
      IF(J-(M+1)) 90,100,100                                            FRIF 930
   90 Q=C1*C-S1*S                                                       FRIF 940
      S=C1*S+S1*C                                                       FRIF 950
      C=Q                                                               FRIF 960
      J=J+1                                                             FRIF 970
      GO TO 70                                                          FRIF 980
  100 A(1)=A(1)*0.5                                                     FRIF 990
      RETURN                                                            FRIF1000
      END                                                               FRIF1010
```

Subroutine FORIT

This subroutine produces the Fourier coefficients of a tabulated function.

Given:

1. Tabulated values of a function f(x) for x between 0 and 2π in steps of $2\pi/(2N+1)$

2. N such that there are 2N+1 tabulated data points: $2K\pi/(2N+1)$, K = 0, 1, 2, . . . , 2N

3. M - the desired order of the Fourier coefficients where $0 \leq M \leq N$

the coefficients of the Fourier series which approximate the given function are calculated as follows:

$$C_1 = \cos\left(\frac{2\pi}{2N+1}\right) \tag{1}$$

$$S_1 = \sin\left(\frac{2\pi}{2N+1}\right) \tag{2}$$

$$U_2 = 0$$

$$U_1 = 0$$

$$C = 1$$

$$S = 0$$

$$J = 1$$

The following recursive sequence is used to compute U_0, U_1, and U_2

$$U_0 = f\left(\frac{2m\pi}{2N+1}\right) + 2\,C\,U_1 - U_2 \tag{3}$$

$$U_2 = U_1$$

$$U_1 = U_0$$

for values of m = 2N, 2N-1, . . . , 1

The coefficients are then:

$$A_J = \frac{2}{2N+1}\left(f(0) + C\,U_1 - U_2\right) \tag{4}$$

$$B_J = \frac{2}{2N+1}\,S\,U_1 \tag{5}$$

The values of C and S are updated to:

$$Q = C_1\,C - S_1\,S$$

$$S = C_1\,S + S_1\,C$$

$$C = Q$$

j is stepped by 1 and the sequence starting at equation (3) is now repeated until M+1 pairs of coefficients have been computed.

```
C                                                                       FRIT  10
C     ...................................................................FRIT  20
C                                                                       FRIT  30
C        SUBROUTINE FORIT                                               FRIT  40
C                                                                       FRIT  50
C        PURPOSE                                                        FRIT  60
C           FOURIER ANALYSIS OF A PERIODICALLY TABULATED FUNCTION.      FRIT  70
C           COMPUTES THE COEFFICIENTS OF THE DESIRED NUMBER OF TERMS    FRIT  80
C           IN THE FOURIER SERIES F(X)=A(0)+SUM(A(K)COS KX+B(K)SIN KX)  FRIT  90
C           WHERE K=1,2,...,M TO APPROXIMATE A GIVEN SET OF             FRIT 100
C           PERIODICALLY TABULATED VALUES OF A FUNCTION.                FRIT 110
C                                                                       FRIT 120
C        USAGE                                                          FRIT 130
C           CALL FORIT(FNT,N,M,A,B,IER)                                 FRIT 140
C                                                                       FRIT 150
C        DESCRIPTION OF PARAMETERS                                      FRIT 160
C           FNT-VECTOR OF TABULATED FUNCTION VALUES OF LENGTH 2N+1      FRIT 170
C           N  -DEFINES THE INTERVAL SUCH THAT 2N+1 POINTS ARE TAKEN    FRIT 180
C              OVER THE INTERVAL (0,2PI). THE SPACING IS THUS 2PI/2N+1  FRIT 190
C           M  -MAXIMUM ORDER OF HARMONICS TO BE FITTED                 FRIT 200
C           A  -RESULTANT VECTOR OF FOURIER COSINE COEFFICIENTS OF      FRIT 210
C              LENGTH M+1                                               FRIT 220
C              A SUB 0, A SUB 1,..., A SUB M                            FRIT 230
C           B  -RESULTANT VECTOR OF FOURIER SINE COEFFICIENTS OF        FRIT 240
C              LENGTH M+1                                               FRIT 250
C              B SUB 0, B SUB 1,..., B SUB M                            FRIT 260
C           IER-RESULTANT ERROR CODE WHERE                              FRIT 270
C              IER=0  NO ERROR                                          FRIT 280
C              IER=1  N NOT GREATER OR EQUAL TO M                       FRIT 290
C              IER=2  M LESS THAN 0                                     FRIT 300
C                                                                       FRIT 310
C        REMARKS                                                        FRIT 320
C           M MUST BE GREATER THAN OR EQUAL TO ZERO                     FRIT 330
C           N MUST BE GREATER THAN OR EQUAL TO M                        FRIT 340
C           THE FIRST ELEMENT OF VECTOR B IS ZERO IN ALL CASES          FRIT 350
C                                                                       FRIT 360
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  FRIT 370
C           NONE                                                        FRIT 380
C                                                                       FRIT 390
C        METHOD                                                         FRIT 400
C           USES RECURSIVE TECHNIQUE DESCRIBED IN A. RALSTON, H. WILF,  FRIT 410
C           'MATHEMATICAL METHODS FOR DIGITAL COMPUTERS', JOHN WILEY    FRIT 420
C           AND SONS, NEW YORK, 1960, CHAPTER 24. THE METHOD OF INDEXINGFRIT 430
C           THROUGH THE PROCEDURE HAS BEEN MODIFIED TO SIMPLIFY THE     FRIT 440
C           COMPUTATION.                                                FRIT 450
C                                                                       FRIT 460
C     ...................................................................FRIT 470
C                                                                       FRIT 480
      SUBROUTINE FORIT(FNT,N,M,A,B,IER)                                 FRIT 490
      DIMENSION A(1),B(1),FNT(1)                                        FRIT 500
C                                                                       FRIT 510
C        CHECK FOR PARAMETER ERRORS                                     FRIT 520
C                                                                       FRIT 530
      IER=0                                                             FRIT 540
   20 IF(M) 30,40,40                                                    FRIT 550
   30 IER=2                                                             FRIT 560
      RETURN                                                            FRIT 570
   40 IF(M-N) 60,60,50                                                  FRIT 580
   50 IER=1                                                             FRIT 590
      RETURN                                                            FRIT 600
C                                                                       FRIT 610
C        COMPUTE AND PRESET CONSTANTS                                   FRIT 620
C                                                                       FRIT 630
   60 AN=N                                                              FRIT 640
      COEF=2.0/(2.0*AN+1.0)                                             FRIT 650
      CONST=3.141593*COEF                                               FRIT 660
      S1=SIN(CONST)                                                     FRIT 670
      C1=COS(CONST)                                                     FRIT 680
      C=1.0                                                             FRIT 690
      S=0.0                                                             FRIT 700
      J=1                                                               FRIT 710
      FNTZ=FNT(1)                                                       FRIT 720
   70 U2=0.0                                                            FRIT 730
      U1=0.0                                                            FRIT 740
      I=2*N+1                                                           FRIT 750
C                                                                       FRIT 760
C        FORM FOURIER COEFFICIENTS RECURSIVELY                          FRIT 770
C                                                                       FRIT 780
   75 U0=FNT(I)+2.0*C*U1-U2                                             FRIT 790
      U2=U1                                                             FRIT 800
      U1=U0                                                             FRIT 810
      I=I-1                                                             FRIT 820
      IF(I-1) 80,80,75                                                  FRIT 830
   80 A(J)=COEF*(FNTZ*C*U1-U2)                                          FRIT 840
      B(J)=COEF*S*U1                                                    FRIT 850
      IF(J-(M+1)) 90,100,100                                            FRIT 860
   90 Q=C1*C-S1*S                                                       FRIT 870
      S=C1*S+S1*C                                                       FRIT 880
      C=Q                                                               FRIT 890
      J=J+1                                                             FRIT 900
      GO TO 70                                                          FRIT 910
  100 A(1)=A(1)*0.5                                                     FRIT 920
      RETURN                                                            FRIT 930
      END                                                               FRIT 940
```

Subroutines HARM and DHARM

Given three integers $M_\nu, \nu = 1, 2, 3$ and a three-dimensional array of complex numbers $A(k_1, k_2, k_3)$ where $k_\nu = 0, 1, \ldots, N_\nu - 1$ and $N_\nu = 2^{M_\nu}$, this subroutine computes the three-dimensional, complex Fourier series:

$$X(j_1, j_2, j_3) = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} \sum_{k_3=0}^{N_3-1} A(k_1, k_2, k_3) \, e^{2\pi i\left(\frac{j_1 \cdot k_1}{N_1} + \frac{j_2 \cdot k_2}{N_2} + \frac{j_3 \cdot k_3}{N_3}\right)} \qquad (1)$$

where

$$j_\nu = 0, 1, \ldots, N_\nu - 1 \text{ and } \nu = 1, 2, 3.$$

The inverse series may also be computed:

$$A(k_1, k_2, k_3) = \frac{1}{N_1 * N_2 * N_3} \sum_{j_1=0}^{N_1-1} \sum_{j_2=0}^{N_2-1} \sum_{j_3=0}^{N_3-1} X(j_1, j_2, j_3) \, e^{-2\pi i\left(\frac{j_1 \cdot k_1}{N_1} + \frac{j_2 \cdot k_2}{N_2} + \frac{j_3 \cdot k_3}{N_3}\right)} \qquad (2)$$

For one dimension, the algorithm for the complex Fourier transform of complex numbers A (k) is:

$$X(j) = \sum_{k=0}^{N-1} A(k) \, w^{jk} \qquad (3)$$

where $w = e^{\frac{2\pi i}{N}}$, $N = 2^M$, $j = 0, 1, 2, \ldots, N - 1$ expresses X(j) as a function of the M arguments $j_{M-1}, j_{M-2}, \ldots, j_1, j_0$ of the binary representation of j:

$$j = j_{M-1} \cdot 2^{M-1} + \ldots + j_1 2 + j_0, \quad j_\nu = 0 \text{ or } 1$$

Then (3) can be written:

$$X(j_{M-1}, \ldots, j_1, j_0) = \sum_{k_0=0}^{1} \sum_{k_1=0}^{1} \cdots \sum_{k_{M-1}=0}^{1} A(k_{M-1}, \ldots, k_0) w^{j \cdot k_{M-1} \cdot 2^{M-1} \cdots w^{j \cdot k_0 \cdot 2^0}} \qquad (4)$$

Then $w^{jk_{M-1}2^{M-1}} = w^{j \cdot k_{M-1} 2^{M-1}}$ since $e^{2\pi i} = 1$, and the innermost sum in (4) yields an array:

$$A_1(j_0, k_{M-2}, \ldots, k_0) = \sum_{k_{M-1}=0}^{1} A(k_{M-1}, \ldots, k_0) w^{j_0 \cdot k_{M-1} \cdot 2^{M-1}} \qquad (5)$$

Successive arrays $A_2, \ldots, A_M$ obtained by summing over $k_{M-2}, \ldots, k_0$ respectively are computed. The general formula is:

$$A_L(j_0, \ldots, j_{L-1}, k_{M-L-1}, \ldots, k_0) = \sum_{k_{M-L}=0}^{1} A_{L-1}(j_0, \ldots, j_{L-2}, k_{M-L}, \ldots, k_0) w^{(j_{L-1} \cdot 2^{L-1} + \ldots + j_0) k_{M-L} 2^{M-L}} \qquad (6)$$

The final array will be the desired X. The storage indexing convention used is to let the M arguments of $A_L(j_0, \ldots, k_0)$ be the binary representation of the index of the storage location for $A_L(j_0, \ldots, k_0)$. In this way, each step of the algorithm involves fetching from two storage locations and returning results in the same two locations, thereby saving storage. However, the elements of the final array:

$$X(j_{M-1}, \ldots, j_1, j_0) = A_M(j_0, k_1, \ldots, j_{M-1})$$

are in the wrong order. The program reverses the order of bits in bit representations to compensate.

A modification of the above algorithm is made for three dimensions, and another modification is made in the L-loop to save computations.

```
C
C     ..................................................................
C
C        SUBROUTINE FARM
C
C        PURPOSE
C           PERFORMS DISCRETE COMPLEX FOURIER TRANSFORMS ON A COMPLEX
C           THREE DIMENSIONAL ARRAY
C
C        USAGE
C           CALL FARM (A,P,INV,S,IFSET,IFERR)
C
C        DESCRIPTION OF PARAMETERS
C           A      - AS INPUT, A CONTAINS THE COMPLEX, 3-DIMENSIONAL
C                    ARRAY TO BE TRANSFORMED.  THE REAL PART OF
C                    A(I1,I2,I3) IS STORED IN VECTOR FASHION IN A CELL
C                    WITH INDEX 2*(I3*N1*N2 + I2*N1 + I1) + 1 WHERE
C                    N1 = 2**P(1), I=1,2,3 AND I1 = 0,1,...,N1-1 ETC.
C                    THE IMAGINARY PART IS IN THE CELL IMMEDIATELY
C                    FOLLOWING.  NOTE THAT THE SUBSCRIPT I1 INCREASES
C                    MOST RAPIDLY AND I3 INCREASES LEAST RAPIDLY.
C                    AS OUTPUT, A CONTAINS THE COMPLEX FOURIER
C                    TRANSFORM.  THE NUMBER OF CORE LOCATIONS OF
C                    ARRAY A IS 2*(N1*N2*N3)
C           P      - A THREE CELL VECTOR WHICH DETERMINES THE SIZES
C                    OF THE 3 DIMENSIONS OF THE ARRAY A.   THE SIZE,
C                    N1, OF THE I DIMENSION OF A IS 2**P(I), I = 1,2,3
C           INV    - A VECTOR WORK AREA FOR BIT AND INDEX MANIPULATION
C                    OF DIMENSION ONE FOURTH OF THE QUANTITY
C                    MAX(N1,N2,N3)
C                    LOCATIONS OF A, VIZ., (1/8)*2*N1*N2*N3
C           S      - A VECTOR WORK AREA FOR SINE TABLES WITH DIMENSION
C                    THE SAME AS INV
C           IFSET  - AN OPTION PARAMETER WITH THE FOLLOWING SETTINGS
C                      0    SET UP SINE AND INV TABLES ONLY
C                      1    SET UP SINE AND INV TABLES ONLY AND
C                           CALCULATE FOURIER TRANSFORM
C                     -1    SET UP SINE AND INV TABLES ONLY AND
C                           CALCULATE INVERSE FOURIER TRANSFORM (FOR
C                           THE MEANING OF INVERSE SEE THE EQUATIONS
C                           UNDER METHOD BELOW)
C                      2    CALCULATE FOURIER TRANSFORM ONLY (ASSUME
C                           SINE AND INV TABLES EXIST)
C                     -2    CALCULATE INVERSE FOURIER TRANSFORM ONLY
C                           (ASSUME SINE AND INV TABLES EXIST)
C           IFERR  - ERROR INDICATOR.   WHEN IFSET IS 0,+1,-1,
C                    IFERR = I MEANS THE MAXIMUM P(I) IS GREATER THAN
C                  20 , I=1,2,3   WHEN IFSET IS 2,-2 , IFERR = 1
C                    MEANS THAT THE SINE AND INV TABLES ARE NOT LARGE
C                    ENOUGH OR HAVE NOT BEEN COMPUTED .
C                                                          E ABOVE
C
C
C        REMARKS
C           THIS SUBROUTINE IS TO BE USED FOR COMPLEX, 3-DIMENSIONAL
C           ARRAYS IN WHICH EACH DIMENSION IS A POWER OF 2.  THE
C           MAXIMUM P(I) MUST NOT BE LESS THAN 3 OR GREATER THAN 20,
C           I = 1,2,3
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           FOR IFSET = +1, OR +2, THE FOURIER TRANSFORM OF COMPLEX
C           ARRAY A IS OBTAINED.
C
C                     N1-1   N2-1   N3-1                    L1   L2   L3
C     X(J1,J2,J3)=SUM    SUM    SUM    A(K1,K2,K3)*W1  *W2  *W3
C                     K1=0   K2=0   K3=0
C
C                     WHERE WI IS THE N(I) ROOT OF UNITY AND L1=K1*J1,
C                           L2=K2*J2, L3=K3*J3
C
C
C
C           FOR IFSET = -1, OR -2, THE INVERSE FOURIER TRANSFORM A OF
C           COMPLEX ARRAY X IS OBTAINED.
C
C     A(K1,K2,K3)=
C                 1      N1-1   N2-1   N3-1                  -L1  -L2  -L3
C                --------- *SUM    SUM    SUM    X(J1,J2,J3)*W1  *W2  *W3
C                N1*N2*N3    J1=0   J2=0   J3=0
C
C
C
C           SEE J.W. COOLEY AND J.W. TUKEY, 'AN ALGORITHM FOR THE
C           MACHINE CALCULATION OF COMPLEX FOURIER SERIES',
C           MATHEMATICS OF COMPUTATIONS, VOL. 19 (APR. 1965), P. 297.
C
C     ..................................................................
C
      SUBROUTINE FARM(A,P,INV,S,IFSET, IFERR)
      DIMENSION A(1),INV(1),S(1),N(3),M(3),NP(3),N(2),W2(2),W3(2)
      EQUIVALENCE (N1,N(1)),(N2,N(2)),(N3,N(3))
    1C IF( ABS(IFSET) - 1) 50C,90C,12
    12 PTT=MAX(P(1),P(2),P(3)) -2
       RCOT2 = SQRT(2.)
       IF (PTT-PT ) 14,14,13
    13 IFERR=1
       RETURN
    14 IFERR=0
       P1=P(1)
       P2=P(2)
       P3=P(3)
       N1=2**P1
       N2=2**P2
       N3=2**P3
    1C IF(IFSET) 1E,18,2C
    1E NX= N1*N2*N3
       FN = NX
       DO 19 I = 1,NX
       A(2*I-1) = A(2*I-1)/FN
    19 A(2*I) = -A(2*I)/FN
    2C NP(1)=N1*2
       NP(2)= NP(1)*N2
       NP(3)=NP(2)*N3
       DO 25C IC=1,3
       IL = NP(3)-NP(IC)
       IL1 = IL+1
       PI = P(IC)
       IF (PI)25C,25C,3C
    3C ICIF=NP(IC)
       KEIT=NP(IC)
       PEV = 2*(PI/2)
       IF (PI - PEV )6C,6C,4C
C
C      P IS ODD. DO L=1 CASE
    4C KEIT=KEIT/2
       KL=KEIT-2
       DO 5C I=1,IL1,ICIF
       KLAST=KL+1
       DO 5C K=1,KLAST,2
       KD=K+KEIT
C
C      DO ONE STEP WITH L=1,J=0
C      A(K)=A(K)+A(KD)
C      A(KD)=A(K)-A(KD)
C
       T=A(KD)
       A(KD)=A(K)-T
       A(K)=A(K)+T
       T=A(KD+1)
       A(KD+1)=A(K+1)-T
    5C A(K+1)=A(K+1)+T
       IF (PI - 1)25C,25C,52
    52 LFIRST =3
C
C      DEF - JLAST = 2**(L-2) -1
       JLAST=1
       DO 7C
C
C      P IS EVEN
    6C LFIRST = 2
       JLAST=0
    7C DO 24C L=LFIRST,PI,2
       JJDIF=KEIT
       KEIT=KEIT/
       KL=KEIT-2
C
C      DO FOR J=0
       DO 8C I=1,IL1,ICIF
       KLAST=I+KL
       DO 8C K=1,KLAST,2
       K1=K+KEIT
       K2=K1+KEIT
       K3=K2+KEIT
C
C      DO TWO STEPS WITH J=0
C      A(K)=A(K)+A(K2)
C      A(K2)=A(K)-A(K2)
C      A(K1)=A(K1)+A(K3)
C      A(K3)=A(K1)-A(K3)
C
C      A(K)=A(K)+A(K1)
C      A(K1)=A(K)-A(K1)
C      A(K2)=A(K2)+A(K3)*I
C      A(K3)=A(K2)-A(K3)*I
C
       T=A(K2)
       A(K2)=A(K)-T
       A(K)=A(K)+T
       T=A(K2+1)
       A(K2+1)=A(K+1)-T
       A(K+1)=A(K+1)+T
C
       T=A(K3)
       A(K3)=A(K1)-T
       A(K1)=A(K1)+T
       T=A(K3+1)
       A(K3+1)=A(K1+1)-T
       A(K1+1)=A(K1+1)+T
C
       T=A(K1)
       A(K1)=A(K)-T
       A(K)=A(K)+T
       T=A(K1+1)
       A(K1+1)=A(K+1)-T
       A(K+1)=A(K+1)+T
C
       R=-A(K2+1)
       T = A(K3)
       A(K3)=A(K2)-R
       A(K2)=A(K2)+R
       A(K3+1)=A(K2+1)-T
    8C A(K2+1)=A(K2+1)+T
       IF (JLAST) 23C,23C,82
    82 JJ=JJDIF   +1
C
C      DO FOR J=1
       ILAST= IL +JJ
       DO 85 I = JJ,ILAST,ICIF
       KLAST = KL+1
       DO 85 K=1,KLAST,2
       K1 = K+KEIT
       K2 = K1+KEIT
       K3 = K2+KEIT
C
C      LETTING W=(1+I)/ROOT2,W3=(-1+I)/ROOT2,W2=I,
C      A(K)=A(K)+A(K2)*I
C      A(K2)=A(K)-A(K2)*I
C      A(K1)=A(K1)*W+A(K3)*W3
C      A(K3)=A(K1)*W-A(K3)*W3
C
C      A(K)=A(K)+A(K1)
C      A(K1)=A(K)-A(K1)
C      A(K2)=A(K2)+A(K3)*I
C      A(K3)=A(K2)-A(K3)*I
C
       R =-A(K2+1)
       T = A(K2)
       A(K2) = A(K)-R
       A(K) = A(K)+R
       A(K2+1)=A(K+1)-T
       A(K+1)=A(K+1)+T
C
       AWR=A(K1)-A(K1+1)
       AWI = A(K1+1)+A(K1)
       R=-A(K3)-A(K3+1)
       T=A(K3)-A(K3+1)
       A(K3)=(AWR-R)/RCOT2
       A(K3+1)=(AWI-T)/RCOT2
       A(K1)=(AWR+R)/RCOT2
       A(K1+1)=(AWI+T)/RCOT2
       T= A(K1)
       A(K1)=A(K)-T
       A(K)=A(K)+T
       T=A(K1+1)
       A(K1+1)=A(K+1)-T
       A(K+1)=A(K+1)+T
       R=-A(K3+1)
```

```
      T=A(K3)
      A(K3)=A(K2)-R
      A(K2)=A(K2)+R
      A(K3+1)=A(K2+1)-T
 85   A(K2+1)=A(K2+1)+T
      IF(JLAST-1) 235,235,90
 90   JJ= JJ + JJCIF

      NOW DO THE REMAINING J'S
      DO 230 J=2,JLAST

      FETCH W'S
      DEF- W=W**INV(J), WE-W- ...
 96   I=INV(J+1)
 98   IC=NT-I
      W(1)=S(IC)
      W(2)=S(I)
      I2=2*I
      I2C=NT-I2
      IF(I2C)120,110,100

      2*I IS IN FIRST QUADRANT
100   W2(1)=S(I2C)
      W2(2)=S(I2)
      GO TO 130
110   W2(1)=0.
      W2(2)=1.
      GO TO 130

      2*I IS IN SECOND QUADRANT
120   I2CC = I2C+NT
      I2C=-I2C
      W2(1)=-S(I2C)
      W2(2)=S(I2CC)
130   I3=I+I2
      I3C=NT-I3
      IF(I3C)160,150,140

      I3 IN FIRST QUADRANT
140   W3(1)=S(I3C)
      W3(2)=S(I3)
      GO TO 200
150   W3(1)=0.
      W3(2)=1.
      GO TO 200

160   I3CC=I3C+NT
      IF(I3CC)180,180,170

      I3 IN SECOND QUADRANT
170   I3C=-I3C
      W3(1)=-S(I3C)
      W3(2)=S(I3CC)
      GO TO 200
180   W3(1)=-1.
      W3(2)=0.
      GO TO 200

      3*I IN THIRD QUADRANT
190   I3CCC=NT+I3CC
      I3CC = -I3CC
      W3(1)=-S(I3CCC)
      W3(2)=-S(I3CC)
200   ILAST=IL+
      DO 220 I=.
      KLAST=KL+.
      DO 220 K=I,KLAST,2
      K1=K+KDIT
      K2=K1+KDIT
      K3=K2+KDIT

      DO TWO STEPS WITH J NOT 0
      A(K)=A(K)+A(K2)*W2
      A(K2)=A(K)-A(K2)*W2
      A(K1)=A(K1)+W*A(K3)*W3
      A(K3)=A(K1)*W-A(K3)*W3

      A(K)=A(K)+A(K1)
      A(K1)=A(K)-A(K1)
      A(K2)=A(K2)+A(K3)*I
      A(K3)=A(K2)-A(K3)*I

      R=A(K2)*W2(1)-A(K2+1)*W2(2)
      T=A(K2)*W2(2)+A(K2+1)*W2(1)
      A(K2)=A(K)-R
      A(K)=A(K)+R
      A(K2+1)=A(K+1)-T
      A(K+1)=A(K+1)+T

      R=A(K3)*W3(1)-A(K3+1)*W3(2)
      T=A(K3)*W3(2)+A(K3+1)*W3(1)
      AWR=A(K1)*W(1)-A(K1+1)*W(2)
      AWI=A(K1)*W(2)+A(K1+1)*W(1)
      A(K3)=AWR-R
      A(K3+1)=AWI-T
      A(K1)=AWR+R
      A(K1+1)=AWI+T
      T=A(K1)
      A(K1)=A(K)-T
      A(K)=A(K)+T
      T=A(K1+1)
      A(K1+1)=A(K+1)-T
      A(K+1)=A(K+1)+T
      R=-A(K3+1)
      T=A(K3)
      A(K3)=A(K2)-R
      A(K2)=A(K2)+R
      A(K3+1)=A(K2+1)-T
220   A(K2+1)=A(K2+1)+T
      END OF I AND K LOOPS

230   JJ=JJCIF+JJ
      END OF J-LOOP

235   JLAST=4*J
240   CONTINUE
      END OF  L

250   CONTINUE
      END OF  IC  LOOP

      WE NOW HAVE THE COMPLEX FOURIER SUMS BUT THEIR ADDRESSES ARE
      BIT-REVERSED.  THE FOLLOWING ROUTINE PUTS THEM IN ORDER
      NTSC=NT+NT
      N3PT=N3-PT
350   IF(N3PT) 370,360,360

      N3 GO. OR EQ. PT
360   IGO3=1
      N3VAT=N3/NT
      MINN3=NT
      GO TO 380
C
C     N3 LESS THAN NT
370   IGO3=2
      N3VAT=1
      NTVN3=NT/N3
      MINN3=N3
380   JJC3 = NTSC/N3
      N2PT=N2-PT
450   IF (N2PT)470,460,460
C
C     N2 GO. OR EQ. NT
460   IGO2=1
      N2VAT=N2/NT
      MINN2=NT
      GO TO 480
C
C     N2 LESS THAN NT
470   IGO2 = 2
      N2VAT=1
      NTVN2=NT/N2
      MINN2=N2
480   JJC2=NTSC/N2
      N1PT=N1-PT
550   IF(N1PT)570,560,560
C
C     N1 GO. OR EQ. NT
560   IGO1=1
      N1VAT=N1/NT
      MINN1=N1
      GO TO 580
C
C     N1 LESS THAN NT
570   IGO1=2
      N1VAT=1
      NTVN1=NT/N1
      MINN1=N1
580   JJC1=NTSC/N1
600   JJ3=1
      J=1
      GO TO 680 JFF3=1,N3VAT
      IPF3=INV(JJ3)
      GO TO 670 JF3=1,MINN3
      GO TO (610,620),IGO3
610   IF3=INV(JF3)*N3VAT
      GO TO 630
620   IF3=INV(JF3)/NTVN3
630   I3=(IPF3+IF3)*N2
700   JJ2=1
      GO TO 670 JFF2=1,N2VAT
      IPF2=INV(JJ2)*I3
      GO TO 660 JF2=1,MINN2
      GO TO (710,720),IGO2
710   IF2=INV(JF2)*N2VAT
      GO TO 730
720   IF2=INV(JF2)/NTVN2
730   I2=(IPF2+IP2)*N1
800   JJ1=1
      GO TO 660 JFF1=1,N1VAT
      IPF1=INV(JJ1)*I2
      GO TO 650 JF1=1,MINN1
      GO TO (810,820),IGO1
810   IF1=INV(JF1)*N1VAT
      GO TO 830
820   IF1=INV(JF1)/NTVN1
830   I=2*(IPF1+IF1)+1
      IF (J-I) 840,850,850
840   T=A(I)
      A(I)=A(J)
      A(J)=T
      T=A(I+1)
      A(I+1)=A(J+1)
      A(J+1)=T
850   J=J+2
860   JJ1=JJ1+JJC1
C     END OF JFF1 AND JF2
C
870   JJ2=JJ2+JJC2
C     END OF JFF2 AND JF3 LOOPS
C
880   JJ3 = JJ3+JJC3
C     END OF JFF3 LOOP
C
890   IF(IFSET)891,895,895
891   DO 892 I = 1,NX
892   A(2*I) = -A(2*I)
895   RETURN
C
C     THE FOLLOWING PROGRAM CO                                  V TABLES.
C
900   MT=MAX0(M(1),M(2),M(3))
      MT = MAX0(2,MT)
904   IF (MT-16) 906,906,12
906   IFERR=0
      NT=2**MT
      NTV2=NT/2
C
C     SET UP SIN TABLE
C     THETA=PIE/2**(L+1) FOR L=1
910   THETA=.78539816334
C
C     JSTEP=2**(MT-L+1) FOR L=1
      JSTEP=NT
C
C     JCIF=2**(MT-L) FOR L=1
      JCIF=NTV2
      S(JCIF)=SIN(THETA)
      DO 950 L=2,MT
      THETA=THETA/2.
      JSTEP2=JSTEP
      JSTEP=JCIF
      JCIF=JSTEP/2
      S(JCIF)=SIN(THETA)
      JC1=NT-JCIF
      S(JC1)=COS(THETA)
      JLAST=NT-JSTEP2
      IF(JLAST - JSTEP) 950,920,920
920   DO 940 J=JSTEP,JLAST,JSTEP
      JC=NT-J
      JC=J+JCIF
940   S(JC)=S(J)*S(JC1)+S(JCIF)*S(JC)
950   CONTINUE
C
C     SET UP INV(J) TABLE
C
960   MTEXP=NTV2
C
C     MTEXP=2**(MT-L). FOR L=1
      LMTEXP=1
```

```
C                                                                       HARM5120
C     LM1EXF=2**(L-1). FOR L=1                                          HARM5130
      INV(1)=0                                                          HARM5140
      DO 96C L=1,MT                                                     HARM5150
      INV(LM1EXF+1) = MTLEXP                                            HARM5160
      DO 97C J=2,LM1EXF                                                 HARM5170
      JJ=J+LM1EXF                                                       HARM5180
   97C INV(JJ)=INV(J)+MTLEXP                                            HARM5190
      MTLEXF=MTLEXP/2                                                   HARM5200
   96C LM1EXF=LM1EXF*2                                                  HARM5210
   982 IF(IFSET)12,855,12                                               HARM5220
      END                                                               HARM5230
```

```
C                                                                       DHAR  10
C     ..................................................................DHAR  20
C                                                                       DHAR  30
C        SUBROUTINE CHARM                                               DHAR  40
C                                                                       DHAR  50
C        PURPOSE                                                        DHAR  60
C           PERFORMS DISCRETE COMPLEX FOURIER TRANSFORMS ON A COMPLEX   DHAR  70
C           DOUBLE PRECISION,THREE DIMENSIONAL ARRAY                    DHAR  80
C                                                                       DHAR  90
C        USAGE                                                          DHAR 100
C           CALL CHARM(A,M,INV,S,IFSET,IFERR)                           DHAR 110
C                                                                       DHAR 120
C        DESCRIPTION OF PARAMETERS                                      DHAR 130
C           A     - A DOUBLE PRECISION VECTOR                           DHAR 140
C                   AS INPUT, A CONTAINS THE COMPLEX, 3-DIMENSIONAL     DHAR 150
C                   ARRAY TO BE TRANSFORMED.  THE REAL PART OF          DHAR 160
C                   A(I1,I2,I3) IS STORED IN VECTOR FASHION IN A CELL   DHAR 170
C                   WITH INDEX 2*(I3*N1*N2 + I2*N1 + I1) + 1 WHERE      DHAR 180
C                   NI = 2**M(I), I=1,2,3 AND I1 = 0,1,...,N1-1 ETC.    DHAR 190
C                   THE IMAGINARY PART IS IN THE CELL IMMEDIATELY       DHAR 200
C                   FOLLOWING.  NOTE THAT THE SUBSCRIPT I1 INCREASES    DHAR 210
C                   MOST RAPIDLY AND I3 INCREASES LEAST RAPIDLY.        DHAR 220
C                   AS OUTPUT, A CONTAINS THE COMPLEX FOURIER           DHAR 230
C                   TRANSFORM.  THE NUMBER OF CORE LOCATIONS OF         DHAR 240
C                   ARRAY A IS 2*(N1*N2*N3)                             DHAR 250
C           M     - A THREE CELL VECTOR WHICH DETERMINES THE SIZES      DHAR 260
C                   OF THE 3 DIMENSIONS OF THE ARRAY A.  THE SIZE,      DHAR 270
C                   NI, OF THE I DIMENSION OF A IS 2**M(I), I = 1,2,3   DHAR 280
C           INV   - A VECTOR WORK AREA FOR BIT AND INDEX MANIPULATION   DHAR 290
C                   OF DIMENSION ONE FOURTH OF THE QUANTITY             DHAR 300
C                   MAX(N1,N2,N3)                                       DHAR 310
C                   LOCATIONS OF A, VIZ., (1/8)*2*N1*N2*N3              DHAR 310
C           S     - A DOUBLE PRECISION VECTOR WORK AREA FOR SINE TABLES DHAR 320
C                   WITH DIMENSION THE SAME AS INV                      DHAR 330
C           IFSET - AN OPTION PARAMETER WITH THE FOLLOWING SETTINGS     DHAR 340
C                      0    SET UP SINE AND INV TABLES ONLY             DHAR 350
C                      1    SET UP SINE AND INV TABLES ONLY AND         DHAR 360
C                           CALCULATE FOURIER TRANSFORM                 DHAR 370
C                     -1    SET UP SINE AND INV TABLES ONLY AND         DHAR 380
C                           CALCULATE INVERSE FOURIER TRANSFORM (FOR    DHAR 390
C                           THE MEANING OF INVERSE SEE THE EQUATIONS    DHAR 400
C                           UNDER METHOD BELOW)                         DHAR 410
C                      2    CALCULATE FOURIER TRANSFORM ONLY (ASSUME    DHAR 420
C                           SINE AND INV TABLES EXIST)                  DHAR 430
C                     -2    CALCULATE INVERSE FOURIER TRANSFORM ONLY    DHAR 440
C                           (ASSUME SINE AND INV TABLES EXIST)          DHAR 450
C           IFERR - ERROR INDICATOR.  WHEN IFSET IS 0,+1,-1,            DHAR 460
C                   IFERR = 1 MEANS THE MAXIMUM M(I) IS GREATER THAN    DHAR 470
C                   20, I=1,2,3   WHEN IFSET IS 2,-2 , IFERR = 1        DHAR 480
C                   MEANS THAT THE SINE AND INV TABLES ARE NOT LARGE    DHAR 490
C                   ENOUGH OR HAVE NOT BEEN COMPUTED .                  DHAR 500
C                   IF ON RETURN IFERR = 0 THEN NONE OF THE ABOVE       DHAR 510
C                   CONDITIONS ARE PRESENT                              DHAR 520
C                                                                       DHAR 530
C        REMARKS                                                        DHAR 540
C           THIS SUBROUTINE IS TO BE USED FOR COMPLEX, DOUBLE PRECISION,DHAR 550
C           3-DIMENSIONAL ARRAYS IN WHICH EACH DIMENSION IS A POWER OF  DHAR 560
C           2. THE MAXIMUM M(I) MUST NOT BE LESS THAN 3 OR GREATER THAN DHAR 570
C           20, I = 1,2,3.                                              DHAR 580
C                                                                       DHAR 590
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DHAR 600
C           NONE                                                        DHAR 610
C                                                                       DHAR 620
C        METHOD                                                         DHAR 630
C           FOR IFSET = +1, OR +2, THE FOURIER TRANSFORM OF COMPLEX     DHAR 640
C           ARRAY A IS OBTAINED.                                        DHAR 650
C                                                                       DHAR 660
C                   N1-1   N2-1   N3-1                   L1   L2   L3    DHAR 670
C     X(J1,J2,J3)=SUM    SUM    SUM    A(K1,K2,K3)*W1  *W2  *W3        DHAR 680
C                   K1=0   K2=0   K3=0                                  DHAR 690
C                                                                       DHAR 700
C                   WHERE WI IS THE N(I) ROOT OF UNITY AND L1=K1*J1,    DHAR 710
C                         L2=K2*J2, L3=K3*J3                            DHAR 720
C                                                                       DHAR 730
C                                                                       DHAR 740
C           FOR IFSET = -1, OR -2, THE INVERSE FOURIER TRANSFORM A OF   DHAR 750
C           COMPLEX ARRAY X IS OBTAINED.                                DHAR 760
C                                                                       DHAR 770
C     A(K1,K2,K3)=                                                      DHAR 780
C                 1       N1-1   N2-1   N3-1                  -L1  -L2  -L3 DHAR 790
C           -------- *SUM    SUM    SUM    X(J1,J2,J3)*W1  *W2  *W3    DHAR 800
C           N1*N2*N3   J1=0   J2=0   J3=0                               DHAR 810
C                                                                       DHAR 820
C                                                                       DHAR 830
C           SEE J.W. COOLEY AND J.W. TUKEY, 'AN ALGORITHM FOR THE       DHAR 840
C           MACHINE CALCULATION OF COMPLEX FOURIER SERIES',             DHAR 850
C           MATHEMATICS OF COMPUTATIONS, VOL. 19 (APR. 1965), P. 297.   DHAR 860
C                                                                       DHAR 870
C     ..................................................................DHAR 880
C                                                                       DHAR 890
      SUBROUTINE CHARM(A,M,INV,S,IFSET,IFERR)                           DHAR 900
      DIMENSION A(1),INV(1),S(1),N(3),M(3),NP(3),W(2),W2(2),W3(2)       DHAR 910
      DOUBLE PRECISION A,W,W3,AW1,THETA,HCCT2,S,T,W,W2,FN,AWR           DHAR 920
      EQUIVALENCE (N1,N(1)),(N2,N(2)),(N3,N(3))                         DHAR 930
   10 IF( IABS(IFSET) - 1) 500,300,12                                   DHAR 940
   12 MTT=MAX0(M(1),M(2),M(3)) -2                                       DHAR 950
      HCCT2=DSQRT(2.0D0)                                                DHAR 960
      IF (MTT-MT ) 14,14,13                                             DHAR 970
   13 IFERR=1                                                           DHAR 980
      RETURN                                                            DHAR 990
   14 IFERR=0                                                           DHAR1000
      N1=M(1)                                                           DHAR1010
      N2=M(2)                                                           DHAR1020
      N3=M(3)                                                           DHAR1030
      N1=2**N1                                                          DHAR1040
      N2=2**N2                                                          DHAR1050
      N3=2**N3                                                          DHAR1060
   16 IF(IFSET) 16,16,20                                                DHAR1070
   16 NX= N1*N2*N3                                                      DHAR1080
      FN = NX                                                           DHAR1090
      DO 19 I = 1,NX                                                    DHAR1100
      A(2*I-1) = A(2*I-1)/FN                                            DHAR1110
   19 A(2*I) = -A(2*I)/FN                                               DHAR1120
   20 NP(1)=N1*2                                                        DHAR1130
      NP(2)= NP(1)*N2                                                   DHAR1140
      NP(3)=NP(2)*N3                                                    DHAR1150
      DO 250 IC=1,3                                                     DHAR1160
      IL = NP(3)-NP(IC)                                                 DHAR1170
      IL1 = IL+1                                                        DHAR1180
      PI = M(IC)                                                        DHAR1190
      IF (PI)250,250,30                                                 DHAR1200
   30 ICIF=NP(IC)                                                       DHAR1210
      KBIT=NP(IC)                                                       DHAR1220
      MEV = 2*(PI/2)                                                    DHAR1230
      IF (PI - MEV )60,60,40                                            DHAR1240
C                                                                       DHAR1250
C     M IS ODD. DO L=1 CASE                                             DHAR1260
   40 KBIT=KBIT/2                                                       DHAR1270
      KL=KBIT-2                                                         DHAR1280
      DO 50 I=1,IL1,ICIF                                                DHAR1290
      KLAST=KL+1                                                        DHAR1300
      DO 50 K=1,KLAST,2                                                 DHAR1310
      KO=K+KBIT                                                         DHAR1320
C                                                                       DHAR1330
C     DO ONE STEP WITH L=1,J=0                                          DHAR1340
C     A(K)=A(K)+A(KO)                                                   DHAR1350
C     A(KO)=A(K)-A(KO)                                                  DHAR1360
C                                                                       DHAR1370
      T=A(KO)                                                           DHAR1380
      A(KO)=A(K)-T                                                      DHAR1390
      A(K)=A(K)+T                                                       DHAR1400
      T=A(KO+1)                                                         DHAR1410
      A(KO+1)=A(K+1)-T                                                  DHAR1420
   50 A(K+1)=A(K+1)+T                                                   DHAR1430
      IF (PI - 1)250,250,52                                             DHAR1440
   52 LFIRST =3                                                         DHAR1450
C                                                                       DHAR1460
C     DEF - JLAST = 2**(L-2) -1                                         DHAR1470
      JLAST=1                                                           DHAR1480
      GO TO 70                                                          DHAR1490
C                                                                       DHAR1500
C     M IS EVEN                                                         DHAR1510
   60 LFIRST = 2                                                        DHAR1520
      JLAST=0                                                           DHAR1530
   70 DO 240 L=LFIRST,MI,2                                              DHAR1540
      JJCIF=KBIT                                                        DHAR1550
      KBIT=KBIT/4                                                       DHAR1560
      KL=KBIT-2                                                         DHAR1570
C                                                                       DHAR1580
C     DO FOR J=0                                                        DHAR1590
      DO 80 I=1,IL1,ICIF                                                DHAR1600
      KLAST=I+KL                                                        DHAR1610
      DO 80 K=1,KLAST,2                                                 DHAR1620
      K1=K+KBIT                                                         DHAR1630
      K2=K1+KBIT                                                        DHAR1640
      K3=K2+KBIT                                                        DHAR1650
C                                                                       DHAR1660
C     DO TWO STEPS WITH J=0                                             DHAR1670
C     A(K)=A(K)+A(K2)                                                   DHAR1680
C     A(K2)=A(K)-A(K2)                                                  DHAR1690
C     A(K1)=A(K1)+A(K3)                                                 DHAR1700
C     A(K3)=A(K1)-A(K3)                                                 DHAR1710
C                                                                       DHAR1720
C     A(K)=A(K)+A(K1)                                                   DHAR1730
C     A(K1)=A(K)-A(K1)                                                  DHAR1740
C     A(K2)=A(K2)+A(K3)*I                                               DHAR1750
C     A(K3)=A(K2)-A(K3)*I                                               DHAR1760
C                                                                       DHAR1770
      T=A(K2)                                                           DHAR1780
      A(K2)=A(K)-T                                                      DHAR1790
      A(K)=A(K)+T                                                       DHAR1800
      T=A(K2+1)                                                         DHAR1810
      A(K2+1)=A(K+1)-T                                                  DHAR1820
      A(K+1)=A(K+1)+T                                                   DHAR1830
C                                                                       DHAR1840
      T=A(K3)                                                           DHAR1850
      A(K3)=A(K1)-T                                                     DHAR1860
      A(K1)=A(K1)+T                                                     DHAR1870
      T=A(K3+1)                                                         DHAR1880
      A(K3+1)=A(K1+1)-T                                                 DHAR1890
      A(K1+1)=A(K1+1)+T                                                 DHAR1900
C                                                                       DHAR1910
      T=A(K1)                                                           DHAR1920
      A(K1)=A(K)-T                                                      DHAR1930
      A(K)=A(K)+T                                                       DHAR1940
      T=A(K1+1)                                                         DHAR1950
      A(K1+1)=A(K+1)-T                                                  DHAR1960
      A(K+1)=A(K+1)+T                                                   DHAR1970
C                                                                       DHAR1980
      R=-A(K3+1)                                                        DHAR1990
      T = A(K3)                                                         DHAR2000
      A(K3)=A(K2)-R                                                     DHAR2010
      A(K2)=A(K2)+R                                                     DHAR2020
      A(K3+1)=A(K2+1)-T                                                 DHAR2030
   80 A(K2+1)=A(K2+1)+T                                                 DHAR2040
      IF (JLAST) 235,235,82                                             DHAR2050
   82 JJ=JJCIF   +1                                                     DHAR2060
C                                                                       DHAR2070
C     DO FOR J=1                                                        DHAR2080
      ILAST= IL +JJ                                                     DHAR2090
      DO 85 I = JJ,ILAST,ICIF                                           DHAR2100
      KLAST = KL+1                                                      DHAR2110
      DO 85 K=1,KLAST,2                                                 DHAR2120
      K1 = K+KBIT                                                       DHAR2130
      K2 = K1+KBIT                                                      DHAR2140
      K3 = K2+KBIT                                                      DHAR2150
C                                                                       DHAR2160
C     LETTING W=(1+I)/HCCT2,W3=(-1+I)/HCCT2,W2=I,                       DHAR2170
C     A(K)=A(K)+A(K2)*I                                                 DHAR2180
C     A(K2)=A(K)-A(K2)*I                                                DHAR2190
C     A(K1)=A(K1)*W+A(K3)*W3                                            DHAR2200
C     A(K3)=A(K1)*W-A(K3)*W3                                            DHAR2210
C                                                                       DHAR2220
C     A(K)=A(K)+A(K1)                                                   DHAR2230
C     A(K1)=A(K)-A(K1)                                                  DHAR2240
C     A(K2)=A(K2)+A(K3)*I                                               DHAR2250
C     A(K3)=A(K2)-A(K3)*I                                               DHAR2260
C                                                                       DHAR2270
      R =-A(K2+1)                                                       DHAR2280
      T = A(K2)                                                         DHAR2290
      A(K2) = A(K)-R                                                    DHAR2300
      A(K) = A(K)+R                                                     DHAR2310
      A(K2+1)=A(K+1)-T                                                  DHAR2320
      A(K+1)=A(K+1)+T                                                   DHAR2330
C                                                                       DHAR2340
      AWR=A(K1)-A(K1+1)                                                 DHAR2350
```

```
      AKI = A(K1+1)+A(K1)                                               DHAR2360
      R=-A(K3)-A(K3+1)                                                  DHAR2370
      T=A(K3)-A(K3+1)                                                   DHAR2380
      A(K3)=(AKR-R)/RCCT2                                               DHAR2390
      A(K3+1)=(AKI-T)/RCCT2                                             DHAR2400
      A(K1)=(AKR+R)/RCCT2                                               DHAR2410
      A(K1+1)=(AKI+T)/RCCT2                                             DHAR2420
      T= A(K1)                                                          DHAR2430
      A(K1)=A(K)-T                                                      DHAR2440
      A(K)=A(K)+T                                                       DHAR2450
      T=A(K1+1)                                                         DHAR2460
      A(K1+1)=A(K+1)-T                                                  DHAR2470
      A(K+1)=A(K+1)+T                                                   DHAR2480
      R=-A(K2+1)                                                        DHAR2490
      T=A(K2)                                                           DHAR2500
      A(K2)=A(K2)-R                                                     DHAR2510
      A(K2)=A(K2)+R                                                     DHAR2520
      A(K2+1)=A(K2+1)-T                                                 DHAR2530
   85 A(K2+1)=A(K2+1)+T                                                 DHAR2540
      IF(JLAST-1) 235,235,90                                            DHAR2550
   90 JJ = JJ + JJDIF                                                   DHAR2560
C                                                                       DHAR2570
C     NOW DO THE REMAINING J'S                                          DHAR2580
      DO 220 J=2,JLAST                                                  DHAR2590
C                                                                       DHAR2600
C     FETCH W'S                                                         DHAR2610
C     DEF- W=W**INV(J), W2=W**2, W3=W**3                                DHAR2620
   95 I=INV(J+1)                                                        DHAR2630
   96 IC=NT-I                                                           DHAR2640
      W(1)=S(IC)                                                        DHAR2650
      W(2)=S(I)                                                         DHAR2660
      I2=2*I                                                            DHAR2670
      I2C=NT-I2                                                         DHAR2680
      IF(I2C)120,110,100                                                DHAR2690
C                                                                       DHAR2700
C     2*I IS IN FIRST QUADRANT                                          DHAR2710
  100 W2(1)=S(I2C)                                                      DHAR2720
      W2(2)=S(I2)                                                       DHAR2730
      GO TO 130                                                         DHAR2740
  110 W2(1)=0.                                                          DHAR2750
      W2(2)=1.                                                          DHAR2760
      GO TO 130                                                         DHAR2770
C                                                                       DHAR2780
C     2*I IS IN SECOND QUADRANT                                         DHAR2790
  120 I2C = I2C+NT                                                      DHAR2800
      I2C=-I2C                                                          DHAR2810
      W2(1)=-S(I2C)                                                     DHAR2820
      W2(2)=S(I2C)                                                      DHAR2830
  130 I3=I+I2                                                           DHAR2840
      I3C=NT-I3                                                         DHAR2850
      IF(I3C)160,150,140                                                DHAR2860
C                                                                       DHAR2870
C     I3 IN FIRST QUADRANT                                              DHAR2880
  140 W3(1)=S(I3C)                                                      DHAR2890
      W3(2)=S(I3)                                                       DHAR2900
      GO TO 200                                                         DHAR2910
  150 W3(1)=0.                                                          DHAR2920
      W3(2)=1.                                                          DHAR2930
      GO TO 200                                                         DHAR2940
C                                                                       DHAR2950
  160 I3CC=I3C+NT                                                       DHAR2960
      IF(I3CC)190,180,170                                               DHAR2970
C                                                                       DHAR2980
C     I3 IN SECOND QUADRANT                                             DHAR2990
  170 I3C=-I3C                                                          DHAR3000
      W3(1)=-S(I3C)                                                     DHAR3010
      W3(2)=S(I3CC)                                                     DHAR3020
      GO TO 200                                                         DHAR3030
  180 W3(1)=-1.                                                         DHAR3040
      W3(2)=0.                                                          DHAR3050
      GO TO 200                                                         DHAR3060
C                                                                       DHAR3070
C     3*I IN THIRD QUADRANT                                             DHAR3080
  190 I3CCC=NT+I3CC                                                     DHAR3090
      I3CC = -I3CC                                                      DHAR3100
      W3(1)=-S(I3CCC)                                                   DHAR3110
      W3(2)=-S(I3CC)                                                    DHAR3120
  200 ILAST=IL+JJ                                                       DHAR3130
      DO 220 I=JJ,ILAST,ICIF                                            DHAR3140
      KLAST=KL+I                                                        DHAR3150
      DO 220 K=I,KLAST,2                                                DHAR3160
      K1=K+KEIT                                                         DHAR3170
      K2=K1+KEIT                                                        DHAR3180
      K3=K2+KEIT                                                        DHAR3190
C                                                                       DHAR3200
C     DO THE STEPS WITH J NOT 0                                         DHAR3210
C     A(K)=A(K)+A(K2)*W2                                                DHAR3220
C     A(K2)=A(K)-A(K2)*W2                                               DHAR3230
C     A(K1)=A(K1)+W*A(K3)*W3                                            DHAR3240
C     A(K3)=A(K1)*W-A(K3)*W3                                            DHAR3250
C                                                                       DHAR3260
C     A(K)=A(K)+A(K1)                                                   DHAR3270
C     A(K1)=A(K)-A(K1)                                                  DHAR3280
C     A(K2)=A(K2)+A(K3)*I                                               DHAR3290
C     A(K3)=A(K2)-A(K3)*I                                               DHAR3300
C                                                                       DHAR3310
      R=A(K2)*W2(1)-A(K2+1)*W2(2)                                       DHAR3320
      T=A(K2)*W2(2)+A(K2+1)*W2(1)                                       DHAR3330
      A(K2)=A(K)-R                                                      DHAR3340
      A(K)=A(K)+R                                                       DHAR3350
      A(K2+1)=A(K+1)-T                                                  DHAR3360
      A(K+1)=A(K+1)+T                                                   DHAR3370
C                                                                       DHAR3380
      R=A(K3)*W3(1)-A(K3+1)*W3(2)                                       DHAR3390
      T=A(K3)*W3(2)+A(K3+1)*W3(1)                                       DHAR3400
      AKR=A(K1)*W(1)-A(K1+1)*W(2)                                       DHAR3410
      AKI=A(K1)*W(2)+A(K1+1)*W(1)                                       DHAR3420
      A(K3)=AKR-R                                                       DHAR3430
      A(K3+1)=AKI-T                                                     DHAR3440
      A(K1)=AKR+R                                                       DHAR3450
      A(K1+1)=AKI+T                                                     DHAR3460
      T=A(K1)                                                           DHAR3470
      A(K1)=A(K)-T                                                      DHAR3480
      A(K)=A(K)+T                                                       DHAR3490
      T=A(K1+1)                                                         DHAR3500
      A(K1+1)=A(K+1)-T                                                  DHAR3510
      A(K+1)=A(K+1)+T                                                   DHAR3520
      R=-A(K2+1)                                                        DHAR3530
      T=A(K2)                                                           DHAR3540
      A(K3)=A(K2)-R                                                     DHAR3550
      A(K2)=A(K2)+R                                                     DHAR3560
      A(K3+1)=A(K2+1)-T                                                 DHAR3570
  220 A(K2+1)=A(K2+1)+T                                                 DHAR3580
C     END OF I AND K LOOPS                                              DHAR3590
C                                                                       DHAR3600
  230 JJ=JJDIF+JJ                                                       DHAR3610
C     END OF J-LOOP                                                     DHAR3620
C                                                                       DHAR3630
  235 JLAST=4*JLAST+3                                                   DHAR3640
  240 CONTINUE                                                          DHAR3650
C     END OF  L  LOOP                                                   DHAR3660
C                                                                       DHAR3670
  250 CONTINUE                                                          DHAR3680
C     END OF  IC  LOOP                                                  DHAR3690
C                                                                       DHAR3700
C     WE NOW HAVE THE COMPLEX FOURIER SUMS BUT THEIR ADDRESSES ARE      DHAR3710
C     BIT-REVERSED.  THE FOLLOWING ROUTINE PUTS THEM IN ORDER           DHAR3720
      NTSQ=NT*NT                                                        DHAR3730
      M3MT=M3-MT                                                        DHAR3740
  350 IF(M3MT) 370,360,360                                              DHAR3750
C                                                                       DHAR3760
C     M3 GR. OR EQ. MT                                                  DHAR3770
  360 IGC3=1                                                            DHAR3780
      N3VNT=N3/NT                                                       DHAR3790
      MINN3=NT                                                          DHAR3800
      GO TO 380                                                         DHAR3810
C                                                                       DHAR3820
C     M3 LESS THAN MT                                                   DHAR3830
  370 IGC3=2                                                            DHAR3840
      N3VNT=1                                                           DHAR3850
      NTVN3=NT/N3                                                       DHAR3860
      MINN3=N3                                                          DHAR3870
  380 JJC3 = NTSQ/N3                                                    DHAR3880
      M2MT=M2-MT                                                        DHAR3890
  450 IF (M2MT)470,460,460                                              DHAR3900
C                                                                       DHAR3910
C     M2 GR. OR EQ. MT                                                  DHAR3920
  460 IGC2=1                                                            DHAR3930
      N2VNT=N2/NT                                                       DHAR3940
      MINN2=NT                                                          DHAR3950
      GO TO 480                                                         DHAR3960
C                                                                       DHAR3970
C     M2 LESS THAN MT                                                   DHAR3980
  470 IGC2 = 2                                                          DHAR3990
      N2VNT=1                                                           DHAR4000
      NTVN2=NT/N2                                                       DHAR4010
      MINN2=N2                                                          DHAR4020
  480 JJC2=NTSQ/N2                                                      DHAR4030
      M1M1=M1-MT                                                        DHAR4040
  550 IF(M1MT)570,560,560                                               DHAR4050
C                                                                       DHAR4060
C     M1 GR. OR EQ. MT                                                  DHAR4070
  560 IGC1=1                                                            DHAR4080
      N1VNT=N1/NT                                                       DHAR4090
      MINN1=NT                                                          DHAR4100
      GO TO 580                                                         DHAR4110
C                                                                       DHAR4120
C     M1 LESS THAN MT                                                   DHAR4130
  570 IGC1=2                                                            DHAR4140
      N1VNT=1                                                           DHAR4150
      NTVN1=NT/N1                                                       DHAR4160
      MINN1=N1                                                          DHAR4170
  580 JJC1=NTSQ/N1                                                      DHAR4180
  600 JJ3=1                                                             DHAR4190
      J=1                                                               DHAR4200
      DO 880 JPP3=1,N3VNT                                               DHAR4210
      IPP3=INV(JJ3)                                                     DHAR4220
      DO 870 JP3=1,MINN3                                                DHAR4230
      GO TO (610,620),IGC3                                              DHAR4240
  610 IP3=INV(JP3)*N3VNT                                                DHAR4250
      GO TO 630                                                         DHAR4260
  620 IP3=INV(JP3)/NTVN3                                                DHAR4270
  630 I3=(IPP3+IP3)*N2                                                  DHAR4280
  700 JJ2=1                                                             DHAR4290
      DO 870 JPP2=1,N2VNT                                               DHAR4300
      IPP2=INV(JJ2)+I3                                                  DHAR4310
      DO 860 JP2=1,MINN2                                                DHAR4320
      GO TO (710,720),IGC2                                              DHAR4330
  710 IP2=INV(JP2)*N2VNT                                                DHAR4340
      GO TO 730                                                         DHAR4350
  720 IP2=INV(JP2)/NTVN2                                                DHAR4360
  730 I2=(IPP2+IP2)*N1                                                  DHAR4370
  800 JJ1=1                                                             DHAR4380
      DO 860 JPP1=1,N1VNT                                               DHAR4390
      IPP1=INV(JJ1)+I2                                                  DHAR4400
      DO 850 JP1=1,MINN1                                                DHAR4410
      GO TO (810,820),IGC1                                              DHAR4420
  810 IP1=INV(JP1)*N1VNT                                                DHAR4430
      GO TO 830                                                         DHAR4440
  820 IP1=INV(JP1)/NTVN1                                                DHAR4450
  830 I=2*(IPP1+IP1)+1                                                  DHAR4460
      IF (J-I) 840,850,850                                              DHAR4470
  840 T=A(I)                                                            DHAR4480
      A(I)=A(J)                                                         DHAR4490
      A(J)=T                                                            DHAR4500
      T=A(I+1)                                                          DHAR4510
      A(I+1)=A(J+1)                                                     DHAR4520
      A(J+1)=T                                                          DHAR4530
  850 J=J+2                                                             DHAR4540
  860 JJ1=JJ1+JJD1                                                      DHAR4550
C                                                                       DHAR4560
  870 JJ2=JJ2+JJD2                                                      DHAR4580
C     END OF JPP2 AND JP3 LOOPS                                         DHAR4590
C                                                                       DHAR4600
  880 JJ3 = JJ3+JJD3                                                    DHAR4610
C     END OF JPP3 LOOP                                                  DHAR4620
C                                                                       DHAR4630
  890 IF(IFSET)891,895,895                                              DHAR4640
  891 DO 892 I = 1,NX                                                   DHAR4650
  892 A(2*I) = -A(2*I)                                                  DHAR4660
  895 RETURN                                                            DHAR4670
C                                                                       DHAR4680
C     THE FOLLOWING PROGRAM COMPUTES THE SIN AND INV TABLES.            DHAR4690
C                                                                       DHAR4700
  900 MT=MAX0(M(1),M(2),M(3)) -2                                        DHAR4710
      MT = MAX0(2,MT)                                                   DHAR4720
  904 IF (MT-18) 906,906,13                                             DHAR4730
  906 IFERR=0                                                           DHAR4740
      NT=2**MT                                                          DHAR4770
      NTV2=NT/2                                                         DHAR4780
C                                                                       DHAR4790
C     SET UP SIN TABLE                                                  DHAR4800
C     THETA=PIE/2**(L+1) FOR L=1                                        DHAR4810
  910 THETA=.7853981633974483                                           DHAR4820
C                                                                       DHAR4830
C     JSTEP=2**(MT-L+1) FOR L=1                                         DHAR4840
      JSTEP=NT                                                          DHAR4850
C                                                                       DHAR4860
C     JDIF=2**(MT-L) FOR L=1                                            DHAR4870
      JDIF=NTV2                                                         DHAR4880
      S(JDIF)=SIN(THETA)                                                DHAR4890
      DO 950 L=2,MT                                                     DHAR4900
      THETA=THETA/2.000                                                 DHAR4910
      JSTEP2=JSTEP                                                      DHAR4920
      JSTEP=JDIF                                                        DHAR4930
      JDIF=JSTEP/2                                                      DHAR4940
      S(JDIF)=SIN(THETA)                                                DHAR4950
                                                                        DHAR4960
```

```
      JC1=NT-JCIF                                                       DHAR4970
      S(JC1)=CCCS(THETA)                                                DHAR4980
      JLAST=NT-JSTEP2                                                   DHAR4990
      IF(JLAST - JSTEP) 950,920,920                                     DHAR5000
  920 DO 940 J=JSTEP,JLAST,JSTEP                                        DHAR5010
      JC=NT-J                                                           DHAR5020
      JC=J+JCIF                                                         DHAR5030
  940 S(JC)=S(J)+S(JC1)+S(JCIF)+S(JC)                                   DHAR5040
  950 CONTINUE                                                          DHAR5050
C                                                                       DHAR5060
C     SET UP INV(J) TABLE                                               DHAR5070
C                                                                       DHAR5080
  960 NTLEXP=NTV2                                                       DHAR5090
C                                                                       DHAR5100
C     NTLEXP=2**(NT-L). FOR L=1                                         DHAR5110
      LM1EXP=1                                                          DHAR5120
C                                                                       DHAR5130
C     LM1EXP=2**(L-1). FOR L=1                                          DHAR5140
      INV(1)=0                                                          DHAR5150
      DO 980 L=1,NT                                                     DHAR5160
      INV(LM1EXP+1) = NTLEXP                                            DHAR5170
      DO 970 J=2,LM1EXP                                                 DHAR5180
      JJ=J+LM1EXP                                                       DHAR5190
  970 INV(JJ)=INV(J)+NTLEXP                                             DHAR5200
      NTLEXP=NTLEXP/2                                                   DHAR5210
  980 LM1EXP=LM1EXP*2                                                   DHAR5220
  982 IF(IFSET)12,995,12                                                DHAR5230
      END                                                               DHAR5240
```

Subroutines RHARM and DRHARM

Given 2N real numbers $X_0, X_1, \ldots, X_{2N-1}$, the subroutine computes Fourier coefficients $a_0, a_1, b_1, a_2, b_2, \ldots, a_{N-1}, b_{N-1}, a_N$ in the equation:

$$X_j = \frac{1}{2} a_0 + \sum_{k=1}^{N-1} \left(a_k \cos\left(\frac{\pi \cdot j \cdot k}{N}\right) + b_k \sin\left(\frac{\pi \cdot j \cdot k}{N}\right)\right) + \frac{1}{2} a_N (-1)^j$$

where

$$j = 0, 1, \ldots, 2N-1$$

The subroutine HARM is called to compute the complex coefficients:

$$A_k = \frac{1}{N} \sum_{j=0}^{N-1} (X_{2j} - iX_{2j+1})\, e^{\frac{2\pi i}{N} \cdot j \cdot k},$$

$$k = 0, 1, 2, \ldots, N-1$$

Then for $k = 1, 2, \ldots, \frac{N}{2} - 1, \frac{N}{2}$ (the bar is conjugation):

$$A'_k = \frac{1}{2} (\bar{A}_k + A_{N-k})$$

$$A''_k = \frac{1}{2} (\bar{A}_{N-k} - A_k)$$

and:

$$C_k = \frac{1}{2} (A'_k + \bar{A}''_k\, e^{(\frac{\pi}{2} - \frac{\pi}{N} k)i})$$

$$\text{for } k = 1, 2, \ldots, \frac{N}{2} - 1, \frac{N}{2}$$

$$C_{N-k} = \frac{1}{2} (\bar{A}'_k - A''_k\, e^{(\frac{\pi}{N-k} - \frac{\pi}{2})i})$$

$$\text{for } k = 1, 2, \ldots, \frac{N}{2} - 1$$

Let:

$$C_o = \frac{1}{2} (\text{Re } A_o - \text{Im } A_o)$$

$$C_N = \frac{1}{2} (\text{Re } A_o + \text{Im } A_o)$$

Finally:

$$a_o = 2\ \mathrm{Re}\ (C_o)$$

$$a_k = 2\ \mathrm{Re}\ (C_k)$$

$$b_k = -2\ \mathrm{Im}\ (C_k)$$

$$a_N = 2\ \mathrm{Re}\ (C_N)$$

```
      AP2RE=A(K6)*CO+A(K6+1)*SI                                         RHAR1000
      AP2IM=-A(K6)*SI+A(K6+1)*CO                                        RHAR1010
      CIRE=.5*(A(2*I-1)+AP2RE)                                          RHAR1020
      CIIM=.5*(A(2*I)+AP2IM)                                            RHAR1030
      CNIRE=.5*(A(2*I-1)-AP2RE)                                         RHAR1040
      CNIIM=.5*(A(2*I)-AP2IM)                                           RHAR1050
      A(2*I-1)=CIRE                                                     RHAR1060
      A(2*I)=CIIM                                                       RHAR1070
      A(K6)=CNIRE                                                       RHAR1080
      A(K6+1)=-CNIIM                                                    RHAR1090
      SIS=SI                                                            RHAR1100
      SI=SI*SC+CO*SS                                                    RHAR1110
  116 CO=CO*SC-SIS*SS                                                   RHAR1120
C                                                                       RHAR1130
C     SHIFT C(J)S FOR J=N/2+1 TO J=N UP ONE SLOT                        RHAR1140
      DO 117 I=1,NTOT,2                                                 RHAR1150
      K8=NTOT+4+I                                                       RHAR1160
      A(K8-2)=A(K8)                                                     RHAR1170
  117 A(K8-1)=A(K8+1)                                                   RHAR1180
      DO 500 I=3,NTOT2,2                                                RHAR1190
      A(I) = 2. * A(I)                                                  RHAR1200
  500 A(I + 1) = -2. * A(I + 1)                                         RHAR1210
      RETURN                                                            RHAR1220
      END                                                               RHAR1230
```

```
                                                                        RHAR  10
 ......................................................................RHAR  20
                                                                        RHAR  30
   SUBROUTINE RHARM                                                     RHAR  40
                                                                        RHAR  50
   PURPOSE                                                              RHAR  60
      FINDS THE FOURIER COEFFICIENTS OF ONE DIMENSIONAL REAL DATA RHAR  70
                                                                        RHAR  80
   USAGE                                                                RHAR  90
      CALL RHARM (A,M,INV,S,IFERR)                                      RHAR 100
                                                                        RHAR 110
   DESCRIPTION OF PARAMETERS                                            RHAR 120
      A      - AS INPUT, CONTAINS ONE DIMENSIONAL REAL DATA. A IS RHAR 130
               2*N+4 CORE LOCATIONS, WHERE N = 2**M. 2*N REAL RHAR 140
               NUMBERS ARE PUT INTO THE FIRST 2*N CORE LOCATIONS RHAR 150
               OF A                                                     RHAR 160
               AS OUTPUT, A CONTAINS THE FOURIER COEFFICIENTS RHAR 170
               A0/2,B0=0,A1,B1,A2,B2,...,AN/2,BN=0 RESPECTIVELY IN RHAR 180
               THE FIRST 2N+2 CORE LOCATIONS OF A                       RHAR 190
      M      - AN INTEGER WHICH DETERMINES THE SIZE OF THE VECTOR RHAR 200
               A. THE SIZE OF A IS 2*(2**M) + 4                         RHAR 210
      INV    - A VECTOR WORK AREA FOR BIT AND INDEX MANIPULATION OFRHAR 220
               DIMENSION ONE EIGHTH THE NUMBER OF REAL INPUT, VIZ.,RHAR 230
               (1/8)*2*(2**M)                                           RHAR 240
      S      - A VECTOR WORK AREA FOR SINE TABLES WITH DIMENSION RHAR 250
               THE SAME AS INV                                          RHAR 260
      IFERR  - A RETURNED VALUE OF 1 MEANS THAT M IS LESS THAN 3 ORRHAR 270
               GREATER THAN 20. OTHERWISE IFERR IS SET = 0              RHAR 280
                                                                        RHAR 290
   REMARKS                                                              RHAR 300
      THIS SUBROUTINE GIVES THE FOURIER COEFFICIENTS OF 2*(2**M) RHAR 310
      REAL POINTS. SEE SUBROUTINE HARM FOR THREE DIMENSIONAL,           RHAR 320
      COMPLEX FOURIER TRANSFORMS                                        RHAR 330
                                                                        RHAR 340
   SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                        RHAR 350
      HARM                                                              RHAR 360
                                                                        RHAR 370
   METHOD                                                               RHAR 380
      THE FOURIER COEFFICIENTS A0,B0=0,A1,B1,...,AN,BN=0 ARE RHAR 390
      OBTAINED FOR INPUT XJ, J=0,1,2,...,2N-1 FOR THE FOLLOWING RHAR 400
      EQUATION (PI = 3.14159...)                                        RHAR 410
                                                                        RHAR 420
            N-1                                                   J     RHAR 430
XJ=(1/2)A0+SUM (AK*COS(PI*J*K/N)+BK*SIN(PI*J*K/N))+(1/2)AN(-1)          RHAR 440
            K=1                                                         RHAR 450
                                                                        RHAR 460
   SEE REFERENCE UNDER SUBROUTINE HARM                                  RHAR 470
                                                                        RHAR 480
 ......................................................................RHAR 490
                                                                        RHAR 500
   SUBROUTINE RHARM(A,M,INV,S,IFERR)                                    RHAR 510
   DIMENSION A(1),L(3),INV(1),S(1)                                      RHAR 520
   IFSET=1                                                              RHAR 530
   L(1)=M                                                               RHAR 540
   L(2)=0                                                               RHAR 550
   L(3)=0                                                               RHAR 560
   NTOT=2**M                                                            RHAR 570
   NTOT2 = 2*NTOT                                                       RHAR 580
   FN = NTOT                                                            RHAR 590
   DO   3 I = 2,NTOT2,2                                                 RHAR 600
 3 A(I) = -A(I)                                                         RHAR 610
   DO   6 I = 1,NTOT2                                                   RHAR 620
 6 A(I) = A(I)/FN                                                       RHAR 630
   CALL HARM(A,L,INV,S ,IFSET,IFERR)                                    RHAR 640
                                                                        RHAR 650
   MOVE LAST HALF OF A(J)S DOWN ONE SLOT AND ADD A(N) AT BOTTOM TO RHAR 660
   GIVE ARRAY FOR A1PRIME AND A2PRIME CALCULATION                       RHAR 670
                                                                        RHAR 680
21 DO  52 I=1,NTOT,2                                                    RHAR 690
   JO=NTOT2+2-I                                                         RHAR 700
   A(JO)=A(JO-2)                                                        RHAR 710
52 A(JO+1)=A(JO-1)                                                      RHAR 720
   A(NTOT2+3)=A(1)                                                      RHAR 730
   A(NTOT2+4)=A(2)                                                      RHAR 740
                                                                        RHAR 750
   CALCULATE A1PRIMES AND STORE IN FIRST N SLOTS                        RHAR 760
   CALCULATE A2PRIMES AND STORE IN SECOND N SLOTS IN REVERSE ORDER RHAR 770
   KO=NTOT+1                                                            RHAR 780
   DO 104 I=1,KO,2                                                      RHAR 790
   K1=NTOT2-I+4                                                         RHAR 800
   AP1RE=.5*(A(I)+A(K1))                                                RHAR 810
   AP2RE=-.5*(A(I+1)+A(K1+1))                                           RHAR 820
   AP1IM=.5*(-A(I+1)+A(K1+1))                                           RHAR 830
   AP2IM=-.5*(A(I)-A(K1))                                               RHAR 840
   A(I)=AP1RE                                                           RHAR 850
   A(I+1)=AP1IM                                                         RHAR 860
   A(K1)=AP2RE                                                          RHAR 870
104 A(K1+1)=AP2IM                                                       RHAR 880
   NTO = NTOT/2                                                         RHAR 890
110 NT=NTO+1                                                            RHAR 900
   DEL=3.1415927/FLOAT(NTOT)                                            RHAR 910
   SS=SIN(DEL)                                                          RHAR 920
   SC=COS(DEL)                                                          RHAR 930
   SI=0.0                                                               RHAR 940
   CO=1.0                                                               RHAR 950
                                                                        RHAR 960
   COMPUTE C(J)S FOR J=0 THRU J=N                                       RHAR 970
114 DO 116 I=1,NT                                                       RHAR 980
   K6=NTOT2-2*I+5                                                       RHAR 990
```

```
C                                                                       DRAR  10
C     ......................................................................DRAR  20
C                                                                       DRAR  30
C        SUBROUTINE DRHARM                                              DRAR  40
C                                                                       DRAR  50
C        PURPOSE                                                        DRAR  60
C           FINDS THE FOURIER COEFFICIENTS OF ONE DIMENSIONAL DOUBLE    DRAR  70
C           PRECISION REAL DATA                                         DRAR  80
C                                                                       DRAR  90
C        USAGE                                                          DRAR 100
C           CALL DRHARM(A,M,INV,S,IFERR)                                DRAR 110
C                                                                       DRAR 120
C        DESCRIPTION OF PARAMETERS                                      DRAR 130
C           A      - A DOUBLE PRECISION VECTOR                          DRAR 140
C                    AS INPUT, CONTAINS ONE DIMENSIONAL REAL DATA. A IS DRAR 150
C                    2*N+4 CORE LOCATIONS, WHERE N = 2**M. 2*N REAL     DRAR 160
C                    NUMBERS ARE PUT INTO THE FIRST 2*N CORE LOCATIONS DRAR 170
C                    OF A                                               DRAR 180
C                    AS OUTPUT, A CONTAINS THE FOURIER COEFFICIENTS     DRAR 190
C                    A0/2,B0=0,A1,B1,A2,B2,...,AN/2,BN=0 RESPECTIVELY IN DRAR 200
C                    THE FIRST 2N+2 CORE LOCATIONS OF A                 DRAR 210
C           M      - AN INTEGER WHICH DETERMINES THE SIZE OF THE VECTOR DRAR 220
C                    A. THE SIZE OF A IS 2*(2**M) + 4                   DRAR 230
C           INV    - A VECTOR WORK AREA FOR BIT AND INDEX MANIPULATION OFDRAR 240
C                    DIMENSION ONE EIGHTH THE NUMBER OF REAL INPUT, VIZ.,DRAR 250
C                    (1/8)*2*(2**M)                                     DRAR 260
C           S      - A DOUBLE PRECISION VECTOR WORK AREA FOR SINE TABLES DRAR 270
C                    WITH DIMENSION THE SAME AS INV                     DRAR 280
C           IFERR  - A RETURNED VALUE OF 1 MEANS THAT M IS LESS THAN 3 ORDRAR 290
C                    GREATER THAN 20. OTHERWISE IFERR IS SET = 0        DRAR 300
C                                                                       DRAR 310
C        REMARKS                                                        DRAR 320
C           THIS SUBROUTINE GIVES THE FOURIER COEFFICIENTS OF 2*(2**M) DRAR 330
C           REAL POINTS. SEE SUBROUTINE DHARM FOR THREE DIMENSIONAL,    DRAR 340
C           DOUBLE PRECISION, COMPLEX FOURIER TRANSFORMS.               DRAR 350
C                                                                       DRAR 360
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DRAR 370
C           DHARM                                                       DRAR 380
C                                                                       DRAR 390
C        METHOD                                                         DRAR 400
C           THE FOURIER COEFFICIENTS A0,B0=0,A1,B1,...,AN,BN=0 ARE      DRAR 410
C           OBTAINED FOR INPUT XJ, J=0,1,2,...,2N-1 FOR THE FOLLOWING   DRAR 420
C           EQUATION (PI = 3.14159...)                                  DRAR 430
C                                                                       DRAR 440
C                 N-1                                             J     DRAR 450
C     XJ=(1/2)A0+SUM (AK*COS(PI*J*K/N)+BK*SIN(PI*J*K/N))+(1/2)AN(-1)    DRAR 460
C                 K=1                                                   DRAR 470
C                                                                       DRAR 480
C           SEE REFERENCE UNDER SUBROUTINE DHARM                        DRAR 490
C                                                                       DRAR 500
C     ......................................................................DRAR 510
C                                                                       DRAR 520
      SUBROUTINE DRHARM(A,M,INV,S,IFERR)                                DRAR 530
      DIMENSION A(1),L(3),INV(1),S(1)                                   DRAR 540
      DOUBLE PRECISION A,SI,AP1IM,FN,CO,CIRE,AP2IM,S,SS,DEL,CIIM,AP1RE, DRAR 550
     1 CNIRE,SC,SIS,AP2RE,CNIIM                                         DRAR 560
      IFSET=1                                                           DRAR 570
      L(1)=M                                                            DRAR 580
      L(2)=0                                                            DRAR 590
      L(3)=0                                                            DRAR 600
      NTOT=2**M                                                         DRAR 610
      NTOT2 = 2*NTOT                                                    DRAR 620
      FN = NTOT                                                         DRAR 630
      DO   3 I = 2,NTOT2,2                                              DRAR 640
    3 A(I) = -A(I)                                                      DRAR 650
      DO   6 I = 1,NTOT2                                                DRAR 660
    6 A(I) = A(I)/FN                                                    DRAR 670
      CALL DHARM(A,L,INV,S,IFSET,IFERR)                                 DRAR 680
C                                                                       DRAR 690
C     MOVE LAST HALF OF A(J)S DOWN ONE SLOT AND ADD A(N) AT BOTTOM TO   DRAR 700
C     GIVE ARRAY FOR A1PRIME AND A2PRIME CALCULATION                    DRAR 710
C                                                                       DRAR 720
   21 DO  52 I=1,NTOT,2                                                 DRAR 730
      JO=NTOT2+2-I                                                      DRAR 740
      A(JO)=A(JO-2)                                                     DRAR 750
   52 A(JO+1)=A(JO-1)                                                   DRAR 760
      A(NTOT2+3)=A(1)                                                   DRAR 770
      A(NTOT2+4)=A(2)                                                   DRAR 780
C                                                                       DRAR 790
C     CALCULATE A1PRIMES AND STORE IN FIRST N SLOTS                     DRAR 800
C     CALCULATE A2PRIMES AND STORE IN SECOND N SLOTS IN REVERSE ORDER   DRAR 810
      KO=NTOT+1                                                         DRAR 820
      DO 104 I=1,KO,2                                                   DRAR 830
      K1=NTOT2-I+4                                                      DRAR 840
      AP1RE=.5*(A(I)+A(K1))                                             DRAR 850
      AP2RE=-.5*(A(I+1)+A(K1+1))                                        DRAR 860
      AP1IM=.5*(-A(I+1)+A(K1+1))                                        DRAR 870
      AP2IM=-.5*(A(I)-A(K1))                                            DRAR 880
      A(I)=AP1RE                                                        DRAR 890
      A(I+1)=AP1IM                                                      DRAR 900
      A(K1)=AP2RE                                                       DRAR 910
  104 A(K1+1)=AP2IM                                                     DRAR 920
      NTO = NTOT/2                                                      DRAR 930
  110 NT=NTO+1                                                          DRAR 940
      DEL=3.1415926535897 93/DFLOAT(NTOT)                               DRAR 950
      SS=DSIN(DEL)                                                      DRAR 960
      SC=DCOS(DEL)                                                      DRAR 970
      SI=0.0                                                            DRAR 980
      CO=1.0                                                            DRAR 990
```

```
C                                                                       DRAR1000
C     COMPUTE C(J)S FOR J=0 THRU J=N                                    DRAR1010
  114 DO 116 I=1,NT                                                     DRAR1020
      K6=NTOT2-2*I+5                                                    DRAR1030
      AP2RE=A(K6)*CO+A(K6+1)*SI                                         DRAR1040
      AP2IM=-A(K6)*SI+A(K6+1)*CO                                        DRAR1050
      CIRE=.5*(A(2*I-1)+AP2RE)                                          DRAR1060
      CIIM=.5*(A(2*I)+AP2IM)                                            DRAR1070
      CNIRE=.5*(A(2*I-1)-AP2RE)                                         DRAR1080
      CNIIM=.5*(A(2*I)-AP2IM)                                           DRAR1090
      A(2*I-1)=CIRE                                                     DRAR1100
      A(2*I)=CIIM                                                       DRAR1110
      A(K6)=CNIRE                                                       DRAR1120
      A(K6+1)=-CNIIM                                                    DRAR1130
      SIS=SI                                                            DRAR1140
      SI=SI*SC+CO*SS                                                    DRAR1150
  116 CO=CO*SC-SIS*SS                                                   DRAR1160
C                                                                       DRAR1170
C     SHIFT C(J)S FOR J=N/2+1 TO J=N UP ONE SLOT                        DRAR1180
      DO 117 I=1,NTOT,2                                                 DRAR1190
      K8=NTOT+4+I                                                       DRAR1200
      A(K8-2)=A(K8)                                                     DRAR1210
  117 A(K8-1)=A(K8+1)                                                   DRAR1220
      DO 500 I=3,NTOT2,2                                                DRAR1230
      A(I) = 2. * A(I)                                                  DRAR1240
  500 A(I + 1) = -2. * A(I + 1)                                         DRAR1250
      RETURN                                                            DRAR1260
      END                                                               DRAR1270
```

Subroutines APMM and DAPMM

These subroutines determine a linear combination of M linearly independent given continuous real functions, which constitute the Chebyshev approximation of a given real function over a discrete range.

1. Mathematical background

Let us consider a set of discrete data $\{x_k, f_k = f(x_k)\}$, $k = 1, \ldots, N$, which defines a real valued function f(x) over a discrete range, and M real continuous given functions $\{\varphi_i(x)\}$, $i = 1, \ldots, M$, on an interval containing $[x_1, x_N]$.

The problem is to determine a set of M real coefficients $\{a_i\}$ that minimizes

$$w = \operatorname*{Max}_{1 \le k \le N} \left| \varphi(x_k) - f_k \right|, \text{ where } \varphi(x) = \sum_{i=1}^{M} a_i \varphi_i(x)$$

This problem can be expressed in terms of linear programming. At first, let us shift the unknowns so that the resulting ones are nonnegative:

$$b_i = a_i + b_{M+1}, \quad i = 1, \ldots, M,$$

where

$$b_{M+1} = -\operatorname{Min}[0, a_1, \ldots, a_M]$$

At each point, the error of approximation is given by

$$e_k = \varphi(x_k) - f_k$$

or

$$e_k = \sum_{i=1}^{M+1} b_i \varphi_i(x_k) - f_k$$

where

$$\varphi_{M+1}(x) = -\sum_{i=1}^{M} \varphi_i(x)$$

By definition

$$w = \operatorname*{Max}_{1 \le k \le N} \left| e_k \right|$$

If we put $\varphi_{ik} = \varphi_i(x_k)$, the original problem can be expressed as: Minimize w, subject to:

$$w + \sum_{i=1}^{M+1} \varphi_{ik} b_i \geq f_k$$

$$w - \sum_{i=1}^{M+1} \varphi_{ik} b_i \geq f_k$$

$$b_i \geq 0 \text{ for } i = 1, \ldots, M+1$$

$$w \geq 0$$

In practice, it is more convenient to solve the following dual problem in the nonnegative variables $\{s_k, t_k\}$, $k = 1, \ldots, N$:

$$\text{Maximize} \quad \sum_{k=1}^{N} f_k (s_k - t_k), \quad \text{subject to:}$$

$$\sum_{k=1}^{N} (s_k + t_k) \leq 1$$

$$\sum_{k=1}^{N} \varphi_{ik} (s_k - t_k) \leq 0, \; i = 1, \ldots, M+1$$

$$s_k \geq 0, \; t_k \geq 0, \text{ for } k = 1, \ldots, N$$

Using the variables of the dual problem as slacks, these n + 2 constraints can be written as equalities:

$$w + \sum_{k=1}^{N} (s_k + t_k) = 1$$

$$b_i + \sum_{k=1}^{N} \varphi_{ik} (s_k - t_k) = 0$$

The dual problem is solved by means of a revised simplex method (inverse matrix algorithm) using the following initial simplex tableau where w, $b_1, \ldots, b_{M+1}$, are basic variables:

Note that the entries in the transformed top row under w, $b_1, \ldots, b_{M+1}$ will be the optimal solution to the primal, that is, w and the b_i's.

A condensed tableau is used for calculation. The correspondence between the subscripts of the full tableau and these of the condensed one is given by means of the index vector IHE.

When the procedure has reached the optimum, the top row contains on return:

w	Maximum error of approximation
b_i, $i = 1, \ldots, M+1$	Coefficients from which we can get the unknowns by $a_i = b_i - b_{M+1}$ for $i = 1, \ldots, M$
s_k, $k = 1, \ldots, N$	The error at the given points, according to $e_k = s_k - w$

	Variables	$s_1 \ldots s_N$	$t_1 \ldots t_N$	w	b_1	b_2	$\ldots$	b_{M+1}
Top Row	Maximize	$f_1 \ldots f_N$	$-f_1 \ldots -f_N$	0	0	0	$\ldots$	0
	1	$= 1 \ldots 1$	$1 \ldots 1$	1	0	0	$\ldots$	0
	0	$= \varphi_{11} \ldots \varphi_{1N}$	$-\varphi_{11} \ldots -\varphi_{1N}$	0	1	0	$\ldots$	0
	0	$= \varphi_{21} \ldots \varphi_{2N}$	$-\varphi_{21} \ldots -\varphi_{2N}$	0	0	1	$\ldots$	0
	0	$= \varphi_{M+1,1} \; \varphi_{M+1,N}$	$-\varphi_{M+1,1} \; -\varphi_{M+1,N}$		0	0	$\ldots$	1

2. Programming considerations

For economy of storage requirements, the full simplex tableau is not stored, but only the top row at any iteration step. Therefore, we must update an inverse matrix T which allows us to generate the needed pivot column at any iteration step from the corresponding initial column. The procedure terminates for the following reasons:

a. An optimal top row is reached, indicated by the resultant error parameter IER, which is set to zero. All the relevant data are contained in this top row. The corresponding subscripts are stored in a part of the integer vector IHE.

b. No suitable pivot can be found. The resultant error parameter IER is set to -1.

c. The number of iterations ITER has reached the internal maximum N+M. The error parameter IER is set to 1 as a warning.

The subroutines APMM and DAPMM contain two tests for loss of significance:

a. The first test is made when an initial column is multiplied by the rows of matrix T. The corresponding result DSUM is set equal to zero if the following inequality is valid:

$$\left|\mathrm{DSUM}\right| \leqq \mathrm{HELP} \quad * \quad \mathrm{EPS}$$

where HELP is the absolutely greatest intermediate sum formed during calculation of the scalar product-sum DSUM, and EPS is an internal tolerance:

$$\mathrm{EPS} = \begin{cases} 10^{-5} & \text{in APMM} \\ 10^{-14} & \text{in DAPMM} \end{cases}$$

b. The second test is made when a new element in the top row TOP is generated. The new element $\overline{\mathrm{TOP}_i}$ is calculated from the previous value TOP_i by the addition

$$\overline{\mathrm{TOP}_i} = \mathrm{TOP}_i + \mathrm{DSUM}_i$$

The new value $\overline{\mathrm{TOP}_i}$ is replaced by zero if

$$\left|\mathrm{TOP}_i + \mathrm{DSUM}_i\right| \leqq \left|\mathrm{DSUM}_i\right| \quad * \quad \mathrm{EPS}$$

where EPS is the above-mentioned internal tolerance.

Some remarks are in order:

a. In cases where the given approximating functions are polynomials, APMM and DAPMM are more efficient when used with orthogonal polynomials than with ordinary polynomials. The number of iterations required usually decreases with orthogonal functions, and the magnitude of the roundoff errors may be significantly reduced. When using an ordinary polynomial approximating function it seems that the roundoff errors are due to the wide range of values of the elements $\varphi_{i,k}$ in the initial tableau.

b. If approximation by normal Chebyshev polynomials, shifted Chebyshev polynomials, Legendre polynomials, Laguerre polynomials, or Hermite polynomials is desired, the subroutines CNP, CSP, LEP, LAP, or HEP, respectively from the SSP (Scientific Subroutine Package) can be used as the external subroutine FCT.

For reference see:

1. I. Barrodale and A. Young, "Algorithms for Best L_1 and L_∞, Linear Approximations on a Discrete Set", Numerische Mathematik, vol. 8, iss. 3 (1966), pp. 295-306.

2. S. Vajda, Mathematical Programming, Addison-Wesley Publishing Company, Inc., London, 1961.


```
C                                                                       APMM  10
C     ..................................................................APMM  20
C                                                                       APMM  30
C        SUBROUTINE APMM                                                APMM  40
C                                                                       APMM  50
C        PURPOSE                                                        APMM  60
C           APPROXIMATE A FUNCTION TABULATED IN N POINTS BY ANY LINEAR  APMM  70
C           COMBINATION OF M GIVEN CONTINUOUS FUNCTIONS IN THE SENSE    APMM  80
C           OF CHEBYSHEV.                                               APMM  90
C                                                                       APMM 100
C        USAGE                                                          APMM 110
C           CALL APMM(FCT,N,M,TOP,IHE,PIV,T,ITER,IER)                   APMM 120
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT IN THE         APMM 130
C           CALLING PROGRAM.                                            APMM 140
C                                                                       APMM 150
C        DESCRIPTION OF PARAMETERS                                      APMM 160
C           FCT    - NAME OF SUBROUTINE TO BE SUPPLIED BY THE USER.     APMM 170
C                    IT COMPUTES VALUES OF M GIVEN FUNCTIONS FOR        APMM 180
C                    ARGUMENT VALUE X.                                  APMM 190
C                    USAGE                                              APMM 200
C                       CALL FCT(Y,X,K)                                 APMM 210
C                    DESCRIPTION OF PARAMETERS                          APMM 220
C                       Y    - RESULT VECTOR OF DIMENSION M CONTAINING  APMM 230
C                              THE VALUES OF GIVEN CONTINUOUS FUNCTIONS APMM 240
C                              FOR GIVEN ARGUMENT X                     APMM 250
C                       X    - ARGUMENT VALUE                           APMM 260
C                       K    - AN INTEGER VALUE WHICH IS EQUAL TO M-1   APMM 270
C                    REMARKS                                            APMM 280
C                       IF APPROXIMATION BY NORMAL CHEBYSHEV, SHIFTED   APMM 290
C                       CHEBYSHEV, LEGENDRE, LAGUERRE, HERMITE POLYNO-  APMM 300
C                       MIALS IS DESIRED SUBROUTINES CNP, CSP, LEP,     APMM 310
C                       LAP, HEP, RESPECTIVELY FROM SSP COULD BE USED.  APMM 320
C           N      - NUMBER OF DATA POINTS DEFINING THE FUNCTION WHICH  APMM 330
C                    IS TO BE APPROXIMATED                              APMM 340
C           M      - NUMBER OF GIVEN CONTINUOUS FUNCTIONS FROM WHICH    APMM 350
C                    THE APPROXIMATING FUNCTION IS CONSTRUCTED.         APMM 360
C           TOP    - VECTOR OF DIMENSION 3*N.                           APMM 370
C                    ON ENTRY IT MUST CONTAIN FROM TOP(1) UP TO TOP(N)  APMM 380
C                    THE GIVEN N FUNCTION VALUES AND FROM TOP(N+1) UP   APMM 390
C                    TO TOP(2*N) THE CORRESPONDING NODES                APMM 400
C                    ON RETURN TOP CONTAINS FROM TOP(1) UP TO TOP(N)    APMM 410
C                    THE ERRORS AT THOSE N NODES.                       APMM 420
C                    OTHER VALUES OF TOP ARE SCRATCH.                   APMM 430
C           IHE    - INTEGER VECTOR OF DIMENSION 3*M+4*N+6              APMM 440
C           PIV    - VECTOR OF DIMENSION 3*M+6.                         APMM 450
C                    ON RETURN PIV CONTAINS AT PIV(1) UP TO PIV(M) THE  APMM 460
C                    RESULTING COEFFICIENTS OF LINEAR APPROXIMATION.    APMM 470
C           T      - AUXILIARY VECTOR OF DIMENSION (M+2)*(M+2)          APMM 480
C           ITER   - RESULTANT INTEGER WHICH SPECIFIES THE NUMBER OF    APMM 490
C                    ITERATIONS NEEDED                                  APMM 500
C           IER    - RESULTANT ERROR PARAMETER CODED IN THE FOLLOWING   APMM 510
C                    FORM                                               APMM 520
C                       IER=0  - NO ERROR                               APMM 530
C                       IER=1  - THE NUMBER OF ITERATIONS HAS REACHED   APMM 540
C                                THE INTERNAL MAXIMUM N+M               APMM 550
C                       IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARA- APMM 560
C                                METER M OR N OR SINCE AT SOME ITERATION APMM 570
C                                NO SUITABLE PIVOT COULD BE FOUND       APMM 580
C                                                                       APMM 590
C        REMARKS                                                        APMM 600
C           NO ACTION BESIDES ERROR MESSAGE IN CASE M LESS THAN 1 OR    APMM 610
C           N LESS THAN 2.                                              APMM 620
C                                                                       APMM 630
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  APMM 640
C           THE EXTERNAL SUBROUTINE FCT MUST BE FURNISHED BY THE USER.  APMM 650
C                                                                       APMM 660
C        METHOD                                                         APMM 670
C           THE PROBLEM OF APPROXIMATION A TABULATED FUNCTION BY ANY    APMM 680
C           LINEAR COMBINATION OF GIVEN FUNCTIONS IN THE SENSE OF       APMM 690
C           CHEBYSHEV (I.E. TO MINIMIZE THE MAXIMUM ERROR) IS TRANS-    APMM 700
C           FORMED INTO A LINEAR PROGRAMMING PROBLEM. APMM USES A       APMM 710
C           REVISED SIMPLEX METHOD TO SOLVE A CORRESPONDING DUAL        APMM 720
C           PROBLEM. FOR REFERENCE, SEE                                 APMM 730
C           I.BARRODALE/A.YOUNG, ALGORITHMS FOR BEST L-SUB-ONE AND      APMM 740
```

```
C          L-SUB-INFINITY, LINEAR APPROXIMATIONS ON A DISCRETE SET,
C          NUMERISCHE MATHEMATIK, VOL.8, ISS.3 (1966), PP.295-306.
C
C     ..................................................................
C
      SUBROUTINE APMM(FCT,N,M,TOP,IHE,PIV,T,ITER,IER)
C
C
      DIMENSION TOP(1),IHE(1),PIV(1),T(1)
      DOUBLE PRECISION DSUM
C
C        TEST ON WRONG INPUT PARAMETERS N AND M
      IER=-1
      IF (N-1) 81,81,1
    1 IF(M) 81,81,2
C
C        INITIALIZE CHARACTERISTIC VECTORS FOR THE TABLEAU
    2 IER=0
C
C        PREPARE TOP-ROW TOP
      DO 3 I=1,N
      K=I+N
      J=K+N
      TOP(J)=TOP(K)
    3 TOP(K)=-TOP(I)
C
C        PREPARE INVERSE TRANSFORMATION MATRIX T
      L=N+2
      LL=L*L
      DO 4 I=1,LL
    4 T(I)=0.
      K=1
      J=L+1
      DO 5 I=1,L
      T(K)=1.
    5 K=K+J
C
C        PREPARE INDEX-VECTOR IHE
      DO 6 I=1,L
      K=I+L
      J=K+L
      IHE(I)=0
      IHE(K)=I
    6 IHE(J)=1-I
      NAN=N+N
      K=L+L+L
      J=K+NAN
      DO 7 I=1,NAN
      K=K+1
      IHE(K)=I
      J=J+1
    7 IHE(J)=I
C
C        SET COUNTER ITER FOR ITERATION-STEPS
      ITER=-1
    8 ITER=ITER+1
C
C        TEST FOR MAXIMUM ITERATION-STEPS
      IF(N+M-ITER) 9,9,10
    9 IER=1
      GO TO 69
C
C        DETERMINE THE COLUMN WITH THE MOST POSITIVE ELEMENT IN TOP
   10 ISE=0
      IPIV=0
      K=L+L+L
      SAVE=0.
C
C        START TOP-LOOP
      DO 14 I=1,NAN
      IDO=K+I
      HELP=TOP(I)
      IF(HELP-SAVE) 12,12,11
   11 SAVE=HELP
      IPIV=I
   12 IF(IHE(IDO)) 14,13,14
   13 ISE=I
   14 CONTINUE
C        END OF TOP-LOOP
C
C        IS OPTIMAL TABLEAU REACHED
      IF(IPIV) 69,69,15
C
C        DETERMINE THE PIVOT-ELEMENT FOR THE COLUMN CHOSEN ABOVE
   15 ILAB=1
      IND=0
      J=ISE
      IF(J) 21,21,34
C
C        TRANSFER K-TH COLUMN FROM T TO PIV
   16 K=(K-1)*L
      DO 17 I=1,L
      J=L+I
      K=K+1
   17 PIV(J)=T(K)
C
C        IS ANOTHER COLUMN NEEDED FOR SEARCH FOR PIVOT-ELEMENT
   18 IF(ISE) 22,22,19
   19 ISE=-ISE
C
C        TRANSFER COLUMNS IN PIV
      J=L+1
      IDO=L+L
      DO 20 I=J,IDO
      K=I+L
   20 PIV(K)=PIV(I)
   21 J=IPIV
      GO TO 34
C
C        SEARCH PIVOT-ELEMENT PIV(IND)
   22 SAVE=1.E38
      IDO=0
      K=L+1
      LL=L+L
      IND=0
C
C        START PIVOT-LOOP
      DO 29 I=K,LL
      J=I+L
      HELP=PIV(I)
      IF(HELP) 29,29,23
   23 HELP=-HELP
      IF(ISE) 26,24,26
   24 IF(IHE(J)) 27,25,27
   25 IDO=I
      GO TO 29
   26 HELP=-PIV(J)/HELP
   27 IF(HELP-SAVE) 28,29,29
   28 SAVE=HELP
      IND=I
   29 CONTINUE
C        END OF PIVOT-LOOP
C
C        TEST FOR SUITABLE PIVOT-ELEMENT
      IF(IND) 30,30,32
   30 IF(IDO) 68,68,31
   31 IND=IDO
C        PIVOT-ELEMENT IS STORED IN PIV(IND)
C
C        COMPUTE THE RECIPROCAL OF THE PIVOT-ELEMENT REPI
   32 REPI=1./PIV(IND)
      IND=IND-L
C
C        UPDATE THE TOP-ROW TOP OF THE TABLEAU
      ILAB=0
      SAVE=-TOP(IPIV)*REPI
      TOP(IPIV)=SAVE
C
C        INITIALIZE J AS COUNTER FOR TOP-LOOP
      J=NAN
   33 IF(J-IPIV) 34,53,34
   34 K=0
C
C        SEARCH COLUMN IN TRANSFORMATION-MATRIX T
      DO 36 I=1,L
      IF(IHE(I)-J) 36,35,36
   35 K=I
      IF(ILAB) 50,50,16
   36 CONTINUE
C
C        GENERATE COLUMN USING SUBROUTINE FCT AND TRANSFORMATION-MATRIX
      I=L+L+L+NAN+J
      I=IHE(I)-N
      IF(I) 37,37,38
   37 I=I+N
      K=1
   38 I=I+NAN
C
C        CALL SUBROUTINE FCT
      CALL FCT(PIV,TOP(I),M-1)
C
C        PREPARE THE CALLED VECTOR PIV
      DSUM=0.D0
      IDO=M
      DO 41 I=1,M
      HELP=PIV(IDO)
      IF(K) 39,39,40
   39 HELP=-HELP
   40 DSUM=DSUM+DBLE(HELP)
      PIV(IDO+1)=HELP
   41 IDO=IDO-1
      PIV(L)=-DSUM
      PIV(1)=1.
C
C        TRANSFORM VECTOR PIV WITH ROWS OF MATRIX T
      IDO=IND
      IF(ILAB) 44,44,42
   42 K=1
   43 IDO=K
   44 DSUM=0.D0
      HELP=0.
C
C        START MULTIPLICATION-LOOP
      DO 46 I=1,L
      DSUM=DSUM+DBLE(PIV(I)*T(IDO))
      TOL=ABS(SNGL(DSUM))
      IF(TOL-HELP) 46,46,45
   45 HELP=TOL
   46 IDO=IDO+L
C        END OF MULTIPLICATION-LOOP
C
      TOL=1.E-5*HELP
      IF(ABS(SNGL(DSUM))-TOL) 47,47,48
   47 DSUM=0.D0
   48 IF(ILAB) 51,51,49
   49 I=K+L
      PIV(I)=DSUM
C
C        TEST FOR LAST COLUMN-TERM
      K=K+1
      IF(K-L) 43,43,18
   50 I=(K-1)*L+IND
      DSUM=T(I)
C
C        COMPUTE NEW TOP-ELEMENT
   51 DSUM=DSUM*DBLE(SAVE)
      TOL=1.E-5*ABS(SNGL(DSUM))
      TOP(J)=TOP(J)+SNGL(DSUM)
      IF(ABS(TOP(J))-TOL) 52,52,53
   52 TOP(J)=0.
C
C        TEST FOR LAST TOP-TERM
   53 J=J-1
      IF(J) 54,54,33
C        END OF TOP-LOOP
C
C        TRANSFORM PIVOT-COLUMN
   54 I=IND+L
      PIV(I)=-1.
      DO 55 I=1,L
      J=I+L
   55 PIV(I)=-PIV(J)*REPI
C
C        UPDATE TRANSFORMATION-MATRIX T
      J=0
      DO 57 I=1,L
      IDO=J+IND
      SAVE=T(IDO)
      T(IDO)=0.
      DO 56 K=1,L
      ISE=K+J
   56 T(ISE)=T(ISE)+SAVE*PIV(K)
   57 J=J+L
C
C        UPDATE INDEX-VECTOR IHE
C        INITIALIZE CHARACTERISTICS
      J=0
      K=0
      ISE=0
      IDO=0
C
C        START QUESTION-LOOP
      DO 61 I=1,L
      LL=I+L
      ILAB=IHE(LL)
      IF(IHE(I)-IPIV) 59,58,59
   58 ISE=I
      J=ILAB
```

```
   59 IF(ILAB-IND) 61,60,61
   60 IDO=I
      K=IHE(I)
   61 CONTINUE
C        END OF QUESTION-LOOP
C
C        START MODIFICATION
      IF(K) 62,62,63
   62 IHE(IDO)=IPIV
      IF(ISE) 67,67,65
   63 IF(IND-J) 64,66,64
   64 LL=L+L+L+NAN
      K=K+LL
      I=IPIV+LL
      ILAB=IHE(K)
      IHE(K)=IHE(I)
      IHE(I)=ILAB
      IF(ISE) 67,67,65
   65 IDO=IDO+L
      I=ISE+L
      IHE(IDO)=J
      IHE(I)=IND
   66 IHE(ISE)=0
   67 LL=L+L
      J=LL+IND
      I=LL+L+IPIV
      ILAB=IHE(I)
      IHE(I)=IHE(J)
      IHE(J)=ILAB
C        END OF MODIFICATION
C
      GO TO 8
C
C        SET ERROR PARAMETER IER=-1 SINCE NO SUITABLE PIVOT IS FOUND
   68 IER=-1
C
C        EVALUATE FINAL TABLEAU
C        COMPUTE SAVE AS MAXIMUM ERROR OF APPROXIMATION AND
C        HELP AS ADDITIVE CONSTANCE FOR RESULTING COEFFICIENTS
   69 SAVE=0.
      HELP=0.
      K=L+L+L
      DO 73 I=1,NAN
      IDO=K+I
      J=IHE(IDO)
      IF(J) 71,70,73
   70 SAVE=-TOP(I)
   71 IF(M+J+1) 73,72,73
   72 HELP=TOP(I)
   73 CONTINUE
C
C        PREPARE T,TOP,PIV
      T(1)=SAVE
      IDO=NAN+1
      J=NAN+N
      DO 74 I=IDO,J
   74 TOP(I)=SAVE
      DO 75 I=1,M
   75 PIV(I)=HELP
C
C        COMPUTE COEFFICIENTS OF RESULTING POLYNOMIAL IN PIV(1) UP TO PI
C        AND CALCULATE ERRORS AT GIVEN NODES IN TOP(1) UP TO TOP(N)
      DO 79 I=1,NAN
      IDO=K+I
      J=IHE(IDO)
      IF(J) 76,79,77
   76 J=-J
      PIV(J)=HELP-TOP(I)
      GO TO 79
   77 IF(J-N) 78,78,79
   78 J=J+NAN
      TOP(J)=SAVE+TOP(I)
   79 CONTINUE
      DO 80 I=1,N
      IDO=NAN+I
   80 TOP(I)=TOP(IDO)
   81 RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DAPMM
C
C        PURPOSE
C           APPROXIMATE A FUNCTION TABULATED IN N POINTS BY ANY LINEAR
C           COMBINATION OF M GIVEN CONTINUOUS FUNCTIONS IN THE SENSE
C           OF CHEBYSHEV.
C
C        USAGE
C           CALL DAPMM(FCT,N,M,TOP,IHE,PIV,T,ITER,IER)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT IN THE
C           CALLING PROGRAM.
C
C        DESCRIPTION OF PARAMETERS
C           FCT     - NAME OF SUBROUTINE TO BE SUPPLIED BY THE USER.
C                     IT COMPUTES VALUES OF M GIVEN FUNCTIONS FOR
C                     ARGUMENT VALUE X.
C                     USAGE
C                        CALL FCT(Y,X,K)
C                     DESCRIPTION OF PARAMETERS
C                        Y   - DOUBLE PRECISION RESULT VECTOR OF DIMEN-
C                              SION M CONTAINING THE VALUES OF GIVEN
C                              CONTINUOUS FUNCTIONS FOR GIVEN ARGUMENT X
C                        X   - DOUBLE PRECISON ARGUMENT VALUE
C                        K   - AN INTEGER VALUE WHICH IS EQUAL TO M-1
C                     REMARKS
C                        IF APPROXIMATION BY NORMAL CHEBYSHEV, SHIFTED
C                        CHEBYSHEV, LEGENDRE, LAGUERRE, HERMITE POLYNO-
C                        MIALS IS DESIRED SUBROUTINES DCNP,DCSP,DLEP,
C                        DLAP,DHEP, RESPECTIVELY FROM SSP COULD BE USED.
C           N       - NUMBER OF DATA POINTS DEFINING THE FUNCTION WHICH
C                     IS TO BE APPROXIMATED
C           M       - NUMBER OF GIVEN CONTINUOUS FUNCTIONS FROM WHICH
C                     THE APPROXIMATING FUNCTION IS CONSTRUCTED.
C           TOP     - DOUBLE PRECISION VECTOR OF DIMENSION 3*N.
C                     ON ENTRY IT MUST CONTAIN FROM TOP(1) UP TO TOP(N)
C                     THE GIVEN N FUNCTION VALUES AND FROM TOP(N+1) UP
C                     TO TOP(2*N) THE CORRESPONDING NODES
C                     ON RETURN TOP CONTAINS FROM TOP(1) UP TO TOP(N)
C                     THE ERRORS AT THOSE N NODES.
C                     OTHER VALUES OF TOP ARE SCRATCH.
C           IHE     - INTEGER VECTOR OF DIMENSION 3*M+4*N+6
C           PIV     - DOUBLE PRECISION VECTOR OF DIMENSION 3*M+6.
C                     ON RETURN PIV CONTAINS AT PIV(1) UP TO PIV(M) THE
C                     RESULTING COEFFICIENTS OF LINEAR APPROXIMATION.
C           T       - DOUBLE PRECISION AUXILIARY VECTOR OF DIMENSION
C                     (M+2)*(M+2)
C           ITER    - RESULTANT INTEGER WHICH SPECIFIES THE NUMBER OF
C                     ITERATIONS NEEDED
C           IER     - RESULTANT ERROR PARAMETER CODED IN THE FOLLOWING
C                     FORM
C                        IER=0  - NO ERROR
C                        IER=1  - THE NUMBER OF ITERATIONS HAS REACHED
C                                 THE INTERNAL MAXIMUM N*M
C                        IER=-1 - NO RESULT BECAUSE OF WRONG INPUT PARA-
C                                 METER M OR N OR SINCE AT SOME ITERATION
C                                 NO SUITABLE PIVOT COULD BE FOUND
C
C        REMARKS
C           NO ACTION BESIDES ERROR MESSAGE IN CASE M LESS THAN 1 OR
C           N LESS THAN 2.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL SUBROUTINE FCT MUST BE FURNISHED BY THE USER.
C
C        METHOD
C           THE PROBLEM OF APPROXIMATION A TABULATED FUNCTION BY ANY
C           LINEAR COMBINATION OF GIVEN FUNCTIONS IN THE SENSE OF
C           CHEBYSHEV (I.E. TO MINIMIZE THE MAXIMUM ERROR) IS TRANS-
C           FORMED INTO A LINEAR PROGRAMMING PROBLEM. DAPMM USES A
C           REVISED SIMPLEX METHOD TO SOLVE A CORRESPONDING DUAL
C           PROBLEM. FOR REFERENCE, SEE
C           I.BARRODALE/A.YOUNG, ALGORITHMS FOR BEST L-SUB-ONE AND
C           L-SUB-INFINITY, LINEAR APPROXIMATIONS ON A DISCRETE SET,
C           NUMERISCHE MATHEMATIK, VOL.8, ISS.3 (1966), PP.295-306.
C
C     ..................................................................
C
      SUBROUTINE DAPMM(FCT,N,M,TOP,IHE,PIV,T,ITER,IER)
C
C
      DIMENSION TOP(1),IHE(1),PIV(1),T(1)
      DOUBLE PRECISION DSUM,TOP,PIV,T,SAVE,HELP,REPI,TOL
C
C        TEST ON WRONG INPUT PARAMETERS N AND M
      IER=-1
      IF (N-1) 81,81,1
    1 IF(M) 81,81,2
C
C        INITIALIZE CHARACTERISTIC VECTORS FOR THE TABLEAU
    2 IER=0
C
C        PREPARE TOP-ROW TOP
      DO 3 I=1,N
      K=I+N
      J=K+N
      TOP(J)=TOP(K)
    3 TOP(K)=-TOP(I)
C
C        PREPARE INVERSE TRANSFORMATION MATRIX T
      L=M+2
      LL=L*L
      DO 4 I=1,LL
    4 T(I)=0.D0
      K=1
      J=L+1
      DO 5 I=1,L
      T(K)=1.D0
    5 K=K+J
C
C        PREPARE INDEX-VECTOR IHE
      DO 6 I=1,L
      K=I+L
      J=K+L
      IHE(I)=0
      IHE(K)=I
    6 IHE(J)=1-I
      NAN=N+N
      K=L+L+L
      J=K+NAN
      DO 7 I=1,NAN
      K=K+1
      IHE(K)=I
      J=J+1
    7 IHE(J)=I
C
C        SET COUNTER ITER FOR ITERATION-STEPS
      ITER=-1
    8 ITER=ITER+1
C
C        TEST FOR MAXIMUM ITERATION-STEPS
      IF(N+M-ITER) 9,9,10
    9 IER=1
      GO TO 69
C
C        DETERMINE THE COLUMN WITH THE MOST POSITIVE ELEMENT IN TOP
   10 ISE=0
      IPIV=0
      K=L+L+L
      SAVE=0.D0
C
C        START TOP-LOOP
      DO 14 I=1,NAN
      IDO=K+I
      HELP=TOP(I)
      IF(HELP-SAVE) 12,12,11
   11 SAVE=HELP
      IPIV=I
   12 IF(IHE(IDO)) 14,13,14
   13 ISE=I
   14 CONTINUE
C        END OF TOP-LOOP
C
C        IS OPTIMAL TABLEAU REACHED
      IF(IPIV) 69,69,15
C
C        DETERMINE THE PIVOT-ELEMENT FOR THE COLUMN CHOSEN UPOVE
   15 ILAB=1
      IND=0
      J=ISE
      IF(J) 21,21,34
C
C        TRANSFER K-TH COLUMN FROM T TO PIV
   16 K=(K-1)*L
      DO 17 I=1,L
      J=L+I
      K=K+1
   17 PIV(J)=T(K)
C
C        IS ANOTHER COLUMN NEEDED FOR SEARCH FOR PIVOT-ELEMENT
   18 IF(ISE) 22,22,19
```

```
   19 ISE=-ISE
C
C        TRANSFER COLUMNS IN PIV
      J=L+1
      IDO=L+L
      DO 20 I=J,IDO
      K=I+L
   20 PIV(K)=PIV(I)
   21 J=IPIV
      GO TO 34
C
C        SEARCH PIVOT-ELEMENT PIV(IND)
   22 SAVE=1.D38
      IDO=0
      K=L+1
      LL=L+L
      IND=0
C
C        START PIVOT-LOOP
      DO 29 I=K,LL
      J=I+L
      HELP=PIV(I)
      IF(HELP) 29,29,23
   23 HELP=-HELP
      IF(ISE) 26,24,26
   24 IF(IHE(J)) 27,25,27
   25 IDO=I
      GO TO 29
   26 HELP=-PIV(J)/HELP
   27 IF(HELP-SAVE) 28,29,29
   28 SAVE=HELP
      IND=I
   29 CONTINUE
C        END OF PIVOT-LOOP
C
C        TEST FOR SUITABLE PIVOT-ELEMENT
      IF(IND) 30,30,32
   30 IF(IDO) 68,68,31
   31 IND=IDO
C        PIVOT-ELEMENT IS STORED IN PIV(IND)
C
C        COMPUTE THE RECIPROCAL OF THE PIVOT-ELEMENT REPI
   32 REPI=1.D0/PIV(IND)
      IND=IND-L
C
C        UPDATE THE TOP-ROW TOP OF THE TABLEAU
      ILAB=0
      SAVE=-TOP(IPIV)*REPI
      TOP(IPIV)=SAVE
C
C        INITIALIZE J AS COUNTER FOR TOP-LOOP
      J=NAN
   33 IF(J-IPIV) 34,53,34
   34 K=0
C
C        SEARCH COLUMN IN TRANSFORMATION-MATRIX T
      DO 36 I=1,L
      IF(IHE(I)-J) 36,35,36
   35 K=I
      IF(ILAB) 50,50,16
   36 CONTINUE
C
C        GENERATE COLUMN USING SUBROUTINE FCT AND TRANSFORMATION-MATRIX
      I=L+L+L+NAN+J
      I=IHE(I)-N
      IF(I) 37,37,38
   37 I=I+N
      K=1
   38 I=I+NAN
C
C        CALL SUBROUTINE FCT
      CALL FCT(PIV,TOP(I),M-1)
C
C        PREPARE THE CALLED VECTOR PIV
      DSUM=0.D0
      IDO=M
      DO 41 I=1,M
      HELP=PIV(IDO)
      IF(K) 39,39,40
   39 HELP=-HELP
   40 DSUM=DSUM+HELP
      PIV(IDO+1)=HELP
   41 IDO=IDO-1
      PIV(L)=-DSUM
      PIV(1)=1.D0
C
C        TRANSFORM VECTOR PIV WITH ROWS OF MATRIX T
      IDO=IND
      IF(ILAB) 44,44,42
   42 K=1
   43 IDO=K
   44 DSUM=0.D0
      HELP=0.D0
C
C        START MULTIPLICATION-LOOP
      DO 46 I=1,L
      DSUM=DSUM+PIV(I)*T(IDO)
      TOL=DABS(DSUM)
      IF(TOL-HELP) 46,46,45
   45 HELP=TOL
   46 IDO=IDO+L
C        END OF MULTIPLICATION-LOOP
C
      TOL=1.D-14*HELP
      IF(DABS(DSUM)-TOL) 47,47,48
   47 DSUM=0.D0
   48 IF(ILAB) 51,51,49
   49 I=K+L
      PIV(I)=DSUM
C
C        TEST FOR LAST COLUMN-TERM
      K=K+1
      IF(K-L) 43,43,18
   50 I=(K-1)*L+IND
      DSUM=T(I)
C
C        COMPUTE NEW TOP-ELEMENT
   51 DSUM=DSUM*SAVE
      TOL=1.D-14*DABS(DSUM)
      TOP(J)=TOP(J)+DSUM
      IF(DABS(TOP(J))-TOL) 52,52,53
   52 TOP(J)=0.D0
C
C        TEST FOR LAST TOP-TERM
   53 J=J-1
      IF(J) 54,54,33
C        END OF TOP-LOOP
C
C        TRANSFORM PIVOT-COLUMN
   54 I=IND+L
      PIV(I)=-1.D0
      DO 55 I=1,L
      J=I+L
   55 PIV(I)=-PIV(J)*REPI
C
C        UPDATE TRANSFORMATION-MATRIX T
      J=0
      DO 57 I=1,L
      IDO=J+IND
      SAVE=T(IDO)
      T(IDO)=0.D0
      DO 56 K=1,L
      ISE=K+J
   56 T(ISE)=T(ISE)+SAVE*PIV(K)
   57 J=J+L
C
C        UPDATE INDEX-VECTOR IHE
C        INITIALIZE CHARACTERISTICS
      J=0
      K=0
      ISE=0
      IDO=0
C
C        START QUESTION-LOOP
      DO 61 I=1,L
      LL=I+L
      ILAB=IHE(LL)
      IF(IHE(I)-IPIV) 59,58,59
   58 ISE=I
      J=ILAB
   59 IF(ILAB-IND) 61,60,61
   60 IDO=I
      K=IHE(I)
   61 CONTINUE
C        END OF QUESTION-LOOP
C
C        START MODIFICATION
      IF(K) 62,62,63
   62 IHE(IDO)=IPIV
      IF(ISE) 67,67,65
   63 IF(IND-J) 64,66,64
   64 LL=L+L+L+NAN
      K=K+LL
      I=IPIV+LL
      ILAB=IHE(K)
      IHE(K)=IHE(I)
      IHE(I)=ILAB
      IF(ISE) 67,67,65
   65 IDO=IDO+L
      I=ISE+L
      IHE(IDO)=J
      IHE(I)=IND
   66 IHE(ISE)=0
   67 LL=L+L
      J=LL+IND
      I=LL+L+IPIV
      ILAB=IHE(I)
      IHE(I)=IHE(J)
      IHE(J)=ILAB
C        END OF MODIFICATION
C
      GO TO 8
C        SET ERROR PARAMETER IER=-1 SINCE NO SUITABLE PIVOT IS FOUND
   68 IER=-1
C
C        EVALUATE FINAL TABLEAU
C        COMPUTE SAVE AS MAXIMUM ERROR OF APPROXIMATION AND
C        HELP AS ADDITIVE CONSTANCE FOR RESULTING COEFFICIENTS
   69 SAVE=0.D0
      HELP=0.D0
      K=L+L
      DO 73 I=1,NAN
      IDO=K+I
      J=IHE(IDO)
      IF(J) 71,70,73
   70 SAVE=-TOP(I)
   71 IF(M+J+1) 73,72,73
   72 HELP=TOP(I)
   73 CONTINUE
C
C        PREPARE T,TOP,PIV
      T(1)=SAVE
      IDO=NAN+1
      J=NAN+N
      DO 74 I=IDO,J
   74 TOP(I)=SAVE
      DO 75 I=1,M
   75 PIV(I)=HELP
C
C        COMPUTE COEFFICIENTS OF RESULTING POLYNOMIAL IN PIV(I) UP TO PIV(M)
C        AND CALCULATE ERRORS AT GIVEN NODES IN TOP(I) UP TO TOP(N)
      DO 79 I=1,NAN
      IDO=K+I
      J=IHE(IDO)
      IF(J) 76,79,77
   76 J=-J
      PIV(J)=HELP-TOP(I)
      GO TO 79
   77 IF(J-N) 78,78,79
   78 J=J+NAN
      TOP(J)=SAVE+TOP(I)
   79 CONTINUE
      DO 80 I=1,N
      IDO=NAN+I
   80 TOP(I)=TOP(IDO)
   81 RETURN
      END
```

Numerical Quadrature

Subroutines QTFG and DQTFG

These subroutines perform the integration of a monotonically tabulated function by trapezoidal rule.

To compute the vector of integral values:

$$z_i = z(x_i) = \int_{x_1}^{x_i} y(x)\,dx \quad (i = 1,2, \ldots, n)$$

for a given table $(x_i, y_i; i = 1,2, \ldots, n)$ of monotonic arguments and their function values, the trapezoidal rule is used. It is assumed that the function to be integrated is continuous and can be differentiated at least twice. Starting with integral value $z_1 = 0$, successive integral values z_i $(i = 2,3,\ldots,n)$ are computed, using the trapezoidal rule in the following form:

$$z_1 = z_{i-1} + \frac{x_i - x_{i-1}}{2}(y_i + y_{i-1}) \quad (i = 2,3,\ldots,n)$$

Local truncation error at each step is:

$$R_i = -\frac{1}{12}(x_i - x_{i-1})^3 \cdot y''(\xi_i) \quad (\xi_i \quad [x_{i-1}, x_i])$$

However, these truncation errors may accumulate.

For reference see F.B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 75.

```
C                                                                       QTFG  10
C     ..................................................................QTFG  20
C                                                                       QTFG  30
C       SUBROUTINE QTFG                                                 QTFG  40
C                                                                       QTFG  50
C       PURPOSE                                                         QTFG  60
C          TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN         QTFG  70
C          GENERAL TABLE OF ARGUMENT AND FUNCTION VALUES.               QTFG  80
C                                                                       QTFG  90
C       USAGE                                                           QTFG 100
C          CALL QTFG (X,Y,Z,NDIM)                                       QTFG 110
C                                                                       QTFG 120
C       DESCRIPTION OF PARAMETERS                                       QTFG 130
C          X      - THE INPUT VECTOR OF ARGUMENT VALUES.                QTFG 140
C          Y      - THE INPUT VECTOR OF FUNCTION VALUES.                QTFG 150
C          Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE   QTFG 160
C                   IDENTICAL WITH X OR Y.                              QTFG 170
C          NDIM   - THE DIMENSION OF VECTORS X,Y,Z.                     QTFG 180
C                                                                       QTFG 190
C       REMARKS                                                         QTFG 200
C          NO ACTION IN CASE NDIM LESS THAN 1.                          QTFG 210
C                                                                       QTFG 220
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   QTFG 230
C          NONE                                                         QTFG 240
C                                                                       QTFG 250
C       METHOD                                                          QTFG 260
C          BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY     QTFG 270
C          MEANS OF TRAPEZOIDAL RULE (SECOND ORDER FORMULA).            QTFG 280
C          FOR REFERENCE, SEE                                           QTFG 290
C          F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,          QTFG 300
C          MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.75.           QTFG 310
C                                                                       QTFG 320
C     ..................................................................QTFG 330
C                                                                       QTFG 340
      SUBROUTINE QTFG(X,Y,Z,NDIM)                                       QTFG 350
C                                                                       QTFG 360
C                                                                       QTFG 370
      DIMENSION X(1),Y(1),Z(1)                                          QTFG 380
C                                                                       QTFG 390
      SUM2=0.                                                           QTFG 400
      IF(NDIM-1)4,3,1                                                   QTFG 410
C                                                                       QTFG 420
C     INTEGRATION LOOP                                                  QTFG 430
    1 DO 2 I=2,NDIM                                                     QTFG 440
      SUM1=SUM2                                                         QTFG 450
      SUM2=SUM2+.5*(X(I)-X(I-1))*(Y(I)+Y(I-1))                          QTFG 460
    2 Z(I-1)=SUM1                                                       QTFG 470
    3 Z(NDIM)=SUM2                                                      QTFG 480
    4 RETURN                                                            QTFG 490
      END                                                               QTFG 500
```

```
C                                                                       DTFG  10
C     ..................................................................DTFG  20
C                                                                       DTFG  30
C       SUBROUTINE DQTFG                                                DTFG  40
C                                                                       DTFG  50
C       PURPOSE                                                         DTFG  60
C          TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN         DTFG  70
C          GENERAL TABLE OF ARGUMENT AND FUNCTION VALUES.               DTFG  80
C                                                                       DTFG  90
C       USAGE                                                           DTFG 100
C          CALL DQTFG (X,Y,Z,NDIM)                                      DTFG 110
C                                                                       DTFG 120
C       DESCRIPTION OF PARAMETERS                                       DTFG 130
C          X      - DOUBLE PRECISION INPUT VECTOR OF ARGUMENT VALUES.   DTFG 140
C          Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.   DTFG 150
C          Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL       DTFG 160
C                   VALUES. Z MAY BE IDENTICAL WITH X OR Y.             DTFG 170
C          NDIM   - THE DIMENSION OF VECTORS X,Y,Z.                     DTFG 180
C                                                                       DTFG 190
C       REMARKS                                                         DTFG 200
C          NO ACTION IN CASE NDIM LESS THAN 1.                          DTFG 210
C                                                                       DTFG 220
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DTFG 230
C          NONE                                                         DTFG 240
C                                                                       DTFG 250
C       METHOD                                                          DTFG 260
C          BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY     DTFG 270
C          MEANS OF TRAPEZOIDAL RULE (SECOND ORDER FORMULA).            DTFG 280
C          FOR REFERENCE, SEE                                           DTFG 290
C          F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,          DTFG 300
C          MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.75.           DTFG 310
C                                                                       DTFG 320
C     ..................................................................DTFG 330
C                                                                       DTFG 340
      SUBROUTINE DQTFG(X,Y,Z,NDIM)                                      DTFG 350
C                                                                       DTFG 360
C                                                                       DTFG 370
      DIMENSION X(1),Y(1),Z(1)                                          DTFG 380
      DOUBLE PRECISION X,Y,Z,SUM1,SUM2                                  DTFG 390
C                                                                       DTFG 400
      SUM2=0.D0                                                         DTFG 410
      IF(NDIM-1)4,3,1                                                   DTFG 420
C                                                                       DTFG 430
C     INTEGRATION LOOP                                                  DTFG 440
    1 DO 2 I=2,NDIM                                                     DTFG 450
      SUM1=SUM2                                                         DTFG 460
      SUM2=SUM2+.5D0*(X(I)-X(I-1))*(Y(I)+Y(I-1))                        DTFG 470
    2 Z(I-1)=SUM1                                                       DTFG 480
    3 Z(NDIM)=SUM2                                                      DTFG 490
    4 RETURN                                                            DTFG 500
      END                                                               DTFG 510
```

ubroutines QTFE and DQTFE

'hese subroutines perform the integration of an quidistantly tabulated function by trapezoidal rule.

To compute the vector of integral values:

$$\left.\begin{array}{l} z_i = z\,(x_i) = \displaystyle\int_a^{x_i} y\,(x)\,dx \\ \\ \text{with } x_i = a + (i-1)\,h \end{array}\right\} (i = 1,2,3,\ldots,n)$$

or a table of function values y_i $(i = 1,2,\ldots,n)$, iven at equidistant points $x_i = a + (i-1)\cdot h$ $i = 1,2,\ldots,n)$, the trapezoidal rule is used. It is ssumed that the function to be integrated is continuous and can be differentiated at least twice. tarting with the integral value $z_1 = 0$, successive ntegral values z_i $(i = 2,3,\ldots,n)$ are computed, using ne trapezoidal rule in the following form:

$$z_i = z_{i-1} + \frac{h}{2}\,(y_i + y_{i-1}) \qquad (i=2,3,\ldots,n)$$

'ith a given increment h of argument values.

As local truncation error at each step is:

$$R_i = -\frac{1}{12}\,h^3 \cdot y''\,(\xi_i) \qquad (\xi_i \in [x_{i-1}, x_i])$$

ne global truncation error R in z_n appears as ollows:

$$R = -\frac{1}{12}\,(n-1)\,h^3\,y''\,(\xi) \qquad (\xi \in [x_1, x_n])$$

r

$$R = -\frac{\ell}{12} \cdot h^2 \cdot y''\,(\xi)$$

'here ℓ is the length of the whole integration interval.

For reference see F.B. Hildebrand, Introduction o Numerical Analysis. McGraw-Hill, New York/ 'oronto/London, 1956, pp. 75.

```
C                                                                       QTFE  10
C                                                                       QTFE  20
C     ..................................................................QTFE  30
C                                                                       QTFE  40
C        SUBROUTINE QTFE                                                QTFE  50
C                                                                       QTFE  60
C        PURPOSE                                                        QTFE  70
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        QTFE  80
C           EQUIDISTANT TABLE OF FUNCTION VALUES.                       QTFE  90
C                                                                       QTFE 100
C        USAGE                                                          QTFE 110
C           CALL QTFE (H,Y,Z,NDIM)                                      QTFE 120
C                                                                       QTFE 130
C        DESCRIPTION OF PARAMETERS                                      QTFE 140
C           H      - THE INCREMENT OF ARGUMENT VALUES.                  QTFE 150
C           Y      - THE INPUT VECTOR OF FUNCTION VALUES.               QTFE 160
C           Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE  QTFE 170
C                    IDENTICAL WITH Y.                                  QTFE 180
C           NDIM   - THE DIMENSION OF VECTORS Y AND Z.                  QTFE 190
C                                                                       QTFE 200
C        REMARKS                                                        QTFE 210
C           NO ACTION IN CASE NDIM LESS THAN 1.                         QTFE 220
C                                                                       QTFE 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QTFE 240
C           NONE                                                        QTFE 250
C                                                                       QTFE 260
C        METHOD                                                         QTFE 270
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    QTFE 280
C           MEANS OF TRAPEZOIDAL RULE (SECOND ORDER FORMULA).           QTFE 290
C           FOR REFERENCE, SEE                                          QTFE 300
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,         QTFE 310
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.75.          QTFE 320
C     ..................................................................QTFE 330
C                                                                       QTFE 340
      SUBROUTINE QTFE(H,Y,Z,NDIM)                                       QTFE 350
C                                                                       QTFE 360
C                                                                       QTFE 370
      DIMENSION Y(1),Z(1)                                               QTFE 380
C                                                                       QTFE 390
      SUM2=0.                                                           QTFE 400
      IF(NDIM-1)4,3,1                                                   QTFE 410
    1 HH=.5*H                                                           QTFE 420
C                                                                       QTFE 430
C     INTEGRATION LOOP                                                  QTFE 440
      DO 2 I=2,NDIM                                                     QTFE 450
      SUM1=SUM2                                                         QTFE 460
      SUM2=SUM2+HH*(Y(I)+Y(I-1))                                        QTFE 470
    2 Z(I-1)=SUM1                                                       QTFE 480
    3 Z(NDIM)=SUM2                                                      QTFE 490
    4 RETURN                                                            QTFE 500
      END                                                               QTFE 510
```

```
C                                                                       DTFE  10
C     ..................................................................DTFE  20
C                                                                       DTFE  30
C        SUBROUTINE DQTFE                                               DTFE  40
C                                                                       DTFE  50
C        PURPOSE                                                        DTFE  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        DTFE  70
C           EQUIDISTANT TABLE OF FUNCTION VALUES.                       DTFE  80
C                                                                       DTFE  90
C        USAGE                                                          DTFE 100
C           CALL DQTFE (H,Y,Z,NDIM)                                     DTFE 110
C                                                                       DTFE 120
C        DESCRIPTION OF PARAMETERS                                      DTFE 130
C           H      - DOUBLE PRECISION INCREMENT OF ARGUMENT VALUES.     DTFE 140
C           Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.  DTFE 150
C           Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL      DTFE 160
C                    VALUES. Z MAY BE IDENTICAL WITH Y.                 DTFE 170
C           NDIM   - THE DIMENSION OF VECTORS Y AND Z.                  DTFE 180
C                                                                       DTFE 190
C        REMARKS                                                        DTFE 200
C           NO ACTION IN CASE NDIM LESS THAN 1.                         DTFE 210
C                                                                       DTFE 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DTFE 230
C           NONE                                                        DTFE 240
C                                                                       DTFE 250
C        METHOD                                                         DTFE 260
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    DTFE 270
C           MEANS OF TRAPEZOIDAL RULE (SECOND ORDER FORMULA).           DTFE 280
C           FOR REFERENCE, SEE                                          DTFE 290
C           F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,         DTFE 300
C           MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.75.          DTFE 310
C                                                                       DTFE 320
C     ..................................................................DTFE 330
C                                                                       DTFE 340
      SUBROUTINE DQTFE(H,Y,Z,NDIM)                                      DTFE 350
C                                                                       DTFE 360
C                                                                       DTFE 370
      DIMENSION Y(1),Z(1)                                               DTFE 380
      DOUBLE PRECISION Y,Z,H,HH,SUM1,SUM2                               DTFE 390
C                                                                       DTFE 400
      SUM2=0.D0                                                         DTFE 410
      IF(NDIM-1)4,3,1                                                   DTFE 420
    1 HH=.5D0*H                                                         DTFE 430
C                                                                       DTFE 440
C     INTEGRATION LOOP                                                  DTFE 450
      DO 2 I=2,NDIM                                                     DTFE 460
      SUM1=SUM2                                                         DTFE 470
      SUM2=SUM2+HH*(Y(I)+Y(I-1))                                        DTFE 480
    2 Z(I-1)=SUM1                                                       DTFE 490
    3 Z(NDIM)=SUM2                                                      DTFE 500
    4 RETURN                                                            DTFE 510
      END                                                               DTFE 520
```

Subroutines QSF and DQSF

These subroutines perform the integration of an equidistantly tabulated function by Simpson's rule. To compute the vector of integral values:

$$\left.\begin{aligned} z_i = z(x_i) &= \int_a^{x_i} y(x)\,dx \\ \text{with } x_i &= a + (i-1)\,h \end{aligned}\right\} \quad (i=1,2,\ldots,n)$$

for a table of function values $y_i\,(i=1,2,\ldots,n)$, given at equidistant points $x_i = a+(i-1)\,h\ (i=1,2,\ldots,n)$, Simpson's rule together with Newton's 3/8 rule or a combination of these two rules is used. Local truncation error is of the order h^5 in all cases with more than three points in the given table. Only z_2 has a truncation error of the order h^4 if there are only three points in the given table. No action takes place if the table consists of less than three sample points.

The function to be integrated is assumed to be continuous and differentiable (three or four times, depending on the rule used).

Formulas used in this subroutine (z_j are integral values, y_j function values) are:

$$z_j = z_{j-1} + \frac{h}{3}\,(1.25\,y_{j-1} + 2y_j - 0.25\,y_{j+1}) \qquad (1)$$

$$z_j = z_{j-2} + \frac{h}{3}\,(y_{j-2} + 4y_{j-1} + y_j) \quad \text{(Simpson's rule)} \qquad (2)$$

$$z_j = z_{j-3} + \frac{3}{8}\,h\,(y_{j-3} + 3y_{j-2} + 3y_{j-1} + y_j) \qquad (3)$$

(Newton's 3/8 rule)

$$z_j = z_{j-5} + \frac{h}{3}\,(y_{j-5} + 3.875\,y_{j-4} + 2.625\,y_{j-3} + 2.625\,y_{j-2} + 3.875\,y_{j-1} + y_j) \qquad (4)$$

[combination of (2) and (3)]

Sometimes formula (2) is used in the following form:

$$z_j = z_{j+2} - \frac{h}{3}\,(y_j + 4\,y_{j+1} + y_{j+2}) \qquad (5)$$

Local truncation errors of formulas (1) . . . (4) are, respectively:

$$R_1 = \frac{1}{24}\,h^4\,y'''\ (\xi_1) \quad (\xi_1 \in [x_{j-1},\ x_{j+1}])$$

$$R_2 = -\frac{1}{90}\,h^5\,y''''\ (\xi_2) \quad (\xi_2 \in [x_{j-2},\ x_j])$$

$$R_3 = -\frac{3}{80}\,h^5\,y''''\ (\xi_3) \quad (\xi_3 \in [x_{j-3},\ x_j])$$

$$R_4 = -\frac{1}{144}\,h^5\,y''''\ (\xi_4) \quad (\xi_4 \in [x_{j-5},\ x_j])$$

However, these truncation errors may accumulate.

For reference see:

(1) F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 71-76.

(2) R. Zurmühl, Praktische Mathematik für Ingenieure und Physiker. Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 214-221.

```
C                                                                       QSF  10
C     ..................................................................QSF  20
C                                                                       QSF  30
C        SUBROUTINE QSF                                                 QSF  40
C                                                                       QSF  50
C        PURPOSE                                                        QSF  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        QSF  70
C           EQUIDISTANT TABLE OF FUNCTION VALUES.                       QSF  80
C                                                                       QSF  90
C        USAGE                                                          QSF 100
C           CALL QSF (H,Y,Z,NDIM)                                       QSF 110
C                                                                       QSF 120
C        DESCRIPTION OF PARAMETERS                                      QSF 130
C           H      - THE INCREMENT OF ARGUMENT VALUES.                  QSF 140
C           Y      - THE INPUT VECTOR OF FUNCTION VALUES.               QSF 150
C           Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE  QSF 160
C                    IDENTICAL WITH Y.                                  QSF 170
C           NDIM   - THE DIMENSION OF VECTORS Y AND Z.                  QSF 180
C                                                                       QSF 190
C        REMARKS                                                        QSF 200
C           NO ACTION IN CASE NDIM LESS THAN 3.                         QSF 210
C                                                                       QSF 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QSF 230
C           NONE                                                        QSF 240
C                                                                       QSF 250
C        METHOD                                                         QSF 260
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    QSF 270
C           MEANS OF SIMPSONS RULE TOGETHER WITH NEWTONS 3/8 RULE OR A  QSF 280
C           COMBINATION OF THESE TWO RULES. TRUNCATION ERROR IS OF      QSF 290
C           ORDER H**5 (I.E. FOURTH ORDER METHOD). ONLY IN CASE NDIM=3   QSF 300
C           TRUNCATION ERROR OF Z(2) IS OF ORDER H**4.                  QSF 310
C           FOR REFERENCE, SEE                                          QSF 320
C           (1) F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,     QSF 330
C               MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.71-76.   QSF 340
C           (2) R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND   QSF 350
C               PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963, QSF 360
C               PP.214-221.                                             QSF 370
C                                                                       QSF 380
C     ..................................................................QSF 390
C                                                                       QSF 400
      SUBROUTINE QSF(H,Y,Z,NDIM)                                        QSF 410
C                                                                       QSF 420
C                                                                       QSF 430
      DIMENSION Y(1),Z(1)                                               QSF 440
C                                                                       QSF 450
      HT=.33333334H                                                     QSF 460
      IF(NDIM-5)7,6,1                                                   QSF 470
C                                                                       QSF 480
C     NDIM IS GREATER THAN 5. PREPARATIONS OF INTEGRATION LOOP          QSF 490
    1 SUM1=Y(2)+Y(2)                                                    QSF 500
      SUM1=SUM1+SUM1                                                    QSF 510
      SUM1=HT*(Y(1)+SUM1+Y(3))                                          QSF 520
      ALX1=Y(4)+Y(4)                                                    QSF 530
      ALX1=ALX1+ALX1                                                    QSF 540
      ALX1=SUM1+HT*(Y(3)+ALX1+Y(5))                                     QSF 550
      ALX2=HT*(Y(1)+3.875*(Y(2)+Y(5))+2.625*(Y(3)+Y(4))+Y(6))           QSF 560
      SUM2=Y(5)+Y(5)                                                    QSF 570
      SUM2=SUM2+SUM2                                                    QSF 580
      SUM2=ALX2-HT*(Y(4)+SUM2+Y(6))                                     QSF 590
      Z(1)=0.                                                           QSF 600
      ALX=Y(3)+Y(3)                                                     QSF 610
      ALX=ALX+ALX                                                       QSF 620
      Z(2)=SUM2-HT*(Y(2)+ALX+Y(4))                                      QSF 630
      Z(3)=SUM1                                                         QSF 640
      Z(4)=SUM2                                                         QSF 650
      IF(NDIM-6)5,5,2                                                   QSF 660
C                                                                       QSF 670
C     INTEGRATION LOOP                                                  QSF 680
    2 DO 4 I=7,NDIM,2                                                   QSF 690
      SUM1=ALX1                                                         QSF 700
      SUM2=ALX2                                                         QSF 710
      ALX1=Y(I-1)+Y(I-1)                                                QSF 720
      ALX1=ALX1+ALX1                                                    QSF 730
      ALX1=SUM1+HT*(Y(I-2)+ALX1+Y(I))                                   QSF 740
      Z(I-2)=SUM1                                                       QSF 750
      IF(I-NDIM)3,6,6                                                   QSF 760
    3 ALX2=Y(I)+Y(I)                                                    QSF 770
      ALX2=ALX2+ALX2                                                    QSF 780
      ALX2=SUM2+HT*(Y(I-1)+ALX2+Y(I+1))                                 QSF 790
    4 Z(I-1)=SUM2                                                       QSF 800
    5 Z(NDIM-1)=ALX1                                                    QSF 810
      Z(NDIM)=ALX2                                                      QSF 820
      RETURN                                                            QSF 830
    6 Z(NDIM-1)=SUM2                                                    QSF 840
      Z(NDIM)=ALX1                                                      QSF 850
      RETURN                                                            QSF 860
C     END OF INTEGRATION LOOP                                           QSF 870
```

```
    7 IF(NDIM-3)12,11,8

      NDIM IS EQUAL TO 4 OR 5
    8 SUP2=1.125*HT*(Y(1)+Y(2)+Y(2)+Y(2)+Y(3)+Y(3)+Y(3)+Y(4))
      SUM1=Y(2)+Y(2)
      SUM1=SUM1+SUM1
      SUM1=HT*(Y(1)+SUM1+Y(3))
      Z(1)=0.
      AUX1=Y(3)+Y(3)
      AUX1=AUX1+AUX1
      Z(2)=SUP2-HT*(Y(2)+AUX1+Y(4))
      IF(NDIM-5)10,9,9
    9 AUX1=Y(4)+Y(4)
      AUX1=AUX1+AUX1
      Z(5)=SUM1+HT*(Y(3)+AUX1+Y(5))
   10 Z(3)=SUM1
      Z(4)=SUP2
      RETURN

      NDIM IS EQUAL TO 3
   11 SUM1=HT*(1.25*Y(1)+Y(2)+Y(2)-.25*Y(3))
      SUM2=Y(2)+Y(2)
      SUM2=SUM2+SUM2
      Z(3)=HT*(Y(1)+SUM2+Y(3))
      Z(1)=0.
      Z(2)=SUM1
   12 RETURN
      END
```

```
    6 Z(NDIM-1)=SUM2
      Z(NDIM)=AUX1
      RETURN
C     END OF INTEGRATION LOOP
C
    7 IF(NDIM-3)12,11,8
C
C     NDIM IS EQUAL TO 4 OR 5
    8 SUM2=1.125D0*HT*(Y(1)+Y(2)+Y(2)+Y(2)+Y(3)+Y(3)+Y(3)+Y(4))
      SUM1=Y(2)+Y(2)
      SUM1=SUM1+SUM1
      SUM1=HT*(Y(1)+SUM1+Y(3))
      Z(1)=0.D0
      AUX1=Y(3)+Y(3)
      AUX1=AUX1+AUX1
      Z(2)=SUM2-HT*(Y(2)+AUX1+Y(4))
      IF(NDIM-5)10,9,9
    9 AUX1=Y(4)+Y(4)
      AUX1=AUX1+AUX1
      Z(5)=SUM1+HT*(Y(3)+AUX1+Y(5))
   10 Z(3)=SUM1
      Z(4)=SUM2
      RETURN
C
C     NDIM IS EQUAL TO 3
   11 SUM1=HT*(1.25D0*Y(1)+Y(2)+Y(2)-.25D0*Y(3))
      SUM2=Y(2)+Y(2)
      SUM2=SUM2+SUM2
      Z(3)=HT*(Y(1)+SUM2+Y(3))
      Z(1)=0.D0
      Z(2)=SUM1
   12 RETURN
      END
```

```
*********************************************************************
      SUBROUTINE DQSF

      PURPOSE
         TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN
         EQUIDISTANT TABLE OF FUNCTION VALUES.

      USAGE
         CALL DQSF (H,Y,Z,NDIM)

      DESCRIPTION OF PARAMETERS
         H      - DOUBLE PRECISION INCREMENT OF ARGUMENT VALUES.
         Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.
         Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL
                  VALUES. Z MAY BE IDENTICAL WITH Y.
         NDIM   - THE DIMENSION OF VECTORS Y AND Z.

      REMARKS
         NO ACTION IN CASE NDIM LESS THAN 3.

      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
         NONE

      METHOD
         BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY
         MEANS OF SIMPSONS RULE TOGETHER WITH NEWTONS 3/8 RULE OR A
         COMBINATION OF THESE TWO RULES. TRUNCATION ERROR IS OF
         ORDER H**5 (I.E. FOURTH ORDER METHOD). ONLY IN CASE NDIM=3
         TRUNCATION ERROR OF Z(2) IS OF ORDER H**4.
         FOR REFERENCE, SEE
         (1) F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,
             MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.71-76.
         (2) R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND
             PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963,
             PP.214-221.
*********************************************************************
      SUBROUTINE DQSF(H,Y,Z,NDIM)


      DIMENSION Y(1),Z(1)
      DOUBLE PRECISION Y,Z,H,HT,SUM1,SUM2,AUX,AUX1,AUX2

      HT=.33333333333333333D0*H
      IF(NDIM-5)7,8,1

      NDIM IS GREATER THAN 5. PREPARATIONS OF INTEGRATION LOOP
    1 SUM1=Y(2)+Y(2)
      SUM1=SUM1+SUM1
      SUM1=HT*(Y(1)+SUM1+Y(3))
      AUX1=Y(4)+Y(4)
      AUX1=AUX1+AUX1
      AUX1=SUM1+HT*(Y(3)+AUX1+Y(5))
      AUX2=HT*(Y(1)+3.875D0*(Y(2)+Y(5))+2.625D0*(Y(3)+Y(4))+Y(6))
      SUM2=Y(5)+Y(5)
      SUM2=SUM2+SUM2
      SUM2=AUX2-HT*(Y(4)+SUM2+Y(6))
      Z(1)=0.D0
      AUX=Y(3)+Y(3)
      AUX=AUX+AUX
      Z(2)=SUM2-HT*(Y(2)+AUX+Y(4))
      Z(3)=SUM1
      Z(4)=SUM2
      IF(NDIM-6)5,5,2

      INTEGRATION LOOP
    2 DO 4 I=7,NDIM,2
      SUM1=AUX1
      SUM2=AUX2
      AUX1=Y(I-1)+Y(I-1)
      AUX1=AUX1+AUX1
      AUX1=SUM1+HT*(Y(I-2)+AUX1+Y(I))
      Z(I-2)=SUM1
      IF(I-NDIM)3,6,6
    3 AUX2=Y(I)+Y(I)
      AUX2=AUX2+AUX2
      AUX2=SUM2+HT*(Y(I-1)+AUX2+Y(I+1))
    4 Z(I-1)=SUM2
    5 Z(NDIM-1)=AUX1
      Z(NDIM)=AUX2
      RETURN
```

Subroutines QHFG and DQHFG

These subroutines perform the integration of a monotonically tabulated function with first derivative by a Hermitian formula of first order. To compute the vector of integral values:

$$z_i = z(x_i) = \int_{x_1}^{x_i} y(x)\,dx \quad (i = 1, 2, \ldots, n)$$

for a given table x_i, y_i, y_i' $(i = 1, 2, \ldots, n)$ of monotonic arguments and their function and derivative values, a first-order Hermitian formula is used. It is assumed that the function to be integrated is continuous and can be differentiated at least four times. Starting with the integral value $z_1 = 0$, successive integral values z_i $(i = 2, 3, \ldots, n)$ are computed by means of the following formula:

$$z_i = z_{i-1} + \frac{x_i - z_{i-1}}{2}\left[y_{i-1} + y_i + \frac{x_i - x_{i-1}}{6}(y_{i-1}' - y_i')\right]$$

$$(i = 2, 3, \ldots, n)$$

Local truncation error at each step is:

$$R_i = \frac{(x_i - x_{i-1})^5}{720} \cdot y^{(4)}(\xi_i) \quad (\xi_i \in [x_{i-1}, x_i])$$

However, these truncation errors may accumulate.

For reference see:

(1) F.B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 314-319.

(2) R. Zurmühl, Praktische Mathematik für Ingenieure und Physiker. Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 227-230.

```
C                                                                       QHFG  10
C     ..................................................................QHFG  20
C                                                                       QHFG  30
C        SUBROUTINE QHFG                                                QHFG  40
C                                                                       QHFG  50
C        PURPOSE                                                        QHFG  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        QHFG  70
C           GENERAL TABLE OF ARGUMENT, FUNCTION, AND DERIVATIVE VALUES. QHFG  80
C                                                                       QHFG  90
C        USAGE                                                          QHFG 100
C           CALL QHFG (X,Y,DERY,Z,NDIM)                                 QHFG 110
C                                                                       QHFG 120
C        DESCRIPTION OF PARAMETERS                                      QHFG 130
C           X      - THE INPUT VECTOR OF ARGUMENT VALUES.               QHFG 140
C           Y      - THE INPUT VECTOR OF FUNCTION VALUES.               QHFG 150
C           DERY   - THE INPUT VECTOR OF DERIVATIVE VALUES.             QHFG 160
C           Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE  QHFG 170
C                    IDENTICAL WITH X,Y OR DERY.                        QHFG 180
C           NDIM   - THE DIMENSION OF VECTORS X,Y,DERY,Z.               QHFG 190
C                                                                       QHFG 200
C        REMARKS                                                        QHFG 210
C           NO ACTION IN CASE NDIM LESS THAN 1.                         QHFG 220
C                                                                       QHFG 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QHFG 240
C           NONE                                                        QHFG 250
C                                                                       QHFG 260
C        METHOD                                                         QHFG 270
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    QHFG 280
C           MEANS OF HERMITEAN FOURTH ORDER INTEGRATION FORMULA.        QHFG 290
C           FOR REFERENCE, SEE                                          QHFG 300
C           (1) F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,     QHFG 310
C               MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.314-319. QHFG 320
C           (2) R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND   QHFG 330
C               PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963, QHFG 340
C               PP.227-230.                                             QHFG 350
C                                                                       QHFG 360
C     ..................................................................QHFG 370
C                                                                       QHFG 380
      SUBROUTINE QHFG(X,Y,DERY,Z,NDIM)                                  QHFG 390
C                                                                       QHFG 400
C                                                                       QHFG 410
      DIMENSION X(1),Y(1),DERY(1),Z(1)                                  QHFG 420
C                                                                       QHFG 430
      SUM2=0.                                                           QHFG 440
      IF(NDIM-1)4,3,1                                                   QHFG 450
C                                                                       QHFG 460
C     INTEGRATION LOOP                                                  QHFG 470
    1 DO 2 I=2,NDIM                                                     QHFG 480
      SUM1=SUM2                                                         QHFG 490
      SUM2=.5*(X(I)-X(I-1))                                             QHFG 500
      SUM2=SUM1+SUM2*((Y(I)+Y(I-1))+.3333333*SUM2*(DERY(I-1)-DERY(I)))  QHFG 510
    2 Z(I-1)=SUM1                                                       QHFG 520
    3 Z(NDIM)=SUM2                                                      QHFG 530
    4 RETURN                                                            QHFG 540
      END                                                               QHFG 550
```

```
C                                                                       DQHG  10
C     ..................................................................DQHG  20
C                                                                       DQHG  30
C        SUBROUTINE DQHFG                                               DQHG  40
C                                                                       DQHG  50
C        PURPOSE                                                        DQHG  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        DQHG  70
C           GENERAL TABLE OF ARGUMENT, FUNCTION, AND DERIVATIVE VALUES. DQHG  80
C                                                                       DQHG  90
C        USAGE                                                          DQHG 100
C           CALL DQHFG (X,Y,DERY,Z,NDIM)                                DQHG 110
C                                                                       DQHG 120
C        DESCRIPTION OF PARAMETERS                                      DQHG 130
C           X      - DOUBLE PRECISION INPUT VECTOR OF ARGUMENT VALUES.  DQHG 140
C           Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.  DQHG 150
C           DERY   - DOUBLE PRECISION INPUT VECTOR OF DERIVATIVE VALUES.DQHG 160
C           Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL      DQHG 170
C                    VALUES. Z MAY BE IDENTICAL WITH X, Y OR DERY.      DQHG 180
C           NDIM   - THE DIMENSION OF VECTORS X,Y,DERY,Z.               DQHG 190
C                                                                       DQHG 200
C        REMARKS                                                        DQHG 210
C           NO ACTION IN CASE NDIM LESS THAN 1.                         DQHG 220
C                                                                       DQHG 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQHG 240
C           NONE                                                        DQHG 250
C                                                                       DQHG 260
C        METHOD                                                         DQHG 270
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    DQHG 280
C           MEANS OF HERMITEAN FOURTH ORDER INTEGRATION FORMULA.        DQHG 290
C           FOR REFERENCE, SEE                                          DQHG 300
C           (1) F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,     DQHG 310
C               MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.314-319. DQHG 320
C           (2) R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND   DQHG 330
C               PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963, DQHG 340
C               PP.227-230.                                             DQHG 350
C                                                                       DQHG 360
C     ..................................................................DQHG 370
C                                                                       DQHG 380
      SUBROUTINE DQHFG(X,Y,DERY,Z,NDIM)                                 DQHG 390
C                                                                       DQHG 400
C                                                                       DQHG 410
      DIMENSION X(1),Y(1),DERY(1),Z(1)                                  DQHG 420
      DOUBLE PRECISION X,Y,DERY,Z,SUM1,SUM2                             DQHG 430
C                                                                       DQHG 440
      SUM2=0.D0                                                         DQHG 450
      IF(NDIM-1)4,3,1                                                   DQHG 460
C                                                                       DQHG 470
C     INTEGRATION LOOP                                                  DQHG 480
    1 DO 2 I=2,NDIM                                                     DQHG 490
      SUM1=SUM2                                                         DQHG 500
      SUM2=.5D0*(X(I)-X(I-1))                                           DQHG 510
      SUM2=SUM1+SUM2*((Y(I)+Y(I-1))+.3333333333333333D0*SUM2*           DQHG 520
     1(DERY(I-1)-DERY(I)))                                              DQHG 530
    2 Z(I-1)=SUM1                                                       DQHG 540
    3 Z(NDIM)=SUM2                                                      DQHG 550
    4 RETURN                                                            DQHG 560
      END                                                               DQHG 570
```

Subroutines QHFE and DQHFE

These subroutines perform the integration of an equidistantly tabulated function with first derivative by a Hermitian formula of first order. To compute the vector of integral values:

$$\left.\begin{aligned} z_i &= z(x_i) = \int_a^{x_i} y(x)\,dx \\ \text{with} \quad x_i &= a + (i-1)\,h \end{aligned}\right\} (i = 1, 2, \ldots, n)$$

for a table of function and first derivative values y_i, y_i' $(i = 1, 2, \ldots, n)$, given at equidistant points $x_i = a + (i - 1)\,h$, a first-order Hermitian formula is used. It is assumed that the function to be integrated is continuous and can be differentiated at least four times. Starting with the integral value $z_1 = 0$, successive integral values z_i $(i = 2, 3, \ldots, n)$ are computed by means of the following formula:

$$z_i = z_{i-1} + \frac{h}{2} \cdot \left[y_{i-1} + y_i + \frac{h}{6} \left(y_{i-1}' - y_i' \right) \right]$$

$$(i = 1, 2, \ldots, n)$$

with a given increment h of argument values.

As the local truncation error at each step is:

$$R_i = \frac{h^5}{720} \cdot y^{(4)}(\xi_i) \qquad (\xi_i \in [x_{i-1}, x_i])$$

the global truncation error R in z_n appears as follows:

$$R = \frac{1}{720} (n-1)\, h^5\, y^{(4)}(\xi) \qquad (\xi \in [x_1, x_n])$$

or

$$R = \frac{\ell}{720}\, h^4\, y^{(4)}(\xi)$$

where ℓ is the length of the whole integration interval.

For reference see:

(1) F.B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 314-319.

(2) R. Zurmühl, Praktische Mathematik für Ingenieure and Physiker. Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 227-230.


```
C                                                                       QHFE  10
C     ..................................................................QHFE  20
C                                                                       QHFE  30
C        SUBROUTINE QHFE                                                QHFE  40
C                                                                       QHFE  50
C        PURPOSE                                                        QHFE  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        QHFE  70
C           EQUIDISTANT TABLE OF FUNCTION AND DERIVATIVE VALUES.        QHFE  80
C                                                                       QHFE  90
C        USAGE                                                          QHFE 100
C           CALL QHFE (H,Y,DERY,Z,NDIM)                                 QHFE 110
C                                                                       QHFE 120
C        DESCRIPTION OF PARAMETERS                                      QHFE 130
C           H      - THE INCREMENT OF ARGUMENT VALUES.                  QHFE 140
C           Y      - THE INPUT VECTOR OF FUNCTION VALUES.               QHFE 150
C           DERY   - THE INPUT VECTOR OF DERIVATIVE VALUES.             QHFE 160
C           Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE  QHFE 170
C                    IDENTICAL WITH Y OR DERY.                          QHFE 180
C           NDIM   - THE DIMENSION OF VECTORS Y,DERY,Z.                 QHFE 190
C                                                                       QHFE 200
C        REMARKS                                                        QHFE 210
C           NO ACTION IN CASE NDIM LESS THAN 1.                         QHFE 220
C                                                                       QHFE 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QHFE 240
C           NONE                                                        QHFE 250
C                                                                       QHFE 260
C        METHOD                                                         QHFE 270
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    QHFE 280
C           MEANS OF HERMITEAN FOURTH ORDER INTEGRATION FORMULA.        QHFE 290
C           FOR REFERENCE, SEE                                          QHFE 300
C           (1) F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,     QHFE 310
C               MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.314-319. QHFE 320
C           (2) R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND   QHFE 330
C               PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963, QHFE 340
C               PP.227-230.                                             QHFE 350
C                                                                       QHFE 360
C     ..................................................................QHFE 370
C                                                                       QHFE 380
      SUBROUTINE QHFE(H,Y,DERY,Z,NDIM)                                  QHFE 390
C                                                                       QHFE 400
C                                                                       QHFE 410
      DIMENSION Y(1),DERY(1),Z(1)                                       QHFE 420
C                                                                       QHFE 430
      SUM2=0.                                                           QHFE 440
      IF(NDIM-1)4,3,1                                                   QHFE 450
    1 HH=.5*H                                                           QHFE 460
      HS=.1666667*H                                                     QHFE 470
C                                                                       QHFE 480
C     INTEGRATION LOOP                                                  QHFE 490
      DO 2 I=2,NDIM                                                     QHFE 500
      SUM1=SUM2                                                         QHFE 510
      SUM2=SUM2+HH*((Y(I)+Y(I-1))+HS*(DERY(I-1)-DERY(I)))               QHFE 520
    2 Z(I-1)=SUM1                                                       QHFE 530
    3 Z(NDIM)=SUM2                                                      QHFE 540
    4 RETURN                                                            QHFE 550
      END                                                               QHFE 560
```

```
C                                                                       DQHE  10
C     ..................................................................DQHE  20
C                                                                       DQHE  30
C        SUBROUTINE DQHFE                                               DQHE  40
C                                                                       DQHE  50
C        PURPOSE                                                        DQHE  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        DQHE  70
C           EQUIDISTANT TABLE OF FUNCTION AND DERIVATIVE VALUES.        DQHE  80
C                                                                       DQHE  90
C        USAGE                                                          DQHE 100
C           CALL DQHFE (H,Y,DERY,Z,NDIM)                                DQHE 110
C                                                                       DQHE 120
C        DESCRIPTION OF PARAMETERS                                      DQHE 130
C           H      - DOUBLE PRECISION INCREMENT OF ARGUMENT VALUES.     DQHE 140
C           Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.  DQHE 150
C           DERY   - DOUBLE PRECISION INPUT VECTOR OF DERIVATIVE VALUES.DQHE 160
C           Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL      DQHE 170
C                    VALUES. Z MAY BE IDENTICAL WITH Y OR DERY.         DQHE 180
C           NDIM   - THE DIMENSION OF VECTORS Y,DERY,Z.                 DQHE 190
C                                                                       DQHE 200
C        REMARKS                                                        DQHE 210
C           NO ACTION IN CASE NDIM LESS THAN 1.                         DQHE 220
C                                                                       DQHE 230
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQHE 240
C           NONE                                                        DQHE 250
C                                                                       DQHE 260
C        METHOD                                                         DQHE 270
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    DQHE 280
C           MEANS OF HERMITEAN FOURTH ORDER INTEGRATION FORMULA.        DQHE 290
C           FOR REFERENCE, SEE                                          DQHE 300
C           (1) F.B.HILDEBRAND, INTRODUCTION TO NUMERICAL ANALYSIS,     DQHE 310
C               MCGRAW-HILL, NEW YORK/TORONTO/LONDON, 1956, PP.314-319. DQHE 320
C           (2) R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND   DQHE 330
C               PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963, DQHE 340
C               PP.227-230.                                             DQHE 350
C                                                                       DQHE 360
C     ..................................................................DQHE 370
C                                                                       DQHE 380
      SUBROUTINE DQHFE(H,Y,DERY,Z,NDIM)                                 DQHE 390
C                                                                       DQHE 400
C                                                                       DQHE 410
      DIMENSION Y(1),DERY(1),Z(1)                                       DQHE 420
      DOUBLE PRECISION Y,DERY,Z,H,HH,HS,SUM1,SUM2                       DQHE 430
C                                                                       DQHE 440
      SUM2=0.D0                                                         DQHE 450
      IF(NDIM-1)4,3,1                                                   DQHE 460
    1 HH=.5D0*H                                                         DQHE 470
      HS=.16666666666666667D0*H                                         DQHE 480
C                                                                       DQHE 490
C     INTEGRATION LOOP                                                  DQHE 500
      DO 2 I=2,NDIM                                                     DQHE 510
      SUM1=SUM2                                                         DQHE 520
      SUM2=SUM2+HH*((Y(I)+Y(I-1))+HS*(DERY(I-1)-DERY(I)))               DQHE 530
    2 Z(I-1)=SUM1                                                       DQHE 540
    3 Z(NDIM)=SUM2                                                      DQHE 550
    4 RETURN                                                            DQHE 560
      END                                                               DQHE 570
```

Subroutines QHSG and DQHSG

These subroutines perform the integration of a monotonically tabulated function with first and second derivatives by a Hermitian formula of second order. To compute the vector of integral values:

$$z_i = z(x_i) = \int_{x_1}^{x_i} y(x)\,dx \quad (i = 1,2,\ldots,n)$$

for a given table x_i, y_i, y_i', y_i'' $(i=1,2,\ldots,n)$ of monotonic arguments and their function and first and second derivative values, a second-order Hermitian formula is used. It is assumed that the function to be integrated is continuous and can be differentiated at least six times. Starting with the integral value $z_i = 0$, successive integral values z_i $(i = 2,3,\ldots,n)$ are computed by means of the following formula:

$$z_i = z_{i-1} + \frac{x_i - x_{i-1}}{2}\left\{y_{i-1} + y_i + \frac{x_i - x_{i-1}}{5}\left[y'_{i-1} - y'_i + \frac{x_i - x_{i-1}}{12}\left(y''_{i-1} + y''_i\right)\right]\right\} (i=2,3,\ldots,n)$$

Local truncation error at each step is:

$$R_i = -\frac{(x_i - x_{i-1})^7}{100\ 800}\, y^{(6)}(\xi_i) \quad (\xi_i \in [x_{i-1}\ \ x_i])$$

However, these truncation errors may accumulate.

For reference see R. Zurmühl, Praktische Mathematik für Ingenieure und Physiker. Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 227-230.

```
C                                                                       QHSG  10
C     ...................................................................QHSG  20
C                                                                       QHSG  30
C        SUBROUTINE QHSG                                                QHSG  40
C                                                                       QHSG  50
C        PURPOSE                                                        QHSG  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        QHSG  70
C           GENERAL TABLE OF ARGUMENT, FUNCTION, FIRST DERIVATIVE,      QHSG  80
C           AND SECOND DERIVATIVE VALUES.                               QHSG  90
C                                                                       QHSG 100
C        USAGE                                                          QHSG 110
C           CALL QHSG (X,Y,FDY,SDY,Z,NDIM)                              QHSG 120
C                                                                       QHSG 130
C        DESCRIPTION OF PARAMETERS                                      QHSG 140
C           X      - THE INPUT VECTOR OF ARGUMENT VALUES.               QHSG 150
C           Y      - THE INPUT VECTOR OF FUNCTION VALUES.               QHSG 160
C           FDY    - THE INPUT VECTOR OF FIRST DERIVATIVE.              QHSG 170
C           SDY    - THE INPUT VECTOR OF SECOND DERIVATIVE.             QHSG 180
C           Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE  QHSG 190
C                    IDENTICAL WITH X,Y,FDY OR SDY.                     QHSG 200
C           NDIM   - THE DIMENSION OF VECTORS X,Y,FDY,SDY,Z.            QHSG 210
C                                                                       QHSG 220
C        REMARKS                                                        QHSG 230
C           NO ACTION IN CASE NDIM LESS THAN 1.                         QHSG 240
C                                                                       QHSG 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QHSG 260
C           NONE                                                        QHSG 270
C                                                                       QHSG 280
C        METHOD                                                         QHSG 290
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    QHSG 300
C           MEANS OF HERMITEAN SIXTH ORDER INTEGRATION FORMULA.         QHSG 310
C           FOR REFERENCE, SEE                                          QHSG 320
C           R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND       QHSG 330
C           PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963,     QHSG 340
C           PP.227-230.                                                 QHSG 350
C                                                                       QHSG 360
C     ...................................................................QHSG 370
C                                                                       QHSG 380
      SUBROUTINE QHSG(X,Y,FDY,SDY,Z,NDIM)                               QHSG 390
C                                                                       QHSG 400
C                                                                       QHSG 410
      DIMENSION X(1),Y(1),FDY(1),SDY(1),Z(1)                            QHSG 420
C                                                                       QHSG 430
      SUM2=0.                                                           QHSG 440
      IF(NDIM-1)4,3,1                                                   QHSG 450
C                                                                       QHSG 460
C     INTEGRATION LOOP                                                  QHSG 470
    1 DO 2 I=2,NDIM                                                     QHSG 480
      SUM1=SUM2                                                         QHSG 490
      SUM2=.5*(X(I)-X(I-1))                                             QHSG 500
     0SUM2=SUM1+SUM2*((Y(I-1)+Y(I))+.4*SUM2*((FDY(I-1)-FDY(I))+         QHSG 510
     1     .1666667*SUM2*(SDY(I-1)+SDY(I))))                            QHSG 520
    2 Z(I-1)=SUM1                                                       QHSG 530
    3 Z(NDIM)=SUM2                                                      QHSG 540
    4 RETURN                                                            QHSG 550
      END                                                               QHSG 560
```

```
C                                                                       DHSG  10
C     ...................................................................DHSG  20
C                                                                       DHSG  30
C        SUBROUTINE DQHSG                                               DHSG  40
C                                                                       DHSG  50
C        PURPOSE                                                        DHSG  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        DHSG  70
C           GENERAL TABLE OF ARGUMENT, FUNCTION, FIRST DERIVATIVE,      DHSG  80
C           AND SECOND DERIVATIVE VALUES.                               DHSG  90
C                                                                       DHSG 100
C        USAGE                                                          DHSG 110
C           CALL DQHSG (X,Y,FDY,SDY,Z,NDIM)                             DHSG 120
C                                                                       DHSG 130
C        DESCRIPTION OF PARAMETERS                                      DHSG 140
C           X      - DOUBLE PRECISION INPUT VECTOR OF ARGUMENT VALUES.  DHSG 150
C           Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.  DHSG 160
C           FDY    - DOUBLE PRECISION INPUT VECTOR OF FIRST DERIVATIVE. DHSG 170
C           SDY    - DOUBLE PRECISION INPUT VECTOR OF SECOND DERIVATIVE.DHSG 180
C           Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL     DHSG 190
C                    VALUES. Z MAY BE IDENTICAL WITH X, Y, FDY OR SDY.  DHSG 200
C           NDIM   - THE DIMENSION OF VECTORS X,Y,FDY,SDY,Z.            DHSG 210
C                                                                       DHSG 220
C        REMARKS                                                        DHSG 230
C           NO ACTION IN CASE NDIM LESS THAN 1.                         DHSG 240
C                                                                       DHSG 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DHSG 260
C           NONE                                                        DHSG 270
C                                                                       DHSG 280
C        METHOD                                                         DHSG 290
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    DHSG 300
C           MEANS OF HERMITEAN SIXTH ORDER INTEGRATION FORMULA.         DHSG 310
C           FOR REFERENCE, SEE                                          DHSG 320
C           R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND       DHSG 330
C           PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963,     DHSG 340
C           PP.227-230.                                                 DHSG 350
C                                                                       DHSG 360
C     ...................................................................DHSG 370
C                                                                       DHSG 380
      SUBROUTINE DQHSG(X,Y,FDY,SDY,Z,NDIM)                              DHSG 390
C                                                                       DHSG 400
C                                                                       DHSG 410
      DIMENSION X(1),Y(1),FDY(1),SDY(1),Z(1)                            DHSG 420
      DOUBLE PRECISION X,Y,FDY,SDY,Z,SUM1,SUM2                          DHSG 430
C                                                                       DHSG 440
      SUM2=0.D0                                                         DHSG 450
      IF(NDIM-1)4,3,1                                                   DHSG 460
C                                                                       DHSG 470
C     INTEGRATION LOOP                                                  DHSG 480
    1 DO 2 I=2,NDIM                                                     DHSG 490
      SUM1=SUM2                                                         DHSG 500
      SUM2=.5D0*(X(I)-X(I-1))                                           DHSG 510
     0SUM2=SUM1+SUM2*((Y(I-1)+Y(I))+.4D0*SUM2*((FDY(I-1)-FDY(I))+       DHSG 520
     1     .16666666666666667D0*SUM2*(SDY(I-1)+SDY(I))))                DHSG 530
    2 Z(I-1)=SUM1                                                       DHSG 540
    3 Z(NDIM)=SUM2                                                      DHSG 550
    4 RETURN                                                            DHSG 560
      END                                                               DHSG 570
```

Subroutines QHSE and DQHSE

These subroutines perform the integration of an equidistantly tabulated function with first and second derivatives by a Hermitian formula of second order.

To compute the vector of integral values:

$$\left.\begin{aligned} z_i = z(x_i) &= \int_a^{x_i} y(x)\,dx \\ \text{with } x_i &= a + (i-1)\,h \end{aligned}\right\} (i = 1, 2, \ldots, n)$$

for a table of function and first and second derivative values y_i, y'_i, y''_i $(i = 1, 2, \ldots, n)$, given at equidistant points $x_i = a + (i-1)\,h$, a second-order Hermitian formula is used. It is assumed that the function to be integrated is continuous and can be differentiated at least six times. Starting with the integral value $z_1 = 0$, successive integral values z_i $(i = 2, 3, \ldots, n)$ are computed by means of the following formula:

$$z_i = z_{i-1} + \frac{h}{2}\left\{y_{i-1} + y_i + \frac{h}{5}\left[y'_{i-1} - y'_i + \frac{h}{12}\left(y''_{i-1} + y''_i\right)\right]\right\} \quad (i=2,3,\ldots,n)$$

with a given increment h of argument values.

As the local truncation error at each step is:

$$R_i = -\frac{h^7}{100\,800}\, y^{(6)}(\xi_i) \quad (\xi_i \in [x_{i-1}, x_i]),$$

the global truncation error R in z_n appears as follows:

$$R = -\frac{1}{100\,800}\,(n-1)\,h^7\, y^{(6)}(\xi) \quad (\xi \in [x_1, xn])$$

or

$$R = -\frac{\ell}{100\,800}\, h^6\, y^{(6)}(\xi)$$

where ℓ is the length of the whole integration interval.

For reference see R. Zurmühl, Praktische Mathematik für Ingenieure und Physiker. Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 227-230.

```
C                                                                       QHSE  10
C     ..................................................................QHSE  20
C                                                                       QHSE  30
C        SUBROUTINE QHSE                                                QHSE  40
C                                                                       QHSE  50
C        PURPOSE                                                        QHSE  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        QHSE  70
C           EQUIDISTANT TABLE OF FUNCTION, FIRST DERIVATIVE,            QHSE  80
C           AND SECOND DERIVATIVE VALUES.                               QHSE  90
C                                                                       QHSE 100
C        USAGE                                                          QHSE 110
C           CALL QHSE (H,Y,FDY,SDY,Z,NDIM)                              QHSE 120
C                                                                       QHSE 130
C        DESCRIPTION OF PARAMETERS                                      QHSE 140
C           H      - THE INCREMENT OF ARGUMENT VALUES.                  QHSE 150
C           Y      - THE INPUT VECTOR OF FUNCTION VALUES.               QHSE 160
C           FDY    - THE INPUT VECTOR OF FIRST DERIVATIVE.              QHSE 170
C           SDY    - THE INPUT VECTOR OF SECOND DERIVATIVE.             QHSE 180
C           Z      - THE RESULTING VECTOR OF INTEGRAL VALUES. Z MAY BE  QHSE 190
C                    IDENTICAL WITH Y,FDY OR SDY.                       QHSE 200
C           NDIM   - THE DIMENSION OF VECTORS Y,FDY,SDY,Z.              QHSE 210
C                                                                       QHSE 220
C        REMARKS                                                        QHSE 230
C           NO ACTION IN CASE NDIM LESS THAN 1.                         QHSE 240
C                                                                       QHSE 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QHSE 260
C           NONE                                                        QHSE 270
C                                                                       QHSE 280
C        METHOD                                                         QHSE 290
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    QHSE 300
C           MEANS OF HERMITEAN SIXTH ORDER INTEGRATION FORMULA.         QHSE 310
C           FOR REFERENCE, SEE                                          QHSE 320
C           R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND       QHSE 330
C           PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963,     QHSE 340
C           PP.227-230.                                                 QHSE 350
C                                                                       QHSE 360
C     ..................................................................QHSE 370
C                                                                       QHSE 380
      SUBROUTINE QHSE(H,Y,FDY,SDY,Z,NDIM)                               QHSE 390
C                                                                       QHSE 400
C                                                                       QHSE 410
      DIMENSION Y(1),FDY(1),SDY(1),Z(1)                                 QHSE 420
C                                                                       QHSE 430
      SUM2=0.                                                           QHSE 440
      IF(NDIM-1)4,3,1                                                   QHSE 450
    1 HH=.5*H                                                           QHSE 460
      HF=.2*H                                                           QHSE 470
      HT=.08333333*H                                                    QHSE 480
C                                                                       QHSE 490
C     INTEGRATION LOOP                                                  QHSE 500
      DO 2 I=2,NDIM                                                     QHSE 510
      SUM1=SUM2                                                         QHSE 520
     OSUM2=SUM2+HH*((Y(I-1)+Y(I))+HF*((FDY(I-1)-FDY(I))+                QHSE 530
     1              HT*(SDY(I-1)+SDY(I))))                              QHSE 540
    2 Z(I-1)=SUM1                                                       QHSE 550
    3 Z(NDIM)=SUM2                                                      QHSE 560
    4 RETURN                                                            QHSE 570
      END                                                               QHSE 580
```

```
C                                                                       DQHS  10
C     ..................................................................DQHS  20
C                                                                       DQHS  30
C        SUBROUTINE DQHSE                                               DQHS  40
C                                                                       DQHS  50
C        PURPOSE                                                        DQHS  60
C           TO COMPUTE THE VECTOR OF INTEGRAL VALUES FOR A GIVEN        DQHS  70
C           EQUIDISTANT TABLE OF FUNCTION, FIRST DERIVATIVE,            DQHS  80
C           AND SECOND DERIVATIVE VALUES.                               DQHS  90
C                                                                       DQHS 100
C        USAGE                                                          DQHS 110
C           CALL DQHSE (H,Y,FDY,SDY,Z,NDIM)                             DQHS 120
C                                                                       DQHS 130
C        DESCRIPTION OF PARAMETERS                                      DQHS 140
C           H      - DOUBLE PRECISION INCREMENT OF ARGUMENT VALUES.     DQHS 150
C           Y      - DOUBLE PRECISION INPUT VECTOR OF FUNCTION VALUES.  DQHS 160
C           FDY    - DOUBLE PRECISION INPUT VECTOR OF FIRST DERIVATIVE. DQHS 170
C           SDY    - DOUBLE PRECISION INPUT VECTOR OF SECOND DERIVATIVE.DQHS 180
C           Z      - RESULTING DOUBLE PRECISION VECTOR OF INTEGRAL     DQHS 190
C                    VALUES. Z MAY BE IDENTICAL WITH Y, FDY OR SDY.     DQHS 200
C           NDIM   - THE DIMENSION OF VECTORS Y,FDY,SDY,Z.              DQHS 210
C                                                                       DQHS 220
C        REMARKS                                                        DQHS 230
C           NO ACTION IN CASE NDIM LESS THAN 1.                         DQHS 240
C                                                                       DQHS 250
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQHS 260
C           NONE                                                        DQHS 270
C                                                                       DQHS 280
C        METHOD                                                         DQHS 290
C           BEGINNING WITH Z(1)=0, EVALUATION OF VECTOR Z IS DONE BY    DQHS 300
C           MEANS OF HERMITEAN SIXTH ORDER INTEGRATION FORMULA.         DQHS 310
C           FOR REFERENCE, SEE                                          DQHS 320
C           R.ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND       DQHS 330
C           PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963,     DQHS 340
C           PP.227-230.                                                 DQHS 350
C                                                                       DQHS 360
C     ..................................................................DQHS 370
C                                                                       DQHS 380
      SUBROUTINE DQHSE(H,Y,FDY,SDY,Z,NDIM)                              DQHS 390
C                                                                       DQHS 400
C                                                                       DQHS 410
      DIMENSION Y(1),FDY(1),SDY(1),Z(1)                                 DQHS 420
      DOUBLE PRECISION Y,FDY,SDY,Z,H,HH,HF,HT,SUM1,SUM2                 DQHS 430
C                                                                       DQHS 440
      SUM2=0.D0                                                         DQHS 450
      IF(NDIM-1)4,3,1                                                   DQHS 460
    1 HH=.5D0*H                                                         DQHS 470
      HF=.2D0*H                                                         DQHS 480
      HT=.08333333333333333D0*H                                         DQHS 490
C                                                                       DQHS 500
C     INTEGRATION LOOP                                                  DQHS 510
      DO 2 I=2,NDIM                                                     DQHS 520
      SUM1=SUM2                                                         DQHS 530
     OSUM2=SUM2+HH*((Y(I-1)+Y(I))+HF*((FDY(I-1)-FDY(I))+                DQHS 540
     1              HT*(SDY(I-1)+SDY(I))))                              DQHS 550
    2 Z(I-1)=SUM1                                                       DQHS 560
    3 Z(NDIM)=SUM2                                                      DQHS 570
    4 RETURN                                                            DQHS 580
      END                                                               DQHS 590
```

Subroutines QATR and DQATR

These subroutines perform the integration of a given function by the trapezoidal rule together with Romberg's extrapolation method in order to compute an approximation for:

$$y = \int_a^b f(x)\,dx \qquad (1)$$

Successively dividing the interval [a,b] into 2^i equidistant subintervals ($i = 0, 1, 2, \ldots$) and using the following notations:

$$h_i = \frac{b-a}{2^i}\ ;\ x_{i,k} = a + k \cdot h_i,\ f_{i,k} = f(x_{i,k})$$

$$(k = 0, 1, 2, \ldots, 2^i)$$

the trapezoidal rule gives approximations $T_{o,i}$ to the integral value y:

$$T_{o,i} = h_i \left\{ \sum_{k=0}^{2^i} f_{i,k} - \frac{1}{2}(f(a) + f(b)) \right\} \qquad (2)$$

Then the following can be written:

$$T_{o,i} = y + \sum_{r=1}^{\infty} C_{o,2r} \cdot h_i^{2r}$$

with unknown coefficients $C_{o,2r}$ which do not depend on i. Thus there is a truncation error of the order h_i^2.

Knowing two successive approximations, $T_{o,i}$ and $T_{o,i+1}$, an extrapolated value can be generated:

$$T_{1,i} = T_{o,\,i+1} + \frac{T_{o,i+1} - T_{o,i}}{2^2 - 1} \qquad (3)$$

This is a better approximation to y because:

$$T_{1,i} = y + \frac{1}{2^2-1} \sum_{r=1}^{\infty} C_{o,2r} (2^2 h_{i+1}^{2r} - h_i^{2r})$$

Noting that $2^2 h_{i+1}^2 - h_i^2 = 0$ and setting:

$$C_{1,2r} = \frac{1}{2^2-1} (2^2 - 2^{2r}) \cdot C_{o,2r}$$

$T_{1,i}$ becomes:

$$T_{1,i} = y + \sum_{r=2}^{\infty} C_{1,2r}\, h_{i+1}^{2r}$$

This gives a truncation error of the order h_{i+1}^4.

Knowing $T_{o,\,i+2}$ also, $T_{1,\,i+1}$ can be generated (formula 3), and:

$$T_{2,i} = T_{1,\,i+1} + \frac{T_{1,i+1} - T_{1,i}}{2^4 - 1} \qquad (4)$$

Thus:

$$T_{2,i} = y + \sum_{r=3}^{\infty} C_{2,2r} \cdot h_{i+2}^{2r}$$

with $C_{2,2r} = \frac{1}{2^4-1} (2^4 - 2^{2r})\, C_{1,2r}$

with a truncation error of the order h_{i+2}^6. Observe that the order of truncation error increases by 2 at each new extrapolation step.

The subroutine uses the scheme shown in Figure 25 for computation of T-values and generates the upward diagonal in the one-dimensional storage array AUX, using the general formula:

$$T_{k,j} = T_{k-1,j+1} + \frac{T_{k-1,j+1} - T_{k-1,j}}{2^{2k}-1} \quad (k+j=i,\ j = i-1, i-2, \ldots, 2, 1, 0) \qquad (5)$$

and storing:

$T_{o,i}$ into AUX (i+1)

$T_{1,i-1}$ into AUX (i) ....

$T_{k,0}$ into AUX (1)

Truncation error		$O(h_i^2)$	$O(h_i^4)$	$O(h_i^6)$	$O(h_i^8)$...
step length h_i	i \ j	0	1	2	3 ...
b-a	0	$T_{o,0}$	$T_{1,0}$	$T_{2,0}$	$T_{3,0}$ ...
$\frac{b-a}{2}$	1	$T_{o,1}$	$T_{1,1}$	$T_{2,1}$	⋮
$\frac{b-a}{4}$	2	$T_{o,2}$	$T_{1,2}$	⋮	
$\frac{b-a}{8}$	3	$T_{o,3}$	⋮		
⋮	⋮	⋮			

Figure 25. Computation of T-values (QATR)

The procedure stops if the difference between two successive values of AUX (1) is less than a given tolerance, or if the values of AUX (1) start oscillating, thus showing the influence of rounding errors.

For reference see S. Filippi, "Das Verfahren von Romberg-Stiefel-Bauer als Spezialfall des allgemeinen Prinzips von Richardson", Mathematik-Technik-Wirtschaft, vol. 11, iss. 2 (1964), pp. 49 - 54.

```
C
C     ...............................................................
C
C        SUBROUTINE QATR
C
C        PURPOSE
C           TO COMPUTE AN APPROXIMATION FOR INTEGRAL(FCT(X), SUMMED
C           OVER X FROM XL TO XU).
C
C        USAGE
C           CALL QATR (XL,XU,EPS,NDIM,FCT,Y,IER,AUX)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.
C
C        DESCRIPTION OF PARAMETERS
C           XL      - THE LOWER BOUND OF THE INTERVAL.
C           XU      - THE UPPER BOUND OF THE INTERVAL.
C           EPS     - THE UPPER BOUND OF THE ABSOLUTE ERROR.
C           NDIM    - THE DIMENSION OF THE AUXILIARY STORAGE ARRAY AUX.
C                     NDIM-1 IS THE MAXIMAL NUMBER OF BISECTIONS OF
C                     THE INTERVAL (XL,XU).
C           FCT     - THE NAME OF THE EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING APPROXIMATION FOR THE INTEGRAL VALUE.
C           IER     - A RESULTING ERROR PARAMETER.
C           AUX     - AN AUXILIARY STORAGE ARRAY WITH DIMENSION NDIM.
C
C        REMARKS
C           ERROR PARAMETER IER IS CODED IN THE FOLLOWING FORM
C           IER=0  - IT WAS POSSIBLE TO REACH THE REQUIRED ACCURACY.
C                    NO ERROR.
C           IER=1  - IT IS IMPOSSIBLE TO REACH THE REQUIRED ACCURACY
C                    BECAUSE OF ROUNDING ERRORS.
C           IER=2  - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE NDIM
C                    IS LESS THAN 5, OR THE REQUIRED ACCURACY COULD NOT
C                    BE REACHED WITHIN NDIM-1 STEPS. NDIM SHOULD BE
C                    INCREASED.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE CODED BY
C           THE USER. ITS ARGUMENT X SHOULD NOT BE DESTROYED.
C
C        METHOD
C           EVALUATION OF Y IS DONE BY MEANS OF TRAPEZOIDAL RULE IN
C           CONNECTION WITH ROMBERGS PRINCIPLE. ON RETURN Y CONTAINS
C           THE BEST POSSIBLE APPROXIMATION OF THE INTEGRAL VALUE AND
C           VECTOR AUX THE UPWARD DIAGONAL OF ROMBERG SCHEME.
C           COMPONENTS AUX(I) (I=1,2,...,IEND, WITH IEND LESS THAN OR
C           EQUAL TO NDIM) BECOME APPROXIMATIONS TO INTEGRAL VALUE WITH
C           DECREASING ACCURACY BY MULTIPLICATION WITH (XU-XL).
C           FOR REFERENCE, SEE
C           (1) FILIPPI, DAS VERFAHREN VON ROMBERG-STIEFEL-BAUER ALS
C               SPEZIALFALL DES ALLGEMEINEN PRINZIPS VON RICHARDSON,
C               MATHEMATIK-TECHNIK-WIRTSCHAFT, VOL.11, ISS.2 (1964),
C               PP.49-54.
C           (2) BAUER, ALGORITHM 60, CACM, VOL.4, ISS.6 (1961), PP.255.
C
C     ...............................................................
C
      SUBROUTINE QATR(XL,XU,EPS,NDIM,FCT,Y,IER,AUX)
C
C
      DIMENSION AUX(1)
C
C     PREPARATIONS OF ROMBERG-LOOP
      AUX(1)=.5*(FCT(XL)+FCT(XU))
      H=XU-XL
      IF(NDIM-1)8,8,1
    1 IF(H)2,10,2
C
C     NDIM IS GREATER THAN 1 AND H IS NOT EQUAL TO 0.
    2 HH=H
      E=EPS/ABS(H)
      DELT2=0.
      P=1.
      JJ=1
      DO 7 I=2,NDIM
      Y=AUX(1)
      DELT1=DELT2
      HD=HH
      HH=.5*HH
      P=.5*P
      X=XL+HH
      SM=0.
      DO 3 J=1,JJ
      SM=SM+FCT(X)
    3 X=X+HD
      AUX(I)=.5*AUX(I-1)+P*SM
C     A NEW APPROXIMATION OF INTEGRAL VALUE IS COMPUTED BY MEANS OF
C     TRAPEZOIDAL RULE.
C
C     START OF ROMBERGS EXTRAPOLATION METHOD.
      Q=1.
      JI=I-1
      DO 4 J=1,JI
      II=I-J
      Q=Q+Q
      Q=Q+Q
    4 AUX(II)=AUX(II+1)+(AUX(II+1)-AUX(II))/(Q-1.)
C     END OF ROMBERG-STEP
C
      DELT2=ABS(Y-AUX(1))
      IF(I-5)7,5,5
    5 IF(DELT2-E)10,10,6
    6 IF(DELT2-DELT1)7,11,11
    7 JJ=JJ+JJ
    8 IER=2
    9 Y=H*AUX(1)
      RETURN
   10 IER=0
      GO TO 9
   11 IER=1
      Y=H*Y
      RETURN
      END
```

```
C
C     ...............................................................
C
C        SUBROUTINE DQATR
C
C
C        PURPOSE
C           TO COMPUTE AN APPROXIMATION FOR INTEGRAL(FCT(X), SUMMED
C           OVER X FROM XL TO XU).
C
C        USAGE
C           CALL DQATR (XL,XU,EPS,NDIM,FCT,Y,IER,AUX)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT.
C
C        DESCRIPTION OF PARAMETERS
C           XL      - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.
C           XU      - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.
C           EPS     - SINGLE PRECISION UPPER BOUND OF THE ABSOLUTE ERROR.
C           NDIM    - THE DIMENSION OF THE AUXILIARY STORAGE ARRAY AUX.
C                     NDIM-1 IS THE MAXIMAL NUMBER OF BISECTIONS OF
C                     THE INTERVAL (XL,XU).
C           FCT     - THE NAME OF THE EXTERNAL DOUBLE PRECISION FUNCTION
C                     SUBPROGRAM USED.
C           Y       - RESULTING DOUBLE PRECISION APPROXIMATION FOR THE
C                     INTEGRAL VALUE.
C           IER     - A RESULTING ERROR PARAMETER.
C           AUX     - AUXILIARY DOUBLE PRECISION STORAGE ARRAY WITH
C                     DIMENSION NDIM.
C
C        REMARKS
C           ERROR PARAMETER IER IS CODED IN THE FOLLOWING FORM
C           IER=0  - IT WAS POSSIBLE TO REACH THE REQUIRED ACCURACY.
C                    NO ERROR.
C           IER=1  - IT IS IMPOSSIBLE TO REACH THE REQUIRED ACCURACY
C                    BECAUSE OF ROUNDING ERRORS.
C           IER=2  - IT WAS IMPOSSIBLE TO CHECK ACCURACY BECAUSE NDIM
C                    IS LESS THAN 5, OR THE REQUIRED ACCURACY COULD NOT
C                    BE REACHED WITHIN NDIM-1 STEPS. NDIM SHOULD BE
C                    INCREASED.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)
C           MUST BE CODED BY THE USER. ITS DOUBLE PRECISION ARGUMENT X
C           SHOULD NOT BE DESTROYED.
C
C        METHOD
C           EVALUATION OF Y IS DONE BY MEANS OF TRAPEZOIDAL RULE IN
C           CONNECTION WITH ROMBERGS PRINCIPLE. ON RETURN Y CONTAINS
C           THE BEST POSSIBLE APPROXIMATION OF THE INTEGRAL VALUE AND
C           VECTOR AUX THE UPWARD DIAGONAL OF ROMBERG SCHEME.
C           COMPONENTS AUX(I) (I=1,2,...,IEND, WITH IEND LESS THAN OR
C           EQUAL TO NDIM) BECOME APPROXIMATIONS TO INTEGRAL VALUE WITH
C           DECREASING ACCURACY BY MULTIPLICATION WITH (XU-XL).
C           FOR REFERENCE, SEE
C           (1) FILIPPI, DAS VERFAHREN VON ROMBERG-STIEFEL-BAUER ALS
C               SPEZIALFALL DES ALLGEMEINEN PRINZIPS VON RICHARDSON,
C               MATHEMATIK-TECHNIK-WIRTSCHAFT, VOL.11, ISS.2 (1964),
C               PP.49-54.
C           (2) BAUER, ALGORITHM 60, CACM, VOL.4, ISS.6 (1961), PP.255.
C
C     ...............................................................
C
      SUBROUTINE DQATR(XL,XU,EPS,NDIM,FCT,Y,IER,AUX)
C
C
      DIMENSION AUX(1)
      DOUBLE PRECISION AUX,XL,XU,X,Y,H,HH,HD,P,Q,SM,FCT
C
C     PREPARATIONS OF ROMBERG-LOOP
      AUX(1)=.5D0*(FCT(XL)+FCT(XU))
      H=XU-XL
      IF(NDIM-1)8,8,1
    1 IF(H)2,10,2
C
C     NDIM IS GREATER THAN 1 AND H IS NOT EQUAL TO 0.
    2 HH=H
      E=EPS/DABS(H)
      DELT2=0.
      P=1.D0
      JJ=1
      DO 7 I=2,NDIM
      Y=AUX(1)
      DELT1=DELT2
      HD=HH
      HH=.5D0*HH
      P=.5D0*P
      X=XL+HH
      SM=0.D0
      DO 3 J=1,JJ
      SM=SM+FCT(X)
    3 X=X+HD
      AUX(I)=.5D0*AUX(I-1)+P*SM
C     A NEW APPROXIMATION OF INTEGRAL VALUE IS COMPUTED BY MEANS OF
C     TRAPEZOIDAL RULE.
C
C     START OF ROMBERGS EXTRAPOLATION METHOD.
      Q=1.D0
      JI=I-1
      DO 4 J=1,JI
      II=I-J
      Q=Q+Q
      Q=Q+Q
    4 AUX(II)=AUX(II+1)+(AUX(II+1)-AUX(II))/(Q-1.D0)
C     END OF ROMBERG-STEP
C
      DELT2=DABS(Y-AUX(1))
      IF(I-5)7,5,5
    5 IF(DELT2-E)10,10,6
    6 IF(DELT2-DELT1)7,11,11
    7 JJ=JJ+JJ
    8 IER=2
    9 Y=H*AUX(1)
      RETURN
   10 IER=0
      GO TO 9
   11 IER=1
      Y=H*Y
      RETURN
      END
```

Subroutines QG2, QG3, . . . , QG10, DQG4, DQG8, DQG12, DQG16, DQG24, DQG32

These subroutines perform the integration of a given function by Gaussian quadrature formulas.

To compute:

$$y = \int_a^b f(x)\,dx$$

Gaussian quadrature formulas with n = 2,3, . . . , 10 (or n = 4,8,12,16,24,32) points are used. Transforming the range x = a . . . b into t = -1 . . . 1 by:

$$t = \frac{2x - (a+b)}{b - a}, \quad x = \frac{b-a}{2}\,t + \frac{b+a}{2}$$

the result is:

$$y = \frac{b-a}{2} \cdot \int_{-1}^{1} \varphi(t)\,dt \text{ with } \varphi(t) = f\left(\frac{b-a}{2} \cdot + \frac{b+a}{2}\right)$$

Using:

$$\int_{-1}^{1} \varphi(t)\,dt = \sum_{k=1}^{n} \left[A_k^{(n)}\,\varphi(t_k^{(n)})\right] (n = 2,3,\ldots)$$

with coefficients $A_k^{(n)}$ and nodes $t_k^{(n)}$ (note that $t_k^{(n)}$ are the roots of Legendre polynomials of degree n), the result is the approximations:

$$y_n = (b-a) \cdot \sum_{k=1}^{n} \left\{\frac{A_k^{(n)}}{2} \cdot f\left[(b-a) \cdot \frac{t_k^{(n)}}{2} + \frac{b+a}{2}\right]\right\} (n = 2,3,\ldots), \tag{1}$$

which are exact whenever f(x) is a polynomial up to the degree 2n-1.

The $A_k^{(n)}$ and $t_k^{(n)}$ are symmetric with respect to t = 0:

$$A_k^{(n)} = A_{n-k+1}^{(n)}, \quad t_k^{(n)} = -t_{n-k+1}^{(n)}$$

For reference see: V.J. Krylov, Approximate Calculation of Integrals. Macmillan, New York/London, 1962, pp. 100-111 and 337-340.

```
C                                                                       QG2   10
C     ..................................................................QG2   20
C                                                                       QG2   30
C        SUBROUTINE QG2                                                 QG2   40
C                                                                       QG2   50
C        PURPOSE                                                        QG2   60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    QG2   70
C                                                                       QG2   80
C        USAGE                                                          QG2   90
C           CALL QG2 (XL,XU,FCT,Y)                                      QG2  100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QG2  110
C                                                                       QG2  120
C        DESCRIPTION OF PARAMETERS                                      QG2  130
C           XL      - THE LOWER BOUND OF THE INTERVAL.                  QG2  140
C           XU      - THE UPPER BOUND OF THE INTERVAL.                  QG2  150
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QG2  160
C           Y       - THE RESULTING INTEGRAL VALUE.                     QG2  170
C                                                                       QG2  180
C        REMARKS                                                        QG2  190
C           NONE                                                        QG2  200
C                                                                       QG2  210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QG2  220
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QG2  230
C           BY THE USER.                                                QG2  240
C                                                                       QG2  250
C        METHOD                                                         QG2  260
C           EVALUATION IS DONE BY MEANS OF 2-POINT GAUSS QUADRATURE     QG2  270
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 3        QG2  280
C           EXACTLY.                                                    QG2  290
C           FOR REFERENCE, SEE                                          QG2  300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QG2  310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.   QG2  320
C                                                                       QG2  330
C     ..................................................................QG2  340
C                                                                       QG2  350
      SUBROUTINE QG2(XL,XU,FCT,Y)                                       QG2  360
C                                                                       QG2  370
C                                                                       QG2  380
      A=.5*(XU+XL)                                                      QG2  390
      B=XU-XL                                                           QG2  400
      Y=.2886751*B                                                      QG2  410
      Y=.5*B*(FCT(A+Y)+FCT(A-Y))                                        QG2  420
      RETURN                                                            QG2  430
      END                                                               QG2  440
```

```
C                                                                       QG3   10
C     ..................................................................QG3   20
C                                                                       QG3   30
C        SUBROUTINE QG3                                                 QG3   40
C                                                                       QG3   50
C        PURPOSE                                                        QG3   60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    QG3   70
C                                                                       QG3   80
C        USAGE                                                          QG3   90
C           CALL QG3 (XL,XU,FCT,Y)                                      QG3  100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QG3  110
C                                                                       QG3  120
C        DESCRIPTION OF PARAMETERS                                      QG3  130
C           XL      - THE LOWER BOUND OF THE INTERVAL.                  QG3  140
C           XU      - THE UPPER BOUND OF THE INTERVAL.                  QG3  150
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QG3  160
C           Y       - THE RESULTING INTEGRAL VALUE.                     QG3  170
C                                                                       QG3  180
C        REMARKS                                                        QG3  190
C           NONE                                                        QG3  200
C                                                                       QG3  210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QG3  220
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QG3  230
C           BY THE USER.                                                QG3  240
C                                                                       QG3  250
C        METHOD                                                         QG3  260
C           EVALUATION IS DONE BY MEANS OF 3-POINT GAUSS QUADRATURE     QG3  270
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 5        QG3  280
C           EXACTLY.                                                    QG3  290
C           FOR REFERENCE, SEE                                          QG3  300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QG3  310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.   QG3  320
C                                                                       QG3  330
C     ..................................................................QG3  340
C                                                                       QG3  350
      SUBROUTINE QG3(XL,XU,FCT,Y)                                       QG3  360
C                                                                       QG3  370
C                                                                       QG3  380
      A=.5*(XU+XL)                                                      QG3  390
      B=XU-XL                                                           QG3  400
      Y=.3872983*B                                                      QG3  410
      Y=.2777778*(FCT(A+Y)+FCT(A-Y))                                    QG3  420
      Y=B*(Y+.4444444*FCT(A))                                           QG3  430
      RETURN                                                            QG3  440
      END                                                               QG3  450
```

```
C
C     ..................................................................
C
C        SUBROUTINE QG4
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)
C
C        USAGE
C           CALL QG4 (XL,XU,FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           XL      - THE LOWER BOUND OF THE INTERVAL.
C           XU      - THE UPPER BOUND OF THE INTERVAL.
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 4-POINT GAUSS QUADRATURE
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 7
C           EXACTLY.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.
C
C     ..................................................................
C
      SUBROUTINE QG4(XL,XU,FCT,Y)
C
C
      A=.5*(XU+XL)
      B=XU-XL
      C=.4305682*B
      Y=.1739274*(FCT(A+C)+FCT(A-C))
      C=.1699905*B
      Y=B*(Y+.3260726*(FCT(A+C)+FCT(A-C)))
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QG5
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)
C
C        USAGE
C           CALL QG5 (XL,XU,FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           XL      - THE LOWER BOUND OF THE INTERVAL.
C           XU      - THE UPPER BOUND OF THE INTERVAL.
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 5-POINT GAUSS QUADRATURE
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 9
C           EXACTLY.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.
C
C     ..................................................................
C
      SUBROUTINE QG5(XL,XU,FCT,Y)
C
C
      A=.5*(XU+XL)
      B=XU-XL
      C=.4530899*B
      Y=.1184634*(FCT(A+C)+FCT(A-C))
      C=.2692347*B
      Y=Y+.2393143*(FCT(A+C)+FCT(A-C))
      Y=B*(Y+.2844444*FCT(A))
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DQG4
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)
C
C        USAGE
C           CALL DQG4 (XL,XU,FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           XL      - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.
C           XU      - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION
C                     SUBPROGRAM USED.
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)
C           MUST BE FURNISHED BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 4-POINT GAUSS QUADRATURE
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 7
C           EXACTLY. FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.
C
C     ..................................................................
C
      SUBROUTINE DQG4(XL,XU,FCT,Y)
C
C
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT
C
      A=.5D0*(XU+XL)
      B=XU-XL
      C=.43056815579702629D0*B
      Y=.173927422568726930D0*(FCT(A+C)+FCT(A-C))
      C=.16999052179242813D0*B
      Y=B*(Y+.326072577431273070D0*(FCT(A+C)+FCT(A-C)))
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QG6
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)
C
C        USAGE
C           CALL QG6 (XL,XU,FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           XL      - THE LOWER BOUND OF THE INTERVAL.
C           XU      - THE UPPER BOUND OF THE INTERVAL.
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 6-POINT GAUSS QUADRATURE
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 11
C           EXACTLY.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.
C
C     ..................................................................
C
      SUBROUTINE QG6(XL,XU,FCT,Y)
C
C
      A=.5*(XU+XL)
      B=XU-XL
      C=.4662348*B
      Y=.08566225*(FCT(A+C)+FCT(A-C))
      C=.3306047*B
      Y=Y+.1803808*(FCT(A+C)+FCT(A-C))
      C=.1193096*B
      Y=B*(Y+.2339570*(FCT(A+C)+FCT(A-C)))
      RETURN
      END
```

```
C                                                                       QG7   10
C     ..................................................................QG7   20
C                                                                       QG7   30
C        SUBROUTINE QG7                                                 QG7   40
C                                                                       QG7   50
C        PURPOSE                                                        QG7   60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    QG7   70
C                                                                       QG7   80
C        USAGE                                                          QG7   90
C           CALL QG7 (XL,XU,FCT,Y)                                      QG7  100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QG7  110
C                                                                       QG7  120
C        DESCRIPTION OF PARAMETERS                                      QG7  130
C           XL      - THE LOWER BOUND OF THE INTERVAL.                  QG7  140
C           XU      - THE UPPER BOUND OF THE INTERVAL.                  QG7  150
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QG7  160
C           Y       - THE RESULTING INTEGRAL VALUE.                     QG7  170
C                                                                       QG7  180
C        REMARKS                                                        QG7  190
C           NONE                                                        QG7  200
C                                                                       QG7  210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QG7  220
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QG7  230
C           BY THE USER.                                                QG7  240
C                                                                       QG7  250
C        METHOD                                                         QG7  260
C           EVALUATION IS DONE BY MEANS OF 7-POINT GAUSS QUADRATURE     QG7  270
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 13       QG7  280
C           EXACTLY.                                                    QG7  290
C           FOR REFERENCE, SEE                                          QG7  300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QG7  310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.   QG7  320
C                                                                       QG7  330
C     ..................................................................QG7  340
C                                                                       QG7  350
      SUBROUTINE QG7(XL,XU,FCT,Y)                                       QG7  360
C                                                                       QG7  370
C                                                                       QG7  380
      A=.5*(XU+XL)                                                      QG7  390
      B=XU-XL                                                           QG7  400
      C=.4745540*B                                                      QG7  410
      Y=.06474248*(FCT(A+C)+FCT(A-C))                                   QG7  420
      C=.3707656*B                                                      QG7  430
      Y=Y+.1398527*(FCT(A+C)+FCT(A-C))                                  QG7  440
      C=.2029226*B                                                      QG7  450
      Y=Y+.1909150*(FCT(A*C)+FCT(A-C))                                  QG7  460
      Y=B*(Y+.2089796*FCT(A))                                           QG7  470
      RETURN                                                            QG7  480
      END                                                               QG7  490
```

```
C                                                                       DQG8  10
C     ..................................................................DQG8  20
C                                                                       DQG8  30
C        SUBROUTINE DQG8                                                DQG8  40
C                                                                       DQG8  50
C        PURPOSE                                                        DQG8  60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    DQG8  70
C                                                                       DQG8  80
C        USAGE                                                          DQG8  90
C           CALL DQG8 (XL,XU,FCT,Y)                                     DQG8 100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DQG8 110
C                                                                       DQG8 120
C        DESCRIPTION OF PARAMETERS                                      DQG8 130
C           XL      - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.     DQG8 140
C           XU      - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.     DQG8 150
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DQG8 160
C                     SUBPROGRAM USED.                                  DQG8 170
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DQG8 180
C                                                                       DQG8 190
C        REMARKS                                                        DQG8 200
C           NONE                                                        DQG8 210
C                                                                       DQG8 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQG8 230
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DQG8 240
C           MUST BE FURNISHED BY THE USER.                              DQG8 250
C                                                                       DQG8 260
C        METHOD                                                         DQG8 270
C           EVALUATION IS DONE BY MEANS OF 8-POINT GAUSS QUADRATURE     DQG8 280
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 15       DQG8 290
C           EXACTLY. FOR REFERENCE, SEE                                 DQG8 300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           DQG8 310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.   DQG8 320
C                                                                       DQG8 330
C     ..................................................................DQG8 340
C                                                                       DQG8 350
      SUBROUTINE DQG8(XL,XU,FCT,Y)                                      DQG8 360
C                                                                       DQG8 370
C                                                                       DQG8 380
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT                                DQG8 390
C                                                                       DQG8 400
      A=.5D0*(XU+XL)                                                    DQG8 410
      B=XU-XL                                                           DQG8 420
      C=.4801449282487681 2D0*B                                         DQG8 430
      Y=.50614268145188130D-1*(FCT(A+C)+FCT(A-C))                       DQG8 440
      C=.39833323870681337D0*B                                          DQG8 450
      Y=Y+.11119051722668724D0*(FCT(A+C)+FCT(A-C))                      DQG8 460
      C=.26276620495816449D0*B                                          DQG8 470
      Y=Y+.15685332293894364D0*(FCT(A+C)+FCT(A-C))                      DQG8 480
      C=.91717321247824 90D-1*B                                         DQG8 490
      Y=B*(Y+.18134189168918099D0*(FCT(A+C)+FCT(A-C)))                  DQG8 500
      RETURN                                                            DQG8 510
      END                                                               DQG8 520
```

```
C                                                                       QG8   10
C     ..................................................................QG8   20
C                                                                       QG8   30
C        SUBROUTINE QG8                                                 QG8   40
C                                                                       QG8   50
C        PURPOSE                                                        QG8   60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    QG8   70
C                                                                       QG8   80
C        USAGE                                                          QG8   90
C           CALL QG8 (XL,XU,FCT,Y)                                      QG8  100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QG8  110
C                                                                       QG8  120
C        DESCRIPTION OF PARAMETERS                                      QG8  130
C           XL      - THE LOWER BOUND OF THE INTERVAL.                  QG8  140
C           XU      - THE UPPER BOUND OF THE INTERVAL.                  QG8  150
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QG8  160
C           Y       - THE RESULTING INTEGRAL VALUE.                     QG8  170
C                                                                       QG8  180
C        REMARKS                                                        QG8  190
C           NONE                                                        QG8  200
C                                                                       QG8  210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QG8  220
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QG8  230
C           BY THE USER.                                                QG8  240
C                                                                       QG8  250
C        METHOD                                                         QG8  260
C           EVALUATION IS DONE BY MEANS OF 8-POINT GAUSS QUADRATURE     QG8  270
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 15       QG8  280
C           EXACTLY.                                                    QG8  290
C           FOR REFERENCE, SEE                                          QG8  300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QG8  310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.   QG8  320
C                                                                       QG8  330
C     ..................................................................QG8  340
C                                                                       QG8  350
      SUBROUTINE QG8(XL,XU,FCT,Y)                                       QG8  360
C                                                                       QG8  370
C                                                                       QG8  380
      A=.5*(XU+XL)                                                      QG8  390
      B=XU-XL                                                           QG8  400
      C=.4801449*B                                                      QG8  410
      Y=.05061427*(FCT(A+C)+FCT(A-C))                                   QG8  420
      C=.3983332*B                                                      QG8  430
      Y=Y+.1111905*(FCT(A+C)+FCT(A-C))                                  QG8  440
      C=.2627662*B                                                      QG8  450
      Y=Y+.1568533*(FCT(A+C)+FCT(A-C))                                  QG8  460
      C=.09171732*B                                                     QG8  470
      Y=B*(Y+.1813419*(FCT(A+C)+FCT(A-C)))                              QG8  480
      RETURN                                                            QG8  490
      END                                                               QG8  500
```

```
C                                                                       QG9   10
C     ..................................................................QG9   20
C                                                                       QG9   30
C        SUBROUTINE QG9                                                 QG9   40
C                                                                       QG9   50
C        PURPOSE                                                        QG9   60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    QG9   70
C                                                                       QG9   80
C        USAGE                                                          QG9   90
C           CALL QG9 (XL,XU,FCT,Y)                                      QG9  100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QG9  110
C                                                                       QG9  120
C        DESCRIPTION OF PARAMETERS                                      QG9  130
C           XL      - THE LOWER BOUND OF THE INTERVAL.                  QG9  140
C           XU      - THE UPPER BOUND OF THE INTERVAL.                  QG9  150
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QG9  160
C           Y       - THE RESULTING INTEGRAL VALUE.                     QG9  170
C                                                                       QG9  180
C        REMARKS                                                        QG9  190
C           NONE                                                        QG9  200
C                                                                       QG9  210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QG9  220
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QG9  230
C           BY THE USER.                                                QG9  240
C                                                                       QG9  250
C        METHOD                                                         QG9  260
C           EVALUATION IS DONE BY MEANS OF 9-POINT GAUSS QUADRATURE     QG9  270
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 17       QG9  280
C           EXACTLY.                                                    QG9  290
C           FOR REFERENCE, SEE                                          QG9  300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QG9  310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.   QG9  320
C                                                                       QG9  330
C     ..................................................................QG9  340
C                                                                       QG9  350
      SUBROUTINE QG9(XL,XU,FCT,Y)                                       QG9  360
C                                                                       QG9  370
C                                                                       QG9  380
      A=.5*(XU+XL)                                                      QG9  390
      B=XU-XL                                                           QG9  400
      C=.4840801*B                                                      QG9  410
      Y=.04063719*(FCT(A+C)+FCT(A-C))                                   QG9  420
      C=.4180156*B                                                      QG9  430
      Y=Y+.09032408*(FCT(A+C)+FCT(A-C))                                 QG9  440
      C=.3066857*B                                                      QG9  450
      Y=Y+.1303053*(FCT(A+C)+FCT(A-C))                                  QG9  460
      C=.1621267*B                                                      QG9  470
      Y=Y+.1561735*(FCT(A+C)+FCT(A-C))                                  QG9  480
      Y=B*(Y+.1651197*FCT(A))                                           QG9  490
      RETURN                                                            QG9  500
      END                                                               QG9  510
```

```
C                                                                       QG10  10
C     ..................................................................QG10  20
C                                                                       QG10  30
C        SUBROUTINE QG10                                                QG10  40
C                                                                       QG10  50
C        PURPOSE                                                        QG10  60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    QG10  70
C                                                                       QG10  80
C        USAGE                                                          QG10  90
C           CALL QG10(XL,XU,FCT,Y)                                      QG10 100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QG10 110
C                                                                       QG10 120
C        DESCRIPTION OF PARAMETERS                                      QG10 130
C           XL     - THE LOWER BOUND OF THE INTERVAL.                   QG10 140
C           XU     - THE UPPER BOUND OF THE INTERVAL.                   QG10 150
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QG10 160
C           Y      - THE RESULTING INTEGRAL VALUE.                      QG10 170
C                                                                       QG10 180
C        REMARKS                                                        QG10 190
C           NONE                                                        QG10 200
C                                                                       QG10 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QG10 220
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QG10 230
C           BY THE USER.                                                QG10 240
C                                                                       QG10 250
C        METHOD                                                         QG10 260
C           EVALUATION IS DONE BY MEANS OF 10-POINT GAUSS QUADRATURE    QG10 270
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 19       QG10 280
C           EXACTLY.                                                    QG10 290
C           FOR REFERENCE, SEE                                          QG10 300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QG10 310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-338.   QG10 320
C                                                                       QG10 330
C     ..................................................................QG10 340
C                                                                       QG10 350
      SUBROUTINE QG10(XL,XU,FCT,Y)                                      QG10 360
C                                                                       QG10 370
C                                                                       QG10 380
      A=.5*(XU+XL)                                                      QG10 390
      B=XU-XL                                                           QG10 400
      C=.4869533*B                                                      QG10 410
      Y=.03333567*(FCT(A+C)+FCT(A-C))                                   QG10 420
      C=.4325317*B                                                      QG10 430
      Y=Y+.07472567*(FCT(A+C)+FCT(A-C))                                 QG10 440
      C=.3397048*B                                                      QG10 450
      Y=Y+.1095432*(FCT(A+C)+FCT(A-C))                                  QG10 460
      C=.2166977*B                                                      QG10 470
      Y=Y+.1346334*(FCT(A+C)+FCT(A-C))                                  QG10 480
      C=.07443717*B                                                     QG10 490
      Y=B*(Y+.1477621*(FCT(A+C)+FCT(A-C)))                              QG10 500
      RETURN                                                            QG10 510
      END                                                               QG10 520
```

```
C                                                                       DG12  10
C     ..................................................................DG12  20
C                                                                       DG12  30
C        SUBROUTINE DQG12                                               DG12  40
C                                                                       DG12  50
C        PURPOSE                                                        DG12  60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    DG12  70
C                                                                       DG12  80
C        USAGE                                                          DG12  90
C           CALL DQG12 (XL,XU,FCT,Y)                                    DG12 100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DG12 110
C                                                                       DG12 120
C        DESCRIPTION OF PARAMETERS                                      DG12 130
C           XL     - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.      DG12 140
C           XU     - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.      DG12 150
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DG12 160
C                    SUBPROGRAM USED.                                   DG12 170
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DG12 180
C                                                                       DG12 190
C        REMARKS                                                        DG12 200
C           NONE                                                        DG12 210
C                                                                       DG12 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DG12 230
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DG12 240
C           MUST BE FURNISHED BY THE USER.                              DG12 250
C                                                                       DG12 260
C        METHOD                                                         DG12 270
C           EVALUATION IS DONE BY MEANS OF 12-POINT GAUSS QUADRATURE    DG12 280
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 23       DG12 290
C           EXACTLY. FOR REFERENCE, SEE                                 DG12 300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           DG12 310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.   DG12 320
C                                                                       DG12 330
C     ..................................................................DG12 340
C                                                                       DG12 350
      SUBROUTINE DQG12(XL,XU,FCT,Y)                                     DG12 360
C                                                                       DG12 370
C                                                                       DG12 380
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT                                DG12 390
C                                                                       DG12 400
      A=.5D0*(XU+XL)                                                    DG12 410
      B=XU-XL                                                           DG12 420
      C=.4907803171233596 3D0*B                                         DG12 430
      Y=.2358766819325591 4D-1*(FCT(A+C)+FCT(A-C))                      DG12 440
      C=.4520586281852374 3D0*B                                         DG12 450
      Y=Y+.5346966299769215D-1*(FCT(A+C)+FCT(A-C))                      DG12 460
      C=.3849513370971523 4D0*B                                         DG12 470
      Y=Y+.8003916427167311D-1*(FCT(A+C)+FCT(A-C))                      DG12 480
      C=.2936589771433087 2D0*B                                         DG12 490
      Y=Y+.1015837133615329 6D0*(FCT(A+C)+FCT(A-C))                     DG12 500
      C=.1839157494990901 0D0*B                                         DG12 510
      Y=Y+.1167462682691774 0D0*(FCT(A+C)+FCT(A-C))                     DG12 520
      C=.6261670425573445 8D-1*B                                        DG12 530
      Y=B*(Y+.1245735229067013 9D0*(FCT(A+C)+FCT(A-C)))                 DG12 540
      RETURN                                                            DG12 550
      END                                                               DG12 560
```

```
C                                                                       DG16  10
C     ..................................................................DG16  20
C                                                                       DG16  30
C        SUBROUTINE DQG16                                               DG16  40
C                                                                       DG16  50
C        PURPOSE                                                        DG16  60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    DG16  70
C                                                                       DG16  80
C        USAGE                                                          DG16  90
C           CALL DQG16 (XL,XU,FCT,Y)                                    DG16 100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DG16 110
C                                                                       DG16 120
C        DESCRIPTION OF PARAMETERS                                      DG16 130
C           XL     - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.      DG16 140
C           XU     - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.      DG16 150
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DG16 160
C                    SUBPROGRAM USED.                                   DG16 170
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DG16 180
C                                                                       DG16 190
C        REMARKS                                                        DG16 200
C           NONE                                                        DG16 210
C                                                                       DG16 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DG16 230
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DG16 240
C           MUST BE FURNISHED BY THE USER.                              DG16 250
C                                                                       DG16 260
C        METHOD                                                         DG16 270
C           EVALUATION IS DONE BY MEANS OF 16-POINT GAUSS QUADRATURE    DG16 280
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 31       DG16 290
C           EXACTLY. FOR REFERENCE, SEE                                 DG16 300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           DG16 310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.   DG16 320
C                                                                       DG16 330
C     ..................................................................DG16 340
C                                                                       DG16 350
      SUBROUTINE DQG16(XL,XU,FCT,Y)                                     DG16 360
C                                                                       DG16 370
C                                                                       DG16 380
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT                                DG16 390
C                                                                       DG16 400
      A=.5D0*(XU+XL)                                                    DG16 410
      B=XU-XL                                                           DG16 420
      C=.4947004674958249 7D0*B                                         DG16 430
      Y=.1357622970587704 7D-1*(FCT(A+C)+FCT(A-C))                      DG16 440
      C=.4722875115366162 9D0*B                                         DG16 450
      Y=Y+.3112676196923946D-1*(FCT(A+C)+FCT(A-C))                      DG16 460
      C=.4328156011939158 7D0*B                                         DG16 470
      Y=Y+.4757925584124639 2D-1*(FCT(A+C)+FCT(A-C))                    DG16 480
      C=.3777022041775015 2D0*B                                         DG16 490
      Y=Y+.6231448562776693 6D-1*(FCT(A+C)+FCT(A-C))                    DG16 500
      C=.3089381222013218 7D0*B                                         DG16 510
      Y=Y+.7479799440828837D-1*(FCT(A+C)+FCT(A-C))                      DG16 520
      C=.2290083888286136 9D0*B                                         DG16 530
      Y=Y+.8457825969750127D-1*(FCT(A+C)+FCT(A-C))                      DG16 540
      C=.1408017753896294 6D0*B                                         DG16 550
      Y=Y+.9130170752246179D-1*(FCT(A+C)+FCT(A-C))                      DG16 560
      C=.4750625491881872 0D-1*B                                        DG16 570
      Y=B*(Y+.9472530522753425D-1*(FCT(A+C)+FCT(A-C)))                  DG16 580
      RETURN                                                            DG16 590
      END                                                               DG16 600
```

```
C                                                                       DG24  10
C     ..................................................................DG24  20
C                                                                       DG24  30
C        SUBROUTINE DQG24                                               DG24  40
C                                                                       DG24  50
C        PURPOSE                                                        DG24  60
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)    DG24  70
C                                                                       DG24  80
C        USAGE                                                          DG24  90
C           CALL DQG24 (XL,XU,FCT,Y)                                    DG24 100
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DG24 110
C                                                                       DG24 120
C        DESCRIPTION OF PARAMETERS                                      DG24 130
C           XL     - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.      DG24 140
C           XU     - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.      DG24 150
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DG24 160
C                    SUBPROGRAM USED.                                   DG24 170
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DG24 180
C                                                                       DG24 190
C        REMARKS                                                        DG24 200
C           NONE                                                        DG24 210
C                                                                       DG24 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DG24 230
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DG24 240
C           MUST BE FURNISHED BY THE USER.                              DG24 250
C                                                                       DG24 260
C        METHOD                                                         DG24 270
C           EVALUATION IS DONE BY MEANS OF 24-POINT GAUSS QUADRATURE    DG24 280
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 47       DG24 290
C           EXACTLY. FOR REFERENCE, SEE                                 DG24 300
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           DG24 310
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.   DG24 320
C                                                                       DG24 330
C     ..................................................................DG24 340
C                                                                       DG24 350
      SUBROUTINE DQG24(XL,XU,FCT,Y)                                     DG24 360
C                                                                       DG24 370
C                                                                       DG24 380
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT                                DG24 390
C                                                                       DG24 400
      A=.5D0*(XU+XL)                                                    DG24 410
      B=XU-XL                                                           DG24 420
      C=.4975936099985106 8D0*B                                         DG24 430
      Y=.6170614899993599 8D-2*(FCT(A+C)+FCT(A-C))                      DG24 440
      C=.4873642779856547 5D0*B                                         DG24 450
      Y=Y+.1426569431446683 2D-1*(FCT(A+C)+FCT(A-C))                    DG24 460
      C=.4691372760013663 8D0*B                                         DG24 470
      Y=Y+.2213871940870990 3D-1*(FCT(A+C)+FCT(A-C))                    DG24 480
      C=.4432077635022005 2D0*B                                         DG24 490
      Y=Y+.2964929245771839 0D-1*(FCT(A+C)+FCT(A-C))                    DG24 500
      C=.4100009929869514 6D0*B                                         DG24 510
      Y=Y+.3667320470554015 3D-1*(FCT(A+C)+FCT(A-C))                    DG24 520
      C=.3700620957892771 8D0*B                                         DG24 530
      Y=Y+.4309508076597663 8D-1*(FCT(A+C)+FCT(A-C))                    DG24 540
      C=.3240468259684877 8D0*B                                         DG24 550
      Y=Y+.4880932605205694 4D-1*(FCT(A+C)+FCT(A-C))                    DG24 560
      C=.2727107356944197 7D0*B                                         DG24 570
      Y=Y+.5372213505798281 7D-1*(FCT(A+C)+FCT(A-C))                    DG24 580
      C=.2168967538130225 7D0*B                                         DG24 590
      Y=Y+.5775283402686280 1D-1*(FCT(A+C)+FCT(A-C))                    DG24 600
      C=.1575213398480816 9D0*B                                         DG24 610
      Y=Y+.6083523646390169 6D-1*(FCT(A+C)+FCT(A-C))                    DG24 620
      C=.9555943373680815D-1*B                                          DG24 630
      Y=Y+.6291872817341414 8D-1*(FCT(A+C)+FCT(A-C))                    DG24 640
      C=.3202844643130281 3D-1*B                                        DG24 650
      Y=B*(Y+.6396909767337607 8D-1*(FCT(A+C)+FCT(A-C)))                DG24 660
      RETURN                                                            DG24 670
      END                                                               DG24 680
```

```
C
C     ..................................................................
C
C        SUBROUTINE DQG32
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(FCT(X), SUMMED OVER X FROM XL TO XU)
C
C        USAGE
C           CALL DQG32 (XL,XU,FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           XL      - DOUBLE PRECISION LOWER BOUND OF THE INTERVAL.
C           XU      - DOUBLE PRECISION UPPER BOUND OF THE INTERVAL.
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION
C                     SUBPROGRAM USED.
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)
C           MUST BE FURNISHED BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 32-POINT GAUSS QUADRATURE
C           FORMULA, WHICH INTEGRATES POLYNOMIALS UP TO DEGREE 63
C           EXACTLY. FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.100-111 AND 337-340.
C
C     ..................................................................
C
      SUBROUTINE DQG32(XL,XU,FCT,Y)
C
C
      DOUBLE PRECISION XL,XU,Y,A,B,C,FCT
C
      A=.5D0*(XU+XL)
      B=XU-XL
      C=.49863193092474078D0*B
      Y=.35093050047350483D-2*(FCT(A+C)+FCT(A-C))
      C=.49280575577263417D0*B
      Y=Y+.81371973654528350D-2*(FCT(A+C)+FCT(A-C))
      C=.48238112779375322D0*B
      Y=Y+.12696032654631030D-1*(FCT(A+C)+FCT(A-C))
      C=.46745303796886984D0*B
      Y=Y+.17136931456510717D-1*(FCT(A+C)+FCT(A-C))
      C=.44816057788302606D0*B
      Y=Y+.21417949011113340D-1*(FCT(A+C)+FCT(A-C))
      C=.42468380686628499D0*B
      Y=Y+.25499029631188088D-1*(FCT(A+C)+FCT(A-C))
      C=.39724189798397120D0*B
      Y=Y+.29342046739267774D-1*(FCT(A+C)+FCT(A-C))
      C=.36609105937014484D0*B
      Y=Y+.32911111388180923D-1*(FCT(A+C)+FCT(A-C))
      C=.33152213346510760D0*B
      Y=Y+.36172897054424253D-1*(FCT(A+C)+FCT(A-C))
      C=.29385787862038116D0*B
      Y=Y+.39096947893535153D-1*(FCT(A+C)+FCT(A-C))
      C=.25344995446611470D0*B
      Y=Y+.41655962113473378D-1*(FCT(A+C)+FCT(A-C))
      C=.21067563806531767D0*B
      Y=Y+.43826046502201906D-1*(FCT(A+C)+FCT(A-C))
      C=.16593430114106382D0*B
      Y=Y+.45586939347881942D-1*(FCT(A+C)+FCT(A-C))
      C=.11964368112606854D0*B
      Y=Y+.46922199540402283D-1*(FCT(A+C)+FCT(A-C))
      C=.72235980791398250-1*B
      Y=Y+.47819360396374430D-1*(FCT(A+C)+FCT(A-C))
      C=.24153832843869158D-1*B
      Y=B*(Y+.48270044257363900D-1*(FCT(A+C)+FCT(A-C)))
      RETURN
      END
```

Subroutines QL2, QL3, . . . , QL10; DQL4, DQL8, DQL12, DQL16, DQL24, DQL32

These subroutines perform the integration of a given function by Gaussian-Laguerre quadrature formulas:

To compute:

$$y = \int_0^{\infty} e^{-x} f(x)\, dx$$

Gaussian-Laguerre quadrature formulas with $n = 2,3,\ldots, 10$ (or $n = 4,8,12,16,24,32$) points are used to get the approximations:

$$y_n = \sum_{k=1}^{n} \left[A_k^{(n)} \cdot f(x_k^{(n)}) \right] \quad (n = 2,3,\ldots) \qquad (1)$$

which are exact whenever f(x) is a polynomial up to the degree 2n-1. Note that nodes $x_k^{(n)}$ are the roots of the Laguerre polynomials $L_n(x)$ of degree n.

For reference see:

(1) H.E. Salzer/R. Zucker, "Table of Zeros and Weight Factors of the First Fifteen Laguerre Polynomials". Bul. Amer. Math. Soc., vol 55 (1949), pp 1004-1012.

(2) V.I. Krylov, Approximate Calculation of Integrals. Macmillan, New York/London, 1962, pp. 130-132 and 347-352.

(3) Shao/Chen/Frank, "Tables of Zeros and Gaussian Weights of Certain Associated Laguerre Polynomials and the Related Generalized Hermite Polynomials", IBM Technical Report TR 00.1100 (March, 1964) pp. 24-25.


```
C
C     ..................................................................
C
C        SUBROUTINE QL2
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0
C                              TO INFINITY).
C
C        USAGE
C           CALL QL2 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 2-POINT GAUSSIAN-LAGUERRE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 3.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.
C
C     ..................................................................
C
      SUBROUTINE QL2(FCT,Y)
C
C
      X=3.414214
      Y=.1464466*FCT(X)
      X=.5857864
      Y=Y+.8535534*FCT(X)
      RETURN
      END
```

```
C                                                                       QL3   10
C     ..................................................................QL3   20
C                                                                       QL3   30
C        SUBROUTINE QL3                                                 QL3   40
C                                                                       QL3   50
C        PURPOSE                                                        QL3   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0    QL3   70
C                                 TO INFINITY).                         QL3   80
C                                                                       QL3   90
C        USAGE                                                          QL3  100
C           CALL QL3 (FCT,Y)                                            QL3  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QL3  120
C                                                                       QL3  130
C        DESCRIPTION OF PARAMETERS                                      QL3  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QL3  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QL3  160
C                                                                       QL3  170
C        REMARKS                                                        QL3  180
C           NONE                                                        QL3  190
C                                                                       QL3  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QL3  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QL3  220
C           BY THE USER.                                                QL3  230
C                                                                       QL3  240
C        METHOD                                                         QL3  250
C           EVALUATION IS DONE BY MEANS OF 3-POINT GAUSSIAN-LAGUERRE    QL3  260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QL3  270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 5.                      QL3  280
C           FOR REFERENCE, SEE                                          QL3  290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QL3  300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.   QL3  310
C                                                                       QL3  320
C     ..................................................................QL3  330
C                                                                       QL3  340
      SUBROUTINE QL3(FCT,Y)                                             QL3  350
C                                                                       QL3  360
C                                                                       QL3  370
      X=6.289945                                                        QL3  380
      Y=.01038926*FCT(X)                                                QL3  390
      X=2.294280                                                        QL3  400
      Y=Y+.2785177*FCT(X)                                               QL3  410
      X=.4157746                                                        QL3  420
      Y=Y+.7110930*FCT(X)                                               QL3  430
      RETURN                                                            QL3  440
      END                                                               QL3  450
```

```
C                                                                       DQL4  10
C     ..................................................................DQL4  20
C                                                                       DQL4  30
C        SUBROUTINE DQL4                                                DQL4  40
C                                                                       DQL4  50
C        PURPOSE                                                        DQL4  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X           DQL4  70
C                                 FROM 0 TO INFINITY).                  DQL4  80
C                                                                       DQL4  90
C        USAGE                                                          DQL4 100
C           CALL DQL4 (FCT,Y)                                           DQL4 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DQL4 120
C                                                                       DQL4 130
C        DESCRIPTION OF PARAMETERS                                      DQL4 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DQL4 150
C                    SUBPROGRAM USED.                                   DQL4 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DQL4 170
C                                                                       DQL4 180
C        REMARKS                                                        DQL4 190
C           NONE                                                        DQL4 200
C                                                                       DQL4 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQL4 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DQL4 230
C           MUST BE FURNISHED BY THE USER.                              DQL4 240
C                                                                       DQL4 250
C        METHOD                                                         DQL4 260
C           EVALUATION IS DONE BY MEANS OF 4-POINT GAUSSIAN-LAGUERRE    DQL4 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,               DQL4 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 7.             DQL4 290
C           FOR REFERENCE, SEE                                          DQL4 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DQL4 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DQL4 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DQL4 330
C           TR00.1100 (MARCH 1964), PP.24-25.                           DQL4 340
C                                                                       DQL4 350
C     ..................................................................DQL4 360
C                                                                       DQL4 370
      SUBROUTINE DQL4(FCT,Y)                                            DQL4 380
C                                                                       DQL4 390
C                                                                       DQL4 400
      DOUBLE PRECISION X,Y,FCT                                          DQL4 410
C                                                                       DQL4 420
      X=.9395070912301133D1                                             DQL4 430
      Y=.5392947055613274 5D-3*FCT(X)                                   DQL4 440
      X=.4536620296921128D1                                             DQL4 450
      Y=Y+.3888790851500538 4D-1*FCT(X)                                 DQL4 460
      X=.1745761101158346601                                            DQL4 470
      Y=Y+.3574186924377996 9D0*FCT(X)                                  DQL4 480
      X=.3225476896193923 1D0                                           DQL4 490
      Y=Y+.6031541C434163360D0*FCT(X)                                   DQL4 500
      RETURN                                                            DQL4 510
      END                                                               DQL4 520
```

```
C                                                                       QL4   10
C     ..................................................................QL4   20
C                                                                       QL4   30
C        SUBROUTINE QL4                                                 QL4   40
C                                                                       QL4   50
C        PURPOSE                                                        QL4   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0    QL4   70
C                                 TO INFINITY).                         QL4   80
C                                                                       QL4   90
C        USAGE                                                          QL4  100
C           CALL QL4 (FCT,Y)                                            QL4  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QL4  120
C                                                                       QL4  130
C        DESCRIPTION OF PARAMETERS                                      QL4  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QL4  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QL4  160
C                                                                       QL4  170
C        REMARKS                                                        QL4  180
C           NONE                                                        QL4  190
C                                                                       QL4  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QL4  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QL4  220
C           BY THE USER.                                                QL4  230
C                                                                       QL4  240
C        METHOD                                                         QL4  250
C           EVALUATION IS DONE BY MEANS OF 4-POINT GAUSSIAN-LAGUERRE    QL4  260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QL4  270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 7.                      QL4  280
C           FOR REFERENCE, SEE                                          QL4  290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QL4  300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.   QL4  310
C                                                                       QL4  320
C     ..................................................................QL4  330
C                                                                       QL4  340
      SUBROUTINE QL4(FCT,Y)                                             QL4  350
C                                                                       QL4  360
C                                                                       QL4  370
      X=9.395071                                                        QL4  380
      Y=.5392947E-3*FCT(X)                                              QL4  390
      X=4.536620                                                        QL4  400
      Y=Y+.03888791*FCT(X)                                              QL4  410
      X=1.745761                                                        QL4  420
      Y=Y+.3574187*FCT(X)                                               QL4  430
      X=.3225477                                                        QL4  440
      Y=Y+.6031541*FCT(X)                                               QL4  450
      RETURN                                                            QL4  460
      END                                                               QL4  470
```

```
C                                                                       QL5   10
C     ..................................................................QL5   20
C                                                                       QL5   30
C        SUBROUTINE QL5                                                 QL5   40
C                                                                       QL5   50
C        PURPOSE                                                        QL5   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0    QL5   70
C                                 TO INFINITY).                         QL5   80
C                                                                       QL5   90
C        USAGE                                                          QL5  100
C           CALL QL5 (FCT,Y)                                            QL5  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QL5  120
C                                                                       QL5  130
C        DESCRIPTION OF PARAMETERS                                      QL5  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QL5  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QL5  160
C                                                                       QL5  170
C        REMARKS                                                        QL5  180
C           NONE                                                        QL5  190
C                                                                       QL5  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QL5  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QL5  220
C           BY THE USER.                                                QL5  230
C                                                                       QL5  240
C        METHOD                                                         QL5  250
C           EVALUATION IS DONE BY MEANS OF 5-POINT GAUSSIAN-LAGUERRE    QL5  260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QL5  270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 9.                      QL5  280
C           FOR REFERENCE, SEE                                          QL5  290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QL5  300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.   QL5  310
C                                                                       QL5  320
C     ..................................................................QL5  330
C                                                                       QL5  340
      SUBROUTINE QL5(FCT,Y)                                             QL5  350
C                                                                       QL5  360
C                                                                       QL5  370
      X=12.64080                                                        QL5  380
      Y=.2336997E-4*FCT(X)                                              QL5  390
      X=7.085810                                                        QL5  400
      Y=Y+.3611759E-2*FCT(X)                                            QL5  410
      X=3.596426                                                        QL5  420
      Y=Y+.07594245*FCT(X)                                              QL5  430
      X=1.413403                                                        QL5  440
      Y=Y+.3986668*FCT(X)                                               QL5  450
      X=.2635603                                                        QL5  460
      Y=Y+.5217556*FCT(X)                                               QL5  470
      RETURN                                                            QL5  480
      END                                                               QL5  490
```

```
C
C     ..................................................................
C
C        SUBROUTINE QL6
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0
C                                 TO INFINITY).
C
C        USAGE
C           CALL QL6 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 6-POINT GAUSSIAN-LAGUERRE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 11.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.
C
C     ..................................................................
C
      SUBROUTINE QL6(FCT,Y)
C
C
      X=15.98287
      Y=.8985479E-6*FCT(X)
      X=9.837467
      Y=Y+.2610172E-3*FCT(X)
      X=5.775144
      Y=Y+.0103992O*FCT(X)
      X=2.992736
      Y=Y+.1133734*FCT(X)
      X=1.188932
      Y=Y+.4170008*FCT(X)
      X=.2228466
      Y=Y+.4589647*FCT(X)
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QL8
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0
C                                 TO INFINITY).
C
C        USAGE
C           CALL QL8 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 8-POINT GAUSSIAN-LAGUERRE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 15.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.
C
C     ..................................................................
C
      SUBROUTINE QL8(FCT,Y)
C
C
      X=22.86313
      Y=.1048001E-8*FCT(X)
      X=15.74068
      Y=Y+.8485747E-6*FCT(X)
      X=10.75852
      Y=Y+.9076509E-4*FCT(X)
      X=7.045905
      Y=Y+.2794536E-2*FCT(X)
      X=4.266700
      Y=Y+.03334349*FCT(X)
      X=2.251087
      Y=Y+.1757950*FCT(X)
      X=.9037018
      Y=Y+.4187868*FCT(X)
      X=.1702796
      Y=Y+.3691886*FCT(X)
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QL7
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0
C                                 TO INFINITY).
C
C        USAGE
C           CALL QL7 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y       - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 7-POINT GAUSSIAN-LAGUERRE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 13.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.
C
C     ..................................................................
C
      SUBROUTINE QL7(FCT,Y)
C
C
      X=19.39573
      Y=.3170315E-7*FCT(X)
      X=12.73418
      Y=Y+.1586546E-4*FCT(X)
      X=8.182153
      Y=Y+.1074010E-2*FCT(X)
      X=4.900353
      Y=Y+.02063351*FCT(X)
      X=2.567877
      Y=Y+.1471263*FCT(X)
      X=1.026665
      Y=Y+.4218313*FCT(X)
      X=.1930437
      Y=Y+.4093190*FCT(X)
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DQL8
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X
C                                 FROM 0 TO INFINITY).
C
C        USAGE
C           CALL DQL8 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION
C                     SUBPROGRAM USED.
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)
C           MUST BE FURNISHED BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 8-POINT GAUSSIAN-LAGUERRE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 15.
C           FOR REFERENCE, SEE
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT
C           TR00.1100 (MARCH 1964), PP.24-25.
C
C     ..................................................................
C
      SUBROUTINE DQL8(FCT,Y)
C
C
      DOUBLE PRECISION X,Y,FCT
C
      X=.2286313173688926402
      Y=.1048001174871510 4D-8*FCT(X)
      X=.1574067864127800502
      Y=Y+.8485746716272532D-6*FCT(X)
      X=.1075851601018099502
      Y=Y+.9076508773358213D-4*FCT(X)
      X=.7045905402393465701
      Y=Y+.2794536235225617250-2*FCT(X)
      X=.4266700170287658801
      Y=Y+.3334349226125652D-1*FCT(X)
      X=.2251086629866130701
      Y=Y+.1757949866371718100*FCT(X)
      X=.9037017769979790D0
      Y=Y+.4187867808143429600*FCT(X)
      X=.1702796323051010000
      Y=Y+.3691885893416375300*FCT(X)
      RETURN
      END
```

```
C                                                                       QL9   10
C     ..................................................................QL9   20
C                                                                       QL9   30
C        SUBROUTINE QL9                                                 QL9   40
C                                                                       QL9   50
C        PURPOSE                                                        QL9   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0    QL9   70
C                                 TO INFINITY).                         QL9   80
C                                                                       QL9   90
C        USAGE                                                          QL9  100
C           CALL QL9 (FCT,Y)                                            QL9  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QL9  120
C                                                                       QL9  130
C        DESCRIPTION OF PARAMETERS                                      QL9  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QL9  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QL9  160
C                                                                       QL9  170
C        REMARKS                                                        QL9  180
C           NONE                                                        QL9  190
C                                                                       QL9  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QL9  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QL9  220
C           BY THE USER.                                                QL9  230
C                                                                       QL9  240
C        METHOD                                                         QL9  250
C           EVALUATION IS DONE BY MEANS OF 9-POINT GAUSSIAN-LAGUERRE    QL9  260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QL9  270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 17.                     QL9  280
C           FOR REFERENCE, SEE                                          QL9  290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QL9  300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.   QL9  310
C                                                                       QL9  320
C     ..................................................................QL9  330
C                                                                       QL9  340
      SUBROUTINE QL9(FCT,Y)                                             QL9  350
C                                                                       QL9  360
C                                                                       QL9  370
      X=26.37407                                                        QL9  380
      Y=.3290874E-10*FCT(X)                                             QL9  390
      X=18.83360                                                        QL9  400
      Y=Y+.4110769E-7*FCT(X)                                            QL9  410
      X=13.46624                                                        QL9  420
      Y=Y+.6592123E-5*FCT(X)                                            QL9  430
      X=9.372985                                                        QL9  440
      Y=Y+.3052498E-3*FCT(X)                                            QL9  450
      X=6.204957                                                        QL9  460
      Y=Y+.0055996 27*FCT(X)                                            QL9  470
      X=3.783474                                                        QL9  480
      Y=Y+.04746056*FCT(X)                                              QL9  490
      X=2.005135                                                        QL9  500
      Y=Y+.1992875*FCT(X)                                               QL9  510
      X=.8072200                                                        QL9  520
      Y=Y+.4112140*FCT(X)                                               QL9  530
      X=.1523222                                                        QL9  540
      Y=Y+.3361264*FCT(X)                                               QL9  550
      RETURN                                                            QL9  560
      END                                                               QL9  570
```

```
C                                                                       QL10  10
C     ..................................................................QL10  20
C                                                                       QL10  30
C        SUBROUTINE QL10                                                QL10  40
C                                                                       QL10  50
C        PURPOSE                                                        QL10  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X FROM 0    QL10  70
C                                 TO INFINITY).                         QL10  80
C                                                                       QL10  90
C        USAGE                                                          QL10 100
C           CALL QL10(FCT,Y)                                            QL10 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QL10 120
C                                                                       QL10 130
C        DESCRIPTION OF PARAMETERS                                      QL10 140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QL10 150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QL10 160
C                                                                       QL10 170
C        REMARKS                                                        QL10 180
C           NONE                                                        QL10 190
C                                                                       QL10 200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QL10 210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QL10 220
C           BY THE USER.                                                QL10 230
C                                                                       QL10 240
C        METHOD                                                         QL10 250
C           EVALUATION IS DONE BY MEANS OF 10-POINT GAUSSIAN-LAGUERRE   QL10 260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QL10 270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 19.                     QL10 280
C           FOR REFERENCE, SEE                                          QL10 290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QL10 300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.130-132 AND 347-352.   QL10 310
C                                                                       QL10 320
C     ..................................................................QL10 330
C                                                                       QL10 340
      SUBROUTINE QL10(FCT,Y)                                            QL10 350
C                                                                       QL10 360
C                                                                       QL10 370
      X=29.92070                                                        QL10 380
      Y=.9911827E-12*FCT(X)                                             QL10 390
      X=21.99659                                                        QL10 400
      Y=Y+.1839565E-8*FCT(X)                                            QL10 410
      X=16.27926                                                        QL10 420
      Y=Y+.4249314E-6*FCT(X)                                            QL10 430
      X=11.84379                                                        QL10 440
      Y=Y+.2825923E-4*FCT(X)                                            QL10 450
      X=8.330153                                                        QL10 460
      Y=Y+.7530084E-3*FCT(X)                                            QL10 470
      X=5.552496                                                        QL10 480
      Y=Y+.009501517*FCT(X)                                             QL10 490
      X=3.401434                                                        QL10 500
      Y=Y+.06208746*FCT(X)                                              QL10 510
      X=1.808343                                                        QL10 520
      Y=Y+.2180683*FCT(X)                                               QL10 530
      X=.7294545                                                        QL10 540
      Y=Y+.4011199*FCT(X)                                               QL10 550
      X=.1377935                                                        QL10 560
      Y=Y+.3084411*FCT(X)                                               QL10 570
      RETURN                                                            QL10 580
      END                                                               QL10 590
```

```
C                                                                       DL12  10
C     ..................................................................DL12  20
C                                                                       DL12  30
C        SUBROUTINE DQL12                                               DL12  40
C                                                                       DL12  50
C        PURPOSE                                                        DL12  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X           DL12  70
C                                 FROM 0 TO INFINITY).                  DL12  80
C                                                                       DL12  90
C        USAGE                                                          DL12 100
C           CALL DQL12 (FCT,Y)                                          DL12 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DL12 120
C                                                                       DL12 130
C        DESCRIPTION OF PARAMETERS                                      DL12 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DL12 150
C                    SUBPROGRAM USED.                                   DL12 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DL12 170
C                                                                       DL12 180
C        REMARKS                                                        DL12 190
C           NONE                                                        DL12 200
C                                                                       DL12 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DL12 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DL12 230
C           MUST BE FURNISHED BY THE USER.                              DL12 240
C                                                                       DL12 250
C        METHOD                                                         DL12 260
C           EVALUATION IS DONE BY MEANS OF 12-POINT GAUSSIAN-LAGUERRE   DL12 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,               DL12 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 23.            DL12 290
C           FOR REFERENCE, SEE                                          DL12 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DL12 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DL12 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DL12 330
C           TR00.1100 (MARCH 1964), PP.24-25.                           DL12 340
C                                                                       DL12 350
C     ..................................................................DL12 360
C                                                                       DL12 370
      SUBROUTINE DQL12(FCT,Y)                                           DL12 380
C                                                                       DL12 390
C                                                                       DL12 400
      DOUBLE PRECISION X,Y,FCT                                          DL12 410
C                                                                       DL12 420
      X=.3709912104446692 0D2                                           DL12 430
      Y=.8148077467426242D-15*FCT(X)                                    DL12 440
      X=.2848796725098400 0D2                                           DL12 450
      Y=Y+.3061601635035 02080D-11*FCT(X)                               DL12 460
      X=.2215109037939700 6D2                                           DL12 470
      Y=Y+.1342391030515 0041D-8*FCT(X)                                 DL12 480
      X=.1711685518746225 6D2                                           DL12 490
      Y=Y+.1668493876549 1030D-6*FCT(X)                                 DL12 500
      X=.1300605499330634 8D2                                           DL12 510
      Y=Y+.8365055856819799D-5*FCT(X)                                   DL12 520
      X=.9621316842568 67D1                                             DL12 530
      Y=Y+.2032315926629 9939D-3*FCT(X)                                 DL12 540
      X=.6844525453115177 3D1                                           DL12 550
      Y=Y+.2663973541865 3159D-2*FCT(X)                                 DL12 560
      X=.4599227639418348 5D1                                           DL12 570
      Y=Y+.2010238115463 4097D-1*FCT(X)                                 DL12 580
      X=.2833751337743507 2D1                                           DL12 590
      Y=Y+.9044922221168 093D-1*FCT(X)                                  DL12 600
      X=.1512610269776418 8D1                                           DL12 610
      Y=Y+.2440820113198 7756D0*FCT(X)                                  DL12 620
      X=.6117574845151306 7D0                                           DL12 630
      Y=Y+.3777592758731 3798D0*FCT(X)                                  DL12 640
      X=.1157221173580206 8D0                                           DL12 650
      Y=Y+.2647313710554 4319D0*FCT(X)                                  DL12 660
      RETURN                                                            DL12 670
      END                                                               DL12 680
```

```
C                                                                       DL16  10
C     ..................................................................DL16  20
C                                                                       DL16  30
C        SUBROUTINE DQL16                                               DL16  40
C                                                                       DL16  50
C        PURPOSE                                                        DL16  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X           DL16  70
C                                 FROM 0 TO INFINITY).                  DL16  80
C                                                                       DL16  90
C        USAGE                                                          DL16 100
C           CALL DQL16 (FCT,Y)                                          DL16 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DL16 120
C                                                                       DL16 130
C        DESCRIPTION OF PARAMETERS                                      DL16 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DL16 150
C                    SUBPROGRAM USED.                                   DL16 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DL16 170
C                                                                       DL16 180
C        REMARKS                                                        DL16 190
C           NONE                                                        DL16 200
C                                                                       DL16 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DL16 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DL16 230
C           MUST BE FURNISHED BY THE USER.                              DL16 240
C                                                                       DL16 250
C        METHOD                                                         DL16 260
C           EVALUATION IS DONE BY MEANS OF 16-POINT GAUSSIAN-LAGUERRE   DL16 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,               DL16 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 31.            DL16 290
C           FOR REFERENCE, SEE                                          DL16 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DL16 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DL16 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DL16 330
C           TR00.1100 (MARCH 1964), PP.24-25.                           DL16 340
C                                                                       DL16 350
C     ..................................................................DL16 360
C                                                                       DL16 370
      SUBROUTINE DQL16(FCT,Y)                                           DL16 380
C                                                                       DL16 390
C                                                                       DL16 400
      DOUBLE PRECISION X,Y,FCT                                          DL16 410
C                                                                       DL16 420
      X=.5170116033954331 8D2                                           DL16 430
      Y=.4161462370372855 2D-21*FCT(X)                                  DL16 440
      X=.4194045264768833 3D2                                           DL16 450
      Y=Y+.5050473700035512 8D-17*FCT(X)                                DL16 460
      X=.3458339870228662 6D2                                           DL16 470
      Y=Y+.6297967002517867 8D-14*FCT(X)                                DL16 480
      X=.2857872974288214 0D2                                           DL16 490
      Y=Y+.2127079033224103 0D-11*FCT(X)                                DL16 500
      X=.2351590569399190 9D2                                           DL16 510
      Y=Y+.2862350242973881 6D-9*FCT(X)                                 DL16 520
      X=.1918015685675313 5D2                                           DL16 530
      Y=Y+.1881024841079673 2D-7*FCT(X)                                 DL16 540
```

```
      X=.1544152736878161 7D2                                           DL16 550
      Y=Y+.6828319330871 1996D-6*FCT(X)                                 DL16 560
      X=.1221422336896615 9D2                                           DL16 570
      Y=Y+.1484458687398 1299D-4*FCT(X)                                 DL16 580
      X=.9438314336391939 D1                                            DL16 590
      Y=Y+.2042719153082 7846D-3*FCT(X)                                 DL16 600
      X=.7070338535048234 1D1                                           DL16 610
      Y=Y+.1849070943526 3109D-2*FCT(X)                                 DL16 620
      X=.5078018614549767 9D1                                           DL16 630
      Y=Y+.1129990C0803394530-1*FCT(X)                                  DL16 640
      X=.3437C866338932066D1                                            DL16 650
      Y=Y+.4732892869412521 9D-1*FCT(X)                                 DL16 660
      X=.2129283645098380 6D1                                           DL16 670
      Y=Y+.1362969342963775 4D0*FCT(X)                                  DL16 680
      X=.1141057774831226 9D1                                           DL16 690
      Y=Y+.2657957776442141 5D0*FCT(X)                                  DL16 700
      X=.4626963289150808 3D0                                           DL16 710
      Y=Y+.3310578549508841 7D0*FCT(X)                                  DL16 720
      X=.8764941047892784D-1                                            DL16 730
      Y=Y+.2061517149578009 9D0*FCT(X)                                  DL16 740
      RETURN                                                            DL16 750
      END                                                               DL16 760
```

```
C                                                                       DL24  10
C     ..................................................................DL24  20
C                                                                       DL24  30
C        SUBROUTINE DQL24                                               DL24  40
C                                                                       DL24  50
C        PURPOSE                                                        DL24  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X           DL24  70
C                                 FROM 0 TO INFINITY).                  DL24  80
C                                                                       DL24  90
C        USAGE                                                          DL24 100
C           CALL DQL24 (FCT,Y)                                          DL24 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DL24 120
C                                                                       DL24 130
C        DESCRIPTION OF PARAMETERS                                      DL24 140
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DL24 150
C                     SUBPROGRAM USED.                                  DL24 160
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DL24 170
C                                                                       DL24 180
C        REMARKS                                                        DL24 190
C           NONE                                                        DL24 200
C                                                                       DL24 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DL24 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DL24 230
C           MUST BE FURNISHED BY THE USER.                              DL24 240
C                                                                       DL24 250
C        METHOD                                                         DL24 260
C           EVALUATION IS DONE BY MEANS OF 24-POINT GAUSSIAN-LAGUERRE   DL24 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,               DL24 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 47.            DL24 290
C           FOR REFERENCE, SEE                                          DL24 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DL24 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DL24 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DL24 330
C           TR00.1100 (MARCH 1964), PP.24-25.                           DL24 340
C                                                                       DL24 350
C     ..................................................................DL24 360
C                                                                       DL24 370
      SUBROUTINE DQL24(FCT,Y)                                           DL24 380
C                                                                       DL24 390
C                                                                       DL24 400
      DOUBLE PRECISION X,Y,FCT                                          DL24 410
C                                                                       DL24 420
      X=.8149827923394889D2                                             DL24 430
      Y=.5575345788328356 8D-34*FCT(X)                                  DL24 440
      X=.6996224003510503 0D2                                           DL24 450
      Y=Y+.4088301593680657 8D-29*FCT(X)                                DL24 460
      X=.6105853144721876 2D2                                           DL24 470
      Y=Y+.2451818845878402 7D-25*FCT(X)                                DL24 480
      X=.5360857454469507 0D2                                           DL24 490
      Y=Y+.3605765864552959 0D-22*FCT(X)                                DL24 500
      X=.4715310644515632 3D2                                           DL24 510
      Y=Y+.2010517464555503 5D-19*FCT(X)                                DL24 520
      X=.4145172048487076 7D2                                           DL24 530
      Y=Y+.5350188813010037 6D-17*FCT(X)                                DL24 540
      X=.3635840580165162 2D2                                           DL24 550
      Y=Y+.7819800382459448 D-15*FCT(X)                                 DL24 560
      X=.3177604135237472 3D2                                           DL24 570
      Y=Y+.6894181052958857 D-13*FCT(X)                                 DL24 580
      X=.2763593717433271 7D2                                           DL24 590
      Y=Y+.3917736515084514 D-11*FCT(X)                                 DL24 600
      X=.2388732984816973 3D2                                           DL24 610
      Y=Y+.1507008226292584 9D-9*FCT(X)                                 DL24 620
      X=.2049146008261642 5D2                                           DL24 630
      Y=Y+.4072858987549999 7D-8*FCT(X)                                 DL24 640
      X=.1741799264650897 9D2                                           DL24 650
      Y=Y+.7960812959133630 D-7*FCT(X)                                  DL24 660
      X=.1464273228959667 4D2                                           DL24 670
      Y=Y+.1151315812737279 9D-5*FCT(X)                                 DL24 680
      X=.1214610271172976 6D2                                           DL24 690
      Y=Y+.1254472197799333 3D-4*FCT(X)                                 DL24 700
      X=.9912098015077706 D1                                            DL24 710
      Y=Y+.1044612146592718 D-3*FCT(X)                                  DL24 720
      X=.7927539247172152 D1                                            DL24 730
      Y=Y+.6721625640935478 9D-3*FCT(X)                                 DL24 740
      X=.6181535118736765 4D1                                           DL24 750
      Y=Y+.3369349058478303 6D-2*FCT(X)                                 DL24 760
      X=.4665083703467170 8D1                                           DL24 770
      Y=Y+.1322601940512015 7D-1*FCT(X)                                 DL24 780
      X=.3370774262089977 D1                                            DL24 790
      Y=Y+.4073247815140864 6D-1*FCT(X)                                 DL24 800
      X=.2292562058632190 3D1                                           DL24 810
      Y=Y+.9816627262991889 D-1*FCT(X)                                  DL24 820
      X=.1425597590803613 1D1                                           DL24 830
      Y=Y+.1833226889777780 2D0*FCT(X)                                  DL24 840
      X=.7660969055459366 D0                                            DL24 850
      Y=Y+.2588067072786980 D0*FCT(X)                                   DL24 860
      X=.3112391461984837 3D0                                           DL24 870
      Y=Y+.2587741075174239 0D0*FCT(X)                                  DL24 880
      X=.5901985218150797 7D-1                                          DL24 890
      Y=Y+.1428119733347818 5D0*FCT(X)                                  DL24 900
      RETURN                                                            DL24 910
      END                                                               DL24 920
```

```
C                                                                       DL32  10
C     ..................................................................DL32  20
C                                                                       DL32  30
C        SUBROUTINE DQL32                                               DL32  40
C                                                                       DL32  50
C        PURPOSE                                                        DL32  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X), SUMMED OVER X           DL32  70
C                                 FROM 0 TO INFINITY).                  DL32  80
C                                                                       DL32  90
C        USAGE                                                          DL32 100
C           CALL DQL32 (FCT,Y)                                          DL32 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DL32 120
C                                                                       DL32 130
C        DESCRIPTION OF PARAMETERS                                      DL32 140
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DL32 150
C                     SUBPROGRAM USED.                                  DL32 160
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DL32 170
C                                                                       DL32 180
C        REMARKS                                                        DL32 190
C           NONE                                                        DL32 200
C                                                                       DL32 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DL32 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DL32 230
C           MUST BE FURNISHED BY THE USER.                              DL32 240
C                                                                       DL32 250
C        METHOD                                                         DL32 260
C           EVALUATION IS DONE BY MEANS OF 32-POINT GAUSSIAN-LAGUERRE   DL32 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,               DL32 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 63.            DL32 290
C           FOR REFERENCE, SEE                                          DL32 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DL32 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DL32 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DL32 330
C           TR00.1100 (MARCH 1964), PP.24-25.                           DL32 340
C                                                                       DL32 350
C     ..................................................................DL32 360
C                                                                       DL32 370
      SUBROUTINE DQL32(FCT,Y)                                           DL32 380
C                                                                       DL32 390
C                                                                       DL32 400
      DOUBLE PRECISION X,Y,FCT                                          DL32 410
C                                                                       DL32 420
      X=.1117513980979377 0D3                                           DL32 430
      Y=.4510536193898974 2D-47*FCT(X)                                  DL32 440
      X=.9882954286828397 D2                                            DL32 450
      Y=Y+.1338616942106256 3D-41*FCT(X)                                DL32 460
      X=.8873534041789240 D2                                            DL32 470
      Y=Y+.2671511219240137 0D-37*FCT(X)                                DL32 480
      X=.8018744697791352 D2                                            DL32 490
      Y=Y+.1192248760098222 4D-33*FCT(X)                                DL32 500
      X=.7268762809066271 D2                                            DL32 510
      Y=Y+.1913375494454224 3D-30*FCT(X)                                DL32 520
      X=.6597537728793505 3D2                                           DL32 530
      Y=Y+.1418560545463036 9D-27*FCT(X)                                DL32 540
      X=.5989250916213401 8D2                                           DL32 550
      Y=Y+.5661294130397359 4D-25*FCT(X)                                DL32 560
      X=.5433372133396907 D2                                            DL32 570
      Y=Y+.1346982586637395 2D-22*FCT(X)                                DL32 580
      X=.4922439498730863 9D2                                           DL32 590
      Y=Y+.2054429673788045 4D-20*FCT(X)                                DL32 600
      X=.4450920799575493 8D2                                           DL32 610
      Y=Y+.2119792290163618 6D-18*FCT(X)                                DL32 620
      X=.4014571977153944 2D2                                           DL32 630
      Y=Y+.1542133833393823 4D-16*FCT(X)                                DL32 640
      X=.3610049480575197 4D2                                           DL32 650
      Y=Y+.8171823446342071 9D-15*FCT(X)                                DL32 660
      X=.3234662915396473 7D2                                           DL32 670
      Y=Y+.3237801657729266 5D-13*FCT(X)                                DL32 680
      X=.2886210181632347 5D2                                           DL32 690
      Y=Y+.9799379288727094 D-12*FCT(X)                                 DL32 700
      X=.2562863602245924 8D2                                           DL32 710
      Y=Y+.2305899491891336 1D-10*FCT(X)                                DL32 720
      X=.2263088901319677 4D2                                           DL32 730
      Y=Y+.4281382971040928 9D-9*FCT(X)                                 DL32 740
      X=.1985586094033605 5D2                                           DL32 750
      Y=Y+.6350602226625806 7D-8*FCT(X)                                 DL32 760
      X=.1729245433671531 5D2                                           DL32 770
      Y=Y+.7604567879120781 D-7*FCT(X)                                  DL32 780
      X=.1493113975552255 7D2                                           DL32 790
      Y=Y+.7416404578667552 D-6*FCT(X)                                  DL32 800
      X=.1276369798674272 5D2                                           DL32 810
      Y=Y+.5934541612868632 9D-5*FCT(X)                                 DL32 820
      X=.1078301863253997 2D2                                           DL32 830
      Y=Y+.3920341967987947 2D-4*FCT(X)                                 DL32 840
      X=.8982940924212596 D1                                            DL32 850
      Y=Y+.2148649188013641 9D-3*FCT(X)                                 DL32 860
      X=.7358126733186241 D1                                            DL32 870
      Y=Y+.9808033C6614951 D-3*FCT(X)                                   DL32 880
      X=.5903958504174243 9D1                                           DL32 890
      Y=Y+.3738816294611524 8D-2*FCT(X)                                 DL32 900
      X=.4616456769749767 4D1                                           DL32 910
      Y=Y+.1191821483483855 7D-1*FCT(X)                                 DL32 920
      X=.3492213273021994 5D1                                           DL32 930
      Y=Y+.3176091250917507 0D-1*FCT(X)                                 DL32 940
      X=.2528336706425794 9D1                                           DL32 950
      Y=Y+.7057862386571744 2D-1*FCT(X)                                 DL32 960
      X=.1722408776444645 4D1                                           DL32 970
      Y=Y+.1299837862860717 6D0*FCT(X)                                  DL32 980
      X=.1072448753817817 6D1                                           DL32 990
      Y=Y+.1959033359728810 4D0*FCT(X)                                  DL321000
      X=.5768846293018864 3D0                                           DL321010
      Y=Y+.2352132296698480 1D0*FCT(X)                                  DL321020
      X=.2345261095196185 4D0                                           DL321030
      Y=Y+.2104431079388132 3D0*FCT(X)                                  DL321040
      X=.4448936583326701 8D-1                                          DL321050
      Y=Y+.1092183419523849 7D0*FCT(X)                                  DL321060
      RETURN                                                            DL321070
      END                                                               DL321080
```

Subroutines QH2, QH3, . . . , QH10; DQH8, DQH16, DQH24, DQH32, DQH48, DQH64

These subroutines perform the integration of a given function by Gaussian-Hermite quadrature formulas.

To compute:

$$y = \int_{-\infty}^{+\infty} e^{-x^2} f(x)\, dx$$

Gaussian-Hermite quadrature formulas with n = 2, 3, . . . , 10 (or n = 8, 16, 24, 32, 48, 64) points are used to get the approximations:

$$y_n = \sum_{x=1}^{n} A_k^{(n)} f(x_k^{(n)}) \qquad (n = 2, 3, \ldots) \qquad (1)$$

which are exact whenever f(x) is a polynomial up to the degree 2n-1. Note that the nodes $x_k^{(n)}$ are the roots of the Hermite polynomials $H_n(x)$ of degree n. As $A_k^{(n)}$ and $x_k^{(n)}$ are symmetric with respect to x=0, therefore:

$$A_k^{(n)} = A_{n-k+1}^{(n)}, \quad x_k^{(n)} = -x_{n-k+1}^{(n)}$$

Note: In case of an even function $f(x) = \phi(x^2)$, the transformation $t=x^2$ transforms the given integral into:

$$y = \int_0^{\infty} \frac{e^{-t}\varphi(t)}{\sqrt{t}}\, dt$$

which is a form suitable to subroutines QA2, QA3, . . . , QA10 and DQA4, . . . , DQA32. Using these subroutines saves approximately half of the computation time.

For reference see:

(1) H.E. Salzer/R. Zucker/R. Capuano, "Table of the Zeros and Weight Factors of the First Twenty Hermite Polynomials", J. Res. Nat. Bur. Standards, vol. 48 (1952), pp. 111-116.

(2) V.I. Krylov, Approximate Calculation of Integrals. Macmillan, New York/London, 1962, pp. 129-130 and 343-346.


```
C                                                                       QH2   10
C     ..................................................................QH2   20
C                                                                       QH2   30
C        SUBROUTINE QH2                                                 QH2   40
C                                                                       QH2   50
C        PURPOSE                                                        QH2   60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    QH2   70
C                              -INFINITY TO +INFINITY).                 QH2   80
C                                                                       QH2   90
C        USAGE                                                          QH2  100
C           CALL QH2 (FCT,Y)                                            QH2  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QH2  120
C                                                                       QH2  130
C        DESCRIPTION OF PARAMETERS                                      QH2  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QH2  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QH2  160
C                                                                       QH2  170
C        REMARKS                                                        QH2  180
C           NONE                                                        QH2  190
C                                                                       QH2  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QH2  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QH2  220
C           BY THE USER.                                                QH2  230
C                                                                       QH2  240
C        METHOD                                                         QH2  250
C           EVALUATION IS DONE BY MEANS OF 2-POINT GAUSSIAN-HERMITE     QH2  260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QH2  270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 3.                      QH2  280
C           FOR REFERENCE, SEE                                          QH2  290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QH2  300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.   QH2  310
C                                                                       QH2  320
C     ..................................................................QH2  330
C                                                                       QH2  340
      SUBROUTINE QH2(FCT,Y)                                             QH2  350
C                                                                       QH2  360
C                                                                       QH2  370
      X=.7071068                                                        QH2  380
      Z=-X                                                              QH2  390
      Y=.8862269*(FCT(X)+FCT(Z))                                        QH2  400
      RETURN                                                            QH2  410
      END                                                               QH2  420
```

```
C                                                                       QH3   10
C     ..................................................................QH3   20
C                                                                       QH3   30
C        SUBROUTINE QH3                                                 QH3   40
C                                                                       QH3   50
C        PURPOSE                                                        QH3   60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    QH3   70
C                              -INFINITY TO +INFINITY).                 QH3   80
C                                                                       QH3   90
C        USAGE                                                          QH3  100
C           CALL QH3 (FCT,Y)                                            QH3  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QH3  120
C                                                                       QH3  130
C        DESCRIPTION OF PARAMETERS                                      QH3  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QH3  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QH3  160
C                                                                       QH3  170
C        REMARKS                                                        QH3  180
C           NONE                                                        QH3  190
C                                                                       QH3  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QH3  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QH3  220
C           BY THE USER.                                                QH3  230
C                                                                       QH3  240
C        METHOD                                                         QH3  250
C           EVALUATION IS DONE BY MEANS OF 3-POINT GAUSSIAN-HERMITE     QH3  260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QH3  270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 5.                      QH3  280
C           FOR REFERENCE, SEE                                          QH3  290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QH3  300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.   QH3  310
C                                                                       QH3  320
C     ..................................................................QH3  330
C                                                                       QH3  340
      SUBROUTINE QH3(FCT,Y)                                             QH3  350
C                                                                       QH3  360
C                                                                       QH3  370
      X=1.224745                                                        QH3  380
      Z=-X                                                              QH3  390
      Y=.2954090*(FCT(X)+FCT(Z))                                        QH3  400
      X=0.                                                              QH3  410
      Y=Y+1.181636*FCT(X)                                               QH3  420
      RETURN                                                            QH3  430
      END                                                               QH3  440
```

```
C
C     ..................................................................
C
C        SUBROUTINE QH4
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM
C                                 -INFINITY TO +INFINITY).
C
C        USAGE
C           CALL QH4 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y      - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 4-POINT GAUSSIAN-HERMITE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 7.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.
C
C     ..................................................................
C
      SUBROUTINE QH4(FCT,Y)
C
C
      X=1.650680
      Z=-X
      Y=.08131284*(FCT(X)+FCT(Z))
      X=.5246476
      Z=-X
      Y=Y+.8049141*(FCT(X)+FCT(Z))
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QH5
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM
C                                 -INFINITY TO +INFINITY).
C
C        USAGE
C           CALL QH5 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y      - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 5-POINT GAUSSIAN-HERMITE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 9.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.
C
C     ..................................................................
C
      SUBROUTINE QH5(FCT,Y)
C
C
      X=2.020183
      Z=-X
      Y=.01995324*(FCT(X)+FCT(Z))
      X=.9585725
      Z=-X
      Y=Y+.3936193*(FCT(X)+FCT(Z))
      X=0.
      Y=Y+.9453087*FCT(X)
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QH6
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM
C                                 -INFINITY TO +INFINITY).
C
C        USAGE
C           CALL QH6 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y      - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 6-POINT GAUSSIAN-HERMITE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 11.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.
C
C     ..................................................................
C
      SUBROUTINE QH6(FCT,Y)
C
C
      X=2.350605
      Z=-X
      Y=.004530010*(FCT(X)+FCT(Z))
      X=1.335849
      Z=-X
      Y=Y+.1570673*(FCT(X)+FCT(Z))
      X=.4360774
      Z=-X
      Y=Y+.7246296*(FCT(X)+FCT(Z))
      RETURN
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE QH7
C
C        PURPOSE
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM
C                                 -INFINITY TO +INFINITY).
C
C        USAGE
C           CALL QH7 (FCT,Y)
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C        DESCRIPTION OF PARAMETERS
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.
C           Y      - THE RESULTING INTEGRAL VALUE.
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED
C           BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF 7-POINT GAUSSIAN-HERMITE
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 13.
C           FOR REFERENCE, SEE
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.
C
C     ..................................................................
C
      SUBROUTINE QH7(FCT,Y)
C
C
      X=2.651961
      Z=-X
      Y=.0009717812*(FCT(X)+FCT(Z))
      X=1.673552
      Z=-X
      Y=Y+.05451558*(FCT(X)+FCT(Z))
      X=.8162879
      Z=-X
      Y=Y+.4256073*(FCT(X)+FCT(Z))
      X=0.
      Y=Y+.8102646*FCT(X)
      RETURN
      END
```

```
C                                                                       QH8  10
C     ...................................................................QH8  20
C                                                                       QH8  30
C        SUBROUTINE QH8                                                 QH8  40
C                                                                       QH8  50
C        PURPOSE                                                        QH8  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    QH8  70
C                              -INFINITY TO +INFINITY).                 QH8  80
C                                                                       QH8  90
C        USAGE                                                          QH8 100
C           CALL QH8 (FCT,Y)                                            QH8 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QH8 120
C                                                                       QH8 130
C        DESCRIPTION OF PARAMETERS                                      QH8 140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QH8 150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QH8 160
C                                                                       QH8 170
C        REMARKS                                                        QH8 180
C           NONE                                                        QH8 190
C                                                                       QH8 200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QH8 210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QH8 220
C           BY THE USER.                                                QH8 230
C                                                                       QH8 240
C        METHOD                                                         QH8 250
C           EVALUATION IS DONE BY MEANS OF 8-POINT GAUSSIAN-HERMITE     QH8 260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QH8 270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 15.                     QH8 280
C           FOR REFERENCE, SEE                                          QH8 290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QH8 300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.   QH8 310
C                                                                       QH8 320
C     ...................................................................QH8 330
C                                                                       QH8 340
      SUBROUTINE QH8(FCT,Y)                                             QH8 350
C                                                                       QH8 360
C                                                                       QH8 370
      X=2.930637                                                        QH8 380
      Z=-X                                                              QH8 390
      Y=.0001996041*(FCT(X)+FCT(Z))                                     QH8 400
      X=1.981657                                                        QH8 410
      Z=-X                                                              QH8 420
      Y=Y+.01707798*(FCT(X)+FCT(Z))                                     QH8 430
      X=1.157194                                                        QH8 440
      Z=-X                                                              QH8 450
      Y=Y+.2078023*(FCT(X)+FCT(Z))                                      QH8 460
      X=.3811870                                                        QH8 470
      Z=-X                                                              QH8 480
      Y=Y+.6611470*(FCT(X)+FCT(Z))                                      QH8 490
      RETURN                                                            QH8 500
      END                                                               QH8 510
```

```
C                                                                       QH9  10
C     ...................................................................QH9  20
C                                                                       QH9  30
C        SUBROUTINE QH9                                                 QH9  40
C                                                                       QH9  50
C        PURPOSE                                                        QH9  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    QH9  70
C                              -INFINITY TO +INFINITY).                 QH9  80
C                                                                       QH9  90
C        USAGE                                                          QH9 100
C           CALL QH9 (FCT,Y)                                            QH9 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QH9 120
C                                                                       QH9 130
C        DESCRIPTION OF PARAMETERS                                      QH9 140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QH9 150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QH9 160
C                                                                       QH9 170
C        REMARKS                                                        QH9 180
C           NONE                                                        QH9 190
C                                                                       QH9 200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QH9 210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QH9 220
C           BY THE USER.                                                QH9 230
C                                                                       QH9 240
C        METHOD                                                         QH9 250
C           EVALUATION IS DONE BY MEANS OF 9-POINT GAUSSIAN-HERMITE     QH9 260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QH9 270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 17.                     QH9 280
C           FOR REFERENCE, SEE                                          QH9 290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QH9 300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.   QH9 310
C                                                                       QH9 320
C     ...................................................................QH9 330
C                                                                       QH9 340
      SUBROUTINE QH9(FCT,Y)                                             QH9 350
C                                                                       QH9 360
C                                                                       QH9 370
      X=3.190993                                                        QH9 380
      Z=-X                                                              QH9 390
      Y=.3960698E-4*(FCT(X)+FCT(Z))                                     QH9 400
      X=2.266581                                                        QH9 410
      Z=-X                                                              QH9 420
      Y=Y+.004943624*(FCT(X)+FCT(Z))                                    QH9 430
      X=1.468553                                                        QH9 440
      Z=-X                                                              QH9 450
      Y=Y+.08847453*(FCT(X)+FCT(Z))                                     QH9 460
      X=.7235510                                                        QH9 470
      Z=-X                                                              QH9 480
      Y=Y+.4326516*(FCT(X)+FCT(Z))                                      QH9 490
      X=0.                                                              QH9 500
      Y=Y+.7202352*FCT(X)                                               QH9 510
      RETURN                                                            QH9 520
      END                                                               QH9 530
```

```
C                                                                       DQH8  10
C     ...................................................................DQH8  20
C                                                                       DQH8  30
C        SUBROUTINE DQH8                                                DQH8  40
C                                                                       DQH8  50
C        PURPOSE                                                        DQH8  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    DQH8  70
C                              -INFINITY TO +INFINITY).                 DQH8  80
C                                                                       DQH8  90
C        USAGE                                                          DQH8 100
C           CALL DQH8 (FCT,Y)                                           DQH8 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DQH8 120
C                                                                       DQH8 130
C        DESCRIPTION OF PARAMETERS                                      DQH8 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DQH8 150
C                    SUBPROGRAM USED.                                   DQH8 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DQH8 170
C                                                                       DQH8 180
C        REMARKS                                                        DQH8 190
C           NONE                                                        DQH8 200
C                                                                       DQH8 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQH8 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DQH8 230
C           MUST BE FURNISHED BY THE USER.                              DQH8 240
C                                                                       DQH8 250
C        METHOD                                                         DQH8 260
C           EVALUATION IS DONE BY MEANS OF 8-POINT GAUSSIAN-HERMITE     DQH8 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       DQH8 280
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 15.                     DQH8 290
C           FOR REFERENCE, SEE                                          DQH8 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DQH8 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DQH8 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DQH8 330
C           TR00.1100 (MARCH 1964), PP.213-214.                         DQH8 340
C                                                                       DQH8 350
C     ...................................................................DQH8 360
C                                                                       DQH8 370
      SUBROUTINE DQH8(FCT,Y)                                            DQH8 380
C                                                                       DQH8 390
C                                                                       DQH8 400
      DOUBLE PRECISION X,Y,Z,FCT                                        DQH8 410
C                                                                       DQH8 420
      X=.2930637420257244001                                            DQH8 430
      Z=-X                                                              DQH8 440
      Y=.1996040722113676 2D-3*(FCT(X)+FCT(Z))                          DQH8 450
      X=.1981656756695842 9D1                                           DQH8 460
      Z=-X                                                              DQH8 470
      Y=Y+.1707798300741347 5D-1*(FCT(X)+FCT(Z))                        DQH8 480
      X=.1157193712446780 2D1                                           DQH8 490
      Z=-X                                                              DQH8 500
      Y=Y+.2078023258148918 8D0*(FCT(X)+FCT(Z))                         DQH8 510
      X=.3811869902073221 2D0                                           DQH8 520
      Z=-X                                                              DQH8 530
      Y=Y+.6611470125582412 9D0*(FCT(X)+FCT(Z))                         DQH8 540
      RETURN                                                            DQH8 550
      END                                                               DQH8 560
```

```
C                                                                       QH10  10
C     ...................................................................QH10  20
C                                                                       QH10  30
C        SUBROUTINE QH10                                                QH10  40
C                                                                       QH10  50
C        PURPOSE                                                        QH10  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    QH10  70
C                              -INFINITY TO +INFINITY).                 QH10  80
C                                                                       QH10  90
C        USAGE                                                          QH10 100
C           CALL QH10(FCT,Y)                                            QH10 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QH10 120
C                                                                       QH10 130
C        DESCRIPTION OF PARAMETERS                                      QH10 140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QH10 150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QH10 160
C                                                                       QH10 170
C        REMARKS                                                        QH10 180
C           NONE                                                        QH10 190
C                                                                       QH10 200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QH10 210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QH10 220
C           BY THE USER.                                                QH10 230
C                                                                       QH10 240
C        METHOD                                                         QH10 250
C           EVALUATION IS DONE BY MEANS OF 10-POINT GAUSSIAN-HERMITE    QH10 260
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       QH10 270
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 19.                     QH10 280
C           FOR REFERENCE, SEE                                          QH10 290
C           V.I.KRYLOV, APPROXIMATE CALCULATION OF INTEGRALS,           QH10 300
C           MACMILLAN, NEW YORK/LONDON, 1962, PP.129-130 AND 343-346.   QH10 310
C                                                                       QH10 320
C     ...................................................................QH10 330
C                                                                       QH10 340
      SUBROUTINE QH10(FCT,Y)                                            QH10 350
C                                                                       QH10 360
C                                                                       QH10 370
      X=3.436159                                                        QH10 380
      Z=-X                                                              QH10 390
      Y=.7640433E-5*(FCT(X)+FCT(Z))                                     QH10 400
      X=2.532732                                                        QH10 410
      Z=-X                                                              QH10 420
      Y=Y+.001343646*(FCT(X)+FCT(Z))                                    QH10 430
      X=1.756684                                                        QH10 440
      Z=-X                                                              QH10 450
      Y=Y+.03387439*(FCT(X)+FCT(Z))                                     QH10 460
      X=1.036611                                                        QH10 470
      Z=-X                                                              QH10 480
      Y=Y+.2401386*(FCT(X)+FCT(Z))                                      QH10 490
      X=.3429013                                                        QH10 500
      Z=-X                                                              QH10 510
      Y=Y+.6108626*(FCT(X)+FCT(Z))                                      QH10 520
      RETURN                                                            QH10 530
      END                                                               QH10 540
```

```
C                                                                       DH16  10
C     ..................................................................DH16  20
C                                                                       DH16  30
C        SUBROUTINE DQH16                                               DH16  40
C                                                                       DH16  50
C        PURPOSE                                                        DH16  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    DH16  70
C                              -INFINITY TO +INFINITY).                 DH16  80
C                                                                       DH16  90
C        USAGE                                                          DH16 100
C           CALL DQH16 (FCT,Y)                                          DH16 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DH16 120
C                                                                       DH16 130
C        DESCRIPTION OF PARAMETERS                                      DH16 140
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DH16 150
C                     SUBPROGRAM USED.                                  DH16 160
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DH16 170
C                                                                       DH16 180
C        REMARKS                                                        DH16 190
C           NONE                                                        DH16 200
C                                                                       DH16 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DH16 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DH16 230
C           MUST BE FURNISHED BY THE USER.                              DH16 240
C                                                                       DH16 250
C        METHOD                                                         DH16 260
C           EVALUATION IS DONE BY MEANS OF 16-POINT GAUSSIAN-HERMITE    DH16 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       DH16 280
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 31.                     DH16 290
C           FOR REFERENCE, SEE                                          DH16 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DH16 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DH16 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DH16 330
C           TR00.1100 (MARCH 1964), PP.213-214.                         DH16 340
C                                                                       DH16 350
C     ..................................................................DH16 360
C                                                                       DH16 370
      SUBROUTINE DQH16(FCT,Y)                                           DH16 380
C                                                                       DH16 390
C                                                                       DH16 400
      DOUBLE PRECISION X,Y,Z,FCT                                        DH16 410
C                                                                       DH16 420
      X=.468873893930581840D                                            DH16 430
      Z=-X                                                              DH16 440
      Y=.265480747401118 22D-9*(FCT(X)+FCT(Z))                          DH16 450
      X=.386944790486012 27D                                            DH16 460
      Z=-X                                                              DH16 470
      Y=Y+.232098084486521 07D-6*(FCT(X)+FCT(Z))                        DH16 480
      X=.317699916197995 6D                                             DH16 490
      Z=-X                                                              DH16 500
      Y=Y+.271186009253788 15D-4*(FCT(X)+FCT(Z))                        DH16 510
      X=.254620215784748 14D                                            DH16 520
      Z=-X                                                              DH16 530
      Y=Y+.932284008624180 5D-3*(FCT(X)+FCT(Z))                         DH16 540
      X=.195178799091625 40D                                            DH16 550
      Z=-X                                                              DH16 560
      Y=Y+.128803115355099 74D-1*(FCT(X)+FCT(Z))                        DH16 570
      X=.138025853919888 08D                                            DH16 580
      Z=-X                                                              DH16 590
      Y=Y+.838100413989858 3D-1*(FCT(X)+FCT(Z))                         DH16 600
      X=.822951449144655 9D0                                            DH16 610
      Z=-X                                                              DH16 620
      Y=Y+.280647458528533 68D0*(FCT(X)+FCT(Z))                         DH16 630
      X=.273481046138152 45D0                                           DH16 640
      Z=-X                                                              DH16 650
      Y=Y+.507929479016613 74D0*(FCT(X)+FCT(Z))                         DH16 660
      RETURN                                                            DH16 670
      END                                                               DH16 680
```

```
C                                                                       DH24  10
C     ..................................................................DH24  20
C                                                                       DH24  30
C        SUBROUTINE DQH24                                               DH24  40
C                                                                       DH24  50
C        PURPOSE                                                        DH24  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    DH24  70
C                              -INFINITY TO +INFINITY).                 DH24  80
C                                                                       DH24  90
C        USAGE                                                          DH24 100
C           CALL DQH24 (FCT,Y)                                          DH24 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DH24 120
C                                                                       DH24 130
C        DESCRIPTION OF PARAMETERS                                      DH24 140
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DH24 150
C                     SUBPROGRAM USED.                                  DH24 160
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DH24 170
C                                                                       DH24 180
C        REMARKS                                                        DH24 190
C           NONE                                                        DH24 200
C                                                                       DH24 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DH24 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DH24 230
C           MUST BE FURNISHED BY THE USER.                              DH24 240
C                                                                       DH24 250
C        METHOD                                                         DH24 260
C           EVALUATION IS DONE BY MEANS OF 24-POINT GAUSSIAN-HERMITE    DH24 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       DH24 280
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 47.                     DH24 290
C           FOR REFERENCE, SEE                                          DH24 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DH24 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DH24 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DH24 330
C           TR00.1100 (MARCH 1964), PP.213-214.                         DH24 340
C                                                                       DH24 350
C     ..................................................................DH24 360
C                                                                       DH24 370
      SUBROUTINE DQH24(FCT,Y)                                           DH24 380
C                                                                       DH24 390
C                                                                       DH24 400
      DOUBLE PRECISION X,Y,Z,FCT                                        DH24 410
C                                                                       DH24 420
      X=.601592556142573 97D1                                           DH24 430
      Z=-X                                                              DH24 440
      Y=.166436849648910 89D-15*(FCT(X)+FCT(Z))                         DH24 450
      X=.525938292766804 44D1                                           DH24 460
      Z=-X                                                              DH24 470
      Y=Y+.658462024307817 01D-12*(FCT(X)+FCT(Z))                       DH24 480
      X=.462566275642378 73D1                                           DH24 490
      Z=-X                                                              DH24 500
      Y=Y+.304625426998756 39D-9*(FCT(X)+FCT(Z))                        DH24 510
      X=.405366440244814 95D1                                           DH24 520
      Z=-X                                                              DH24 530
      Y=Y+.401897117496142 97D-7*(FCT(X)+FCT(Z))                        DH24 540
      X=.352000681303452 47D1                                           DH24 550
      Z=-X                                                              DH24 560
      Y=Y+.215824570490233 36D-5*(FCT(X)+FCT(Z))                        DH24 570
      X=.301254613756556 48D1                                           DH24 580
      Z=-X                                                              DH24 590
      Y=Y+.568869163640437 98D-4*(FCT(X)+FCT(Z))                        DH24 600
      X=.252388101701142 7D1                                            DH24 610
      Z=-X                                                              DH24 620
      Y=Y+.823692482688417 5D-3*(FCT(X)+FCT(Z))                         DH24 630
      X=.204900357366169 8D1                                            DH24 640
      Z=-X                                                              DH24 650
      Y=Y+.704835581007267 10D-2*(FCT(X)+FCT(Z))                        DH24 660
      X=.158425001096169 41D1                                           DH24 670
      Z=-X                                                              DH24 680
      Y=Y+.374454705032307 46D-1*(FCT(X)+FCT(Z))                        DH24 690
      X=.112676081761124 51D1                                           DH24 700
      Z=-X                                                              DH24 710
      Y=Y+.127739621784559 16D0*(FCT(X)+FCT(Z))                         DH24 720
      X=.674171107037212 2D0                                            DH24 730
      Z=-X                                                              DH24 740
      Y=Y+.286179535346443 0 2D0*(FCT(X)+FCT(Z))                        DH24 750
      X=.224414547472515 5D0                                            DH24 760
      Z=-X                                                              DH24 770
      Y=Y+.426931163868699 2D0*(FCT(X)+FCT(Z))                          DH24 780
      RETURN                                                            DH24 790
      END                                                               DH24 800
```

```
C                                                                       DH32  10
C     ..................................................................DH32  20
C                                                                       DH32  30
C     SUBROUTINE DQH32                                                  DH32  40
C                                                                       DH32  50
C     PURPOSE                                                           DH32  60
C        TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM       DH32  70
C                          -INFINITY TO +INFINITY).                     DH32  80
C                                                                       DH32  90
C     USAGE                                                             DH32 100
C        CALL DQH32 (FCT,Y)                                             DH32 110
C        PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                   DH32 120
C                                                                       DH32 130
C     DESCRIPTION OF PARAMETERS                                         DH32 140
C        FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION    DH32 150
C                  SUBPROGRAM USED.                                     DH32 160
C        Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.       DH32 170
C                                                                       DH32 180
C     REMARKS                                                           DH32 190
C        NONE                                                           DH32 200
C                                                                       DH32 210
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     DH32 220
C        THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)       DH32 230
C        MUST BE FURNISHED BY THE USER.                                 DH32 240
C                                                                       DH32 250
C     METHOD                                                            DH32 260
C        EVALUATION IS DONE BY MEANS OF 32-POINT GAUSSIAN-HERMITE       DH32 270
C        QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER          DH32 280
C        FCT(X) IS A POLYNOMIAL UP TO DEGREE 63.                        DH32 290
C        FOR REFERENCE, SEE                                             DH32 300
C        SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF       DH32 310
C        CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED        DH32 320
C        GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT          DH32 330
C        TR00.1100 (MARCH 1964), PP.213-214.                            DH32 340
C                                                                       DH32 350
C     ..................................................................DH32 360
C                                                                       DH32 370
      SUBROUTINE DQH32(FCT,Y)                                           DH32 380
C                                                                       DH32 390
C                                                                       DH32 400
      DOUBLE PRECISION X,Y,Z,FCT                                        DH32 410
C                                                                       DH32 420
      X=.712581390983072760D1                                           DH32 430
      Z=-X                                                              DH32 440
      Y=.731067642738416 20D-22*(FCT(X)+FCT(Z))                         DH32 450
      X=.640949814926966040D1                                           DH32 460
      Z=-X                                                              DH32 470
      Y=Y+.923173653651829 2D-18*(FCT(X)+FCT(Z))                        DH32 480
      X=.581222594951591380D1                                           DH32 490
      Z=-X                                                              DH32 500
      Y=Y+.119734401709284870D-14*(FCT(X)+FCT(Z))                       DH32 510
      X=.527555098651588010D1                                           DH32 520
      Z=-X                                                              DH32 530
      Y=Y+.421501021132644760D-12*(FCT(X)+FCT(Z))                       DH32 540
      X=.477716450350259640D1                                           DH32 550
      Z=-X                                                              DH32 560
      Y=Y+.593329146339663860D-10*(FCT(X)+FCT(Z))                       DH32 570
      X=.430554795335119840D1                                           DH32 580
      Z=-X                                                              DH32 590
      Y=Y+.409883216477089660D-8*(FCT(X)+FCT(Z))                        DH32 600
      X=.385375548547144460D1                                           DH32 610
      Z=-X                                                              DH32 620
      Y=Y+.157416779254559400D-6*(FCT(X)+FCT(Z))                        DH32 630
      X=.341716749281857070D1                                           DH32 640
      Z=-X                                                              DH32 650
      Y=Y+.365058512956237610D-5*(FCT(X)+FCT(Z))                        DH32 660
      X=.299249082500237420D1                                           DH32 670
      Z=-X                                                              DH32 680
      Y=Y+.541658406181998260D-4*(FCT(X)+FCT(Z))                        DH32 690
      X=.257724953773231750D1                                           DH32 700
      Z=-X                                                              DH32 710
      Y=Y+.536268365527972050D-3*(FCT(X)+FCT(Z))                        DH32 720
      X=.216949918360611220D1                                           DH32 730
      Z=-X                                                              DH32 740
      Y=Y+.365489032665442810D-2*(FCT(X)+FCT(Z))                        DH32 750
      X=.176765410946320160D1                                           DH32 760
      Z=-X                                                              DH32 770
      Y=Y+.175534288315734300D-1*(FCT(X)+FCT(Z))                        DH32 780
      X=.137037641095287180D1                                           DH32 790
      Z=-X                                                              DH32 800
      Y=Y+.604581309559126140D-1*(FCT(X)+FCT(Z))                        DH32 810
      X=.976500463589682800                                             DH32 820
      Z=-X                                                              DH32 830
      Y=Y+.151269734076642480D0*(FCT(X)+FCT(Z))                         DH32 840
      X=.584978765435932450                                             DH32 850
      Z=-X                                                              DH32 860
      Y=Y+.277458142302529900D0*(FCT(X)+FCT(Z))                         DH32 870
      X=.194840741569399330                                             DH32 880
      Z=-X                                                              DH32 890
      Y=Y+.375238352592802390D0*(FCT(X)+FCT(Z))                         DH32 900
      RETURN                                                            DH32 910
      END                                                               DH32 920
```

```
C                                                                       DH48  10
C     ..................................................................DH48  20
C                                                                       DH48  30
C     SUBROUTINE DQH48                                                  DH48  40
C                                                                       DH48  50
C     PURPOSE                                                           DH48  60
C        TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM       DH48  70
C                          -INFINITY TO +INFINITY).                     DH48  80
C                                                                       DH48  90
C     USAGE                                                             DH48 100
C        CALL DQH48 (FCT,Y)                                             DH48 110
C        PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                   DH48 120
C                                                                       DH48 130
C     DESCRIPTION OF PARAMETERS                                         DH48 140
C        FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION    DH48 150
C                  SUBPROGRAM USED.                                     DH48 160
C        Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.       DH48 170
C                                                                       DH48 180
C     REMARKS                                                           DH48 190
C        NONE                                                           DH48 200
C                                                                       DH48 210
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     DH48 220
C        THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)       DH48 230
C        MUST BE FURNISHED BY THE USER.                                 DH48 240
C                                                                       DH48 250
C     METHOD                                                            DH48 260
C        EVALUATION IS DONE BY MEANS OF 48-POINT GAUSSIAN-HERMITE       DH48 270
C        QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER          DH48 280
C        FCT(X) IS A POLYNOMIAL UP TO DEGREE 95.                        DH48 290
C        FOR REFERENCE, SEE                                             DH48 300
C        SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF       DH48 310
C        CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED        DH48 320
C        GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT          DH48 330
C        TR00.1100 (MARCH 19641, PP.213-214.                            DH48 340
C                                                                       DH48 350
C     ..................................................................DH48 360
C                                                                       DH48 370
      SUBROUTINE DQH48(FCT,Y)                                           DH48 380
C                                                                       DH48 390
C                                                                       DH48 400
      DOUBLE PRECISION X,Y,Z,FCT                                        DH48 410
C                                                                       DH48 420
      X=.897531508193168701                                             DH48 430
      Z=-X                                                              DH48 440
      Y=.793555146077399 7D-35*(FCT(X)+FCT(Z))                          DH48 450
      X=.831075219070478401                                             DH48 460
      Z=-X                                                              DH48 470
      Y=Y+.598461269331387840-30*(FCT(X)+FCT(Z))                        DH48 480
      X=.775929551976577501                                             DH48 490
      Z=-X                                                              DH48 500
      Y=Y+.368503608015066990-26*(FCT(X)+FCT(Z))                        DH48 510
      X=.726604655416435001                                             DH48 520
      Z=-X                                                              DH48 530
      Y=Y+.556457746890228480-23*(FCT(X)+FCT(Z))                        DH48 540
      X=.681006457807414101                                             DH48 550
      Z=-X                                                              DH48 560
      Y=Y+.318838732350513840-20*(FCT(X)+FCT(Z))                        DH48 570
      X=.638056409618641060 1                                           DH48 580
      Z=-X                                                              DH48 590
      Y=Y+.873015960118667 7D-18*(FCT(X)+FCT(Z))                        DH48 600
      X=.597107222501354540 1                                           DH48 610
      Z=-X                                                              DH48 620
      Y=Y+.131515962265840 85D-15*(FCT(X)+FCT(Z))                       DH48 630
      X=.557731698122372860 1                                           DH48 640
      Z=-X                                                              DH48 650
      Y=Y+.119758986547917 94D-13*(FCT(X)+FCT(Z))                       DH48 660
      X=.519628771879236450 1                                           DH48 670
      Z=-X                                                              DH48 680
      Y=Y+.704693258145889 1D-12*(FCT(X)+FCT(Z))                        DH48 690
      X=.482575722813320950 1                                           DH48 700
      Z=-X                                                              DH48 710
      Y=Y+.281529653783816 91D-10*(FCT(X)+FCT(Z))                       DH48 720
      X=.446401454693445890 1                                           DH48 730
      Z=-X                                                              DH48 740
      Y=Y+.793046749516538 2D-9*(FCT(X)+FCT(Z))                         DH48 750
      X=.410970460356059020 1                                           DH48 760
      Z=-X                                                              DH48 770
      Y=Y+.162251413589576 98D-7*(FCT(X)+FCT(Z))                        DH48 780
      X=.376172649022835780 1                                           DH48 790
      Z=-X                                                              DH48 800
      Y=Y+.246865899366975 05D-6*(FCT(X)+FCT(Z))                        DH48 810
      X=.341916596936388460 1                                           DH48 820
      Z=-X                                                              DH48 830
      Y=Y+.284725869173484 81D-5*(FCT(X)+FCT(Z))                        DH48 840
      X=.308124898864510580 1                                           DH48 850
      Z=-X                                                              DH48 860
      Y=Y+.252859902774848 89D-4*(FCT(X)+FCT(Z))                        DH48 870
      X=.274730862482238320 1                                           DH48 880
      Z=-X                                                              DH48 890
      Y=Y+.175150431801172 83D-3*(FCT(X)+FCT(Z))                        DH48 900
      X=.241676090487321650 1                                           DH48 910
      Z=-X                                                              DH48 920
      Y=Y+.956392319819415 3D-3*(FCT(X)+FCT(Z))                         DH48 930
      X=.208908666094427640 1                                           DH48 940
      Z=-X                                                              DH48 950
      Y=Y+.415300491197755 25D-2*(FCT(X)+FCT(Z))                        DH48 960
      X=.176381757989530000 1                                           DH48 970
      Z=-X                                                              DH48 980
      Y=Y+.144449615749810 99D-1*(FCT(X)+FCT(Z))                        DH48 990
      X=.144052522013756520 1                                           DH481000
      Z=-X                                                              DH481010
      Y=Y+.404796769846038 49D-1*(FCT(X)+FCT(Z))                        DH481020
      X=.111881215240215660 1                                           DH481030
      Z=-X                                                              DH481040
      Y=Y+.918222970792851 8D-1*(FCT(X)+FCT(Z))                         DH481050
      X=.798304627778562200                                             DH481060
      Z=-X                                                              DH481070
      Y=Y+.169204471945641110D0*(FCT(X)+FCT(Z))                         DH481080
      X=.478646337594496100                                             DH481090
      Z=-X                                                              DH481100
      Y=Y+.253961542664759 10D0*(FCT(X)+FCT(Z))                         DH481110
      X=.159492935848862470                                             DH481120
      Z=-X                                                              DH481130
      Y=Y+.311001030377963 08D0*(FCT(X)+FCT(Z))                         DH481140
      RETURN                                                            DH481150
      END                                                               DH481160
```

```
C                                                                       DH64  10
C     ...................................................................DH64  20
C                                                                       DH64  30
C        SUBROUTINE DQH64                                               DH64  40
C                                                                       DH64  50
C        PURPOSE                                                        DH64  60
C           TO COMPUTE INTEGRAL(EXP(-X*X)*FCT(X), SUMMED OVER X FROM    DH64  70
C                              -INFINITY TO +INFINITY).                 DH64  80
C                                                                       DH64  90
C        USAGE                                                          DH64 100
C           CALL DQH64 (FCT,Y)                                          DH64 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DH64 120
C                                                                       DH64 130
C        DESCRIPTION OF PARAMETERS                                      DH64 140
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DH64 150
C                     SUBPROGRAM USED.                                  DH64 160
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DH64 170
C                                                                       DH64 180
C        REMARKS                                                        DH64 190
C           NONE                                                        DH64 200
C                                                                       DH64 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DH64 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DH64 230
C           MUST BE FURNISHED BY THE USER.                              DH64 240
C                                                                       DH64 250
C        METHOD                                                         DH64 260
C           EVALUATION IS DONE BY MEANS OF 64-POINT GAUSSIAN-HERMITE    DH64 270
C           QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY WHENEVER       DH64 280
C           FCT(X) IS A POLYNOMIAL UP TO DEGREE 127.                    DH64 290
C           FOR REFERENCE, SEE                                          DH64 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DH64 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DH64 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DH64 330
C           TR00.1100 (MARCH 1964), PP.213-214.                         DH64 340
C                                                                       DH64 350
C     ...................................................................DH64 360
C                                                                       DH64 370
      SUBROUTINE DQH64(FCT,Y)                                           DH64 380
C                                                                       DH64 390
C                                                                       DH64 400
      DOUBLE PRECISION X,Y,Z,FCT                                        DH64 410
C                                                                       DH64 420
      X=.1052612316796054 6D2                                           DH64 430
      Z=-X                                                              DH64 440
      Y=.5535706535856942 8D-48*(FCT(X)+FCT(Z))                         DH64 450
      X=.9895287586829539D1                                             DH64 460
      Z=-X                                                              DH64 470
      Y=Y+.1679747990108159 2D-42*(FCT(X)+FCT(Z))                       DH64 480
      X=.9373159549646721D1                                             DH64 490
      Z=-X                                                              DH64 500
      Y=Y+.3421138011255740 5D-38*(FCT(X)+FCT(Z))                       DH64 510
      X=.8907249099964770D1                                             DH64 520
      Z=-X                                                              DH64 530
      Y=Y+.1557390624629763 8D-34*(FCT(X)+FCT(Z))                       DH64 540
      X=.8477529083379863D1                                             DH64 550
      Z=-X                                                              DH64 560
      Y=Y+.2549660899112999 3D-31*(FCT(X)+FCT(Z))                       DH64 570
      X=.8073687285010225D1                                             DH64 580
      Z=-X                                                              DH64 590
      Y=Y+.1929103595464966 9D-28*(FCT(X)+FCT(Z))                       DH64 600
      X=.7689540164040497D1                                             DH64 610
      Z=-X                                                              DH64 620
      Y=Y+.7861797788925910D-26*(FCT(X)+FCT(Z))                         DH64 630
      X=.7321013032780949D1                                             DH64 640
      Z=-X                                                              DH64 650
      Y=Y+.1911706883300642 8D-23*(FCT(X)+FCT(Z))                       DH64 660
      X=.6965241120551107 5D1                                           DH64 670
      Z=-X                                                              DH64 680
      Y=Y+.2982862784279851 2D-21*(FCT(X)+FCT(Z))                       DH64 690
      X=.6620112262636027 4D1                                           DH64 700
      Z=-X                                                              DH64 710
      Y=Y+.3152254565037814 0D-19*(FCT(X)+FCT(Z))                       DH64 720
      X=.6284011228774828 2D1                                           DH64 730
      Z=-X                                                              DH64 740
      Y=Y+.2351884710675819 1D-17*(FCT(X)+FCT(Z))                       DH64 750
      X=.5955666326799486 0D1                                           DH64 760
      Z=-X                                                              DH64 770
      Y=Y+.1280093391322438 0D-15*(FCT(X)+FCT(Z))                       DH64 780
      X=.5634052164349972 1D1                                           DH64 790
      Z=-X                                                              DH64 800
      Y=Y+.5218623726590847 5D-14*(FCT(X)+FCT(Z))                       DH64 810
      X=.5318325224663270 9D1                                           DH64 820
      Z=-X                                                              DH64 830
      Y=Y+.1628340730709720 4D-12*(FCT(X)+FCT(Z))                       DH64 840
      X=.5007779602198768 2D1                                           DH64 850
      Z=-X                                                              DH64 860
      Y=Y+.3959177766947723 9D-11*(FCT(X)+FCT(Z))                       DH64 870
      X=.4701815647407499 8D1                                           DH64 880
      Z=-X                                                              DH64 890
      Y=Y+.7615217250145451D-10*(FCT(X)+FCT(Z))                         DH64 900
      X=.4399917168228137 6D1                                           DH64 910
      Z=-X                                                              DH64 920
      Y=Y+.1173616742321549 3D-8*(FCT(X)+FCT(Z))                        DH64 930
      X=.4101634474566656 7D1                                           DH64 940
      Z=-X                                                              DH64 950
      Y=Y+.1465125316476109 4D-7*(FCT(X)+FCT(Z))                        DH64 960
      X=.3806571513945360 5D1                                           DH64 970
      Z=-X                                                              DH64 980
      Y=Y+.1495532936727247 1D-6*(FCT(X)+FCT(Z))                        DH64 990
      X=.3514375935740906 2D1                                           DH641000
      Z=-X                                                              DH641010
      Y=Y+.1258340251031184 6D-5*(FCT(X)+FCT(Z))                        DH641020
      X=.3224731291992035 7D1                                           DH641030
      Z=-X                                                              DH641040
      Y=Y+.8788499230850359D-5*(FCT(X)+FCT(Z))                          DH641050
      X=.2937350823004621 8D1                                           DH641060
      Z=-X                                                              DH641070
      Y=Y+.5125929135786274 7D-4*(FCT(X)+FCT(Z))                        DH641080
      X=.2651972435430635 0D1                                           DH641090
      Z=-X                                                              DH641100
      Y=Y+.2509836985130624 9D-3*(FCT(X)+FCT(Z))                        DH641110
      X=.2368354588632401 4D1                                           DH641120
      Z=-X                                                              DH641130
      Y=Y+.1036329099507577 7D-2*(FCT(X)+FCT(Z))                        DH641140
      X=.2086272879881762 0D1                                           DH641150
      Z=-X                                                              DH641160
      Y=Y+.3622586978534458 8D-2*(FCT(X)+FCT(Z))                        DH641170
      X=.1805517171465544 9D1                                           DH641180
      Z=-X                                                              DH641190
      Y=Y+.1075604050987913 7D-1*(FCT(X)+FCT(Z))                        DH641200
      X=.1525889140209863 7D1                                           DH641210
      Z=-X                                                              DH641220
      Y=Y+.2720312895368891 8D-1*(FCT(X)+FCT(Z))                        DH641230
      X=.1247200156943117 9D1                                           DH641240
      Z=-X                                                              DH641250
      Y=Y+.5873998196409943 5D-1*(FCT(X)+FCT(Z))                        DH641260
      X=.9692694230711780D0                                             DH641270
      Z=-X                                                              DH641280
      Y=Y+.1084983493061868 4D0*(FCT(X)+FCT(Z))                         DH641290
      X=.6919223058100445 8D0                                           DH641300
      Z=-X                                                              DH641310
      Y=Y+.1716858423490837 0D0*(FCT(X)+FCT(Z))                         DH641320
      X=.4149888241210786 8D0                                           DH641330
      Z=-X                                                              DH641340
      Y=Y+.2329947860626780 5D0*(FCT(X)+FCT(Z))                         DH641350
      X=.1383022449870097 2D0                                           DH641360
      Z=-X                                                              DH641370
      Y=Y+.2713774249413039 8D0*(FCT(X)+FCT(Z))                         DH641380
      RETURN                                                            DH641390
      END                                                               DH641400
```

Subroutines QA2, QA3, . . . , QA10, DQA4, DQA8, DQA12, DQA16, DQA24, DQA32

These subroutines perform the integration of a given function by associated Gaussian-Laguerre quadrature formulas:

To compute:

$$y = \int_0^\infty \frac{e^{-x} f(x)}{\sqrt{x}} \, dx$$

generalized Gaussian-Laguerre quadrature formulas with n = 2,3, . . . , 10 (or n = 4,8,12,16,24,32) points are used to get approximations:

$$y_n = \sum_{k=1}^{n} A_k^{(n)} f(x_k^{(n)}) \quad (n = 2,3,\ldots) \quad (1)$$

which are exact whenever f(x) is a polynomial up to the degree 2n-1. Note that the nodes $x_k^{(n)}$ are the roots of the associated Laguerre polynomials $L_n^{(-\frac{1}{2})}(x)$ of degree n.

For reference see:

(1) Concus/Cassatt/Jaehnig/Melby, "Tables for the Evaluation of $\int_0^\infty x^\beta \cdot e^{-x} \cdot f(x) \cdot dx$ by Gauss-Laguerre Quadrature", MTAC, vol. 17, No. 83 (1963), pp. 245-256.

(2) Shao/Chen/Frank, "Tables of Zeros and Gaussian Weights of Certain Associated Laguerre Polynomials and the Related Generalized Hermite Polynomials", IBM Technical Report TR 00.1100 (March 1964), pp. 15-16.

```
C                                                                       QA2   10
C     ..................................................................QA2   20
C                                                                       QA2   30
C        SUBROUTINE QA2                                                 QA2   40
C                                                                       QA2   50
C        PURPOSE                                                        QA2   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA2   70
C                              FROM 0 TO INFINITY).                     QA2   80
C                                                                       QA2   90
C        USAGE                                                          QA2  100
C           CALL QA2 (FCT,Y)                                            QA2  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA2  120
C                                                                       QA2  130
C        DESCRIPTION OF PARAMETERS                                      QA2  140
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QA2  150
C           Y       - THE RESULTING INTEGRAL VALUE.                     QA2  160
C                                                                       QA2  170
C        REMARKS                                                        QA2  180
C           NONE                                                        QA2  190
C                                                                       QA2  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA2  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA2  220
C           BY THE USER.                                                QA2  230
C                                                                       QA2  240
C        METHOD                                                         QA2  250
C           EVALUATION IS DONE BY MEANS OF 2-POINT GENERALIZED GAUSSIAN-QA2  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA2  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 3.             QA2  280
C           FOR REFERENCE, SEE                                          QA2  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA2  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA2  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA2  320
C           ISS.83 (1963), PP.245-256.                                  QA2  330
C                                                                       QA2  340
C     ..................................................................QA2  350
C                                                                       QA2  360
      SUBROUTINE QA2(FCT,Y)                                             QA2  370
C                                                                       QA2  380
C                                                                       QA2  390
      X=2.724745                                                        QA2  400
      Y=.1626257*FCT(X)                                                 QA2  410
      X=.2752551                                                        QA2  420
      Y=Y+1.609828*FCT(X)                                               QA2  430
      RETURN                                                            QA2  440
      END                                                               QA2  450
```

```
C                                                                       QA3   10
C     ..................................................................QA3   20
C                                                                       QA3   30
C        SUBROUTINE QA3                                                 QA3   40
C                                                                       QA3   50
C        PURPOSE                                                        QA3   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA3   70
C                              FROM 0 TO INFINITY).                     QA3   80
C                                                                       QA3   90
C        USAGE                                                          QA3  100
C           CALL QA3 (FCT,Y)                                            QA3  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA3  120
C                                                                       QA3  130
C        DESCRIPTION OF PARAMETERS                                      QA3  140
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QA3  150
C           Y       - THE RESULTING INTEGRAL VALUE.                     QA3  160
C                                                                       QA3  170
C        REMARKS                                                        QA3  180
C           NONE                                                        QA3  190
C                                                                       QA3  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA3  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA3  220
C           BY THE USER.                                                QA3  230
C                                                                       QA3  240
C        METHOD                                                         QA3  250
C           EVALUATION IS DONE BY MEANS OF 3-POINT GENERALIZED GAUSSIAN-QA3  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA3  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 5.             QA3  280
C           FOR REFERENCE, SEE                                          QA3  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA3  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA3  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA3  320
C           ISS.83 (1963), PP.245-256.                                  QA3  330
C                                                                       QA3  340
C     ..................................................................QA3  350
C                                                                       QA3  360
      SUBROUTINE QA3(FCT,Y)                                             QA3  370
C                                                                       QA3  380
C                                                                       QA3  390
      X=5.525344                                                        QA3  400
      Y=.009060020*FCT(X)                                               QA3  410
      X=1.784493                                                        QA3  420
      Y=Y+.3141346*FCT(X)                                               QA3  430
      X=.1901635                                                        QA3  440
      Y=Y+1.449259*FCT(X)                                               QA3  450
      RETURN                                                            QA3  460
      END                                                               QA3  470
```

```
C                                                                       QA4   10
C     ..................................................................QA4   20
C                                                                       QA4   30
C        SUBROUTINE QA4                                                 QA4   40
C                                                                       QA4   50
C        PURPOSE                                                        QA4   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA4   70
C                              FROM 0 TO INFINITY).                     QA4   80
C                                                                       QA4   90
C        USAGE                                                          QA4  100
C           CALL QA4 (FCT,Y)                                            QA4  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA4  120
C                                                                       QA4  130
C        DESCRIPTION OF PARAMETERS                                      QA4  140
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QA4  150
C           Y       - THE RESULTING INTEGRAL VALUE.                     QA4  160
C                                                                       QA4  170
C        REMARKS                                                        QA4  180
C           NONE                                                        QA4  190
C                                                                       QA4  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA4  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA4  220
C           BY THE USER.                                                QA4  230
C                                                                       QA4  240
C        METHOD                                                         QA4  250
C           EVALUATION IS DONE BY MEANS OF 4-POINT GENERALIZED GAUSSIAN-QA4  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA4  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 7.             QA4  280
C           FOR REFERENCE, SEE                                          QA4  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA4  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA4  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA4  320
C           ISS.83 (1963), PP.245-256.                                  QA4  330
C                                                                       QA4  340
C     ..................................................................QA4  350
C                                                                       QA4  360
      SUBROUTINE QA4(FCT,Y)                                             QA4  370
C                                                                       QA4  380
C                                                                       QA4  390
      X=8.588636                                                        QA4  400
      Y=.0003992081*FCT(X)                                              QA4  410
      X=3.926964                                                        QA4  420
      Y=Y+.03415597*FCT(X)                                              QA4  430
      X=1.339097                                                        QA4  440
      Y=Y+.4156047*FCT(X)                                               QA4  450
      X=.1453035                                                        QA4  460
      Y=Y+1.322294*FCT(X)                                               QA4  470
      RETURN                                                            QA4  480
      END                                                               QA4  490
```

```
C                                                                       DQA4  10
C     ..................................................................DQA4  20
C                                                                       DQA4  30
C        SUBROUTINE DQA4                                                DQA4  40
C                                                                       DQA4  50
C        PURPOSE                                                        DQA4  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   DQA4  70
C                              FROM 0 TO INFINITY).                     DQA4  80
C                                                                       DQA4  90
C        USAGE                                                          DQA4 100
C           CALL DQA4 (FCT,Y)                                           DQA4 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DQA4 120
C                                                                       DQA4 130
C        DESCRIPTION OF PARAMETERS                                      DQA4 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DQA4 150
C                    SUBPROGRAM USED.                                   DQA4 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DQA4 170
C                                                                       DQA4 180
C        REMARKS                                                        DQA4 190
C           NONE                                                        DQA4 200
C                                                                       DQA4 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQA4 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DQA4 230
C           MUST BE FURNISHED BY THE USER.                              DQA4 240
C                                                                       DQA4 250
C        METHOD                                                         DQA4 260
C           EVALUATION IS DONE BY MEANS OF 4-POINT GENERALIZED GAUSS-   DQA4 270
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY       DQA4 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 7.             DQA4 290
C           FOR REFERENCE, SEE                                          DQA4 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DQA4 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DQA4 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DQA4 330
C           TR00.1100 (MARCH 1964), PP.15-16.                           DQA4 340
C                                                                       DQA4 350
C     ..................................................................DQA4 360
C                                                                       DQA4 370
      SUBROUTINE DQA4(FCT,Y)                                            DQA4 380
C                                                                       DQA4 390
C                                                                       DQA4 400
      DOUBLE PRECISION X,Y,FCT                                          DQA4 410
C                                                                       DQA4 420
      X=.85886356890120340D1                                            DQA4 430
      Y=.39920814442273524D-3*FCT(X)                                    DQA4 440
      X=.39269635013582872D1                                            DQA4 450
      Y=Y+.34155986014826951D-1*FCT(X)                                  DQA4 460
      X=.13390972881263614D1                                            DQA4 470
      Y=Y+.41560465162978376D0*FCT(X)                                   DQA4 480
      X=.14530352150331709D0                                            DQA4 490
      Y=Y+.13222940251164826D1*FCT(X)                                   DQA4 500
      RETURN                                                            DQA4 510
      END                                                               DQA4 520
```

```
C                                                                       QA6   10
C     ..................................................................QA6   20
C                                                                       QA6   30
C        SUBROUTINE QA6                                                 QA6   40
C                                                                       QA6   50
C        PURPOSE                                                        QA6   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA6   70
C                              FROM 0 TO INFINITY).                     QA6   80
C                                                                       QA6   90
C        USAGE                                                          QA6  100
C           CALL QA6 (FCT,Y)                                            QA6  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA6  120
C                                                                       QA6  130
C        DESCRIPTION OF PARAMETERS                                      QA6  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QA6  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QA6  160
C                                                                       QA6  170
C        REMARKS                                                        QA6  180
C           NONE                                                        QA6  190
C                                                                       QA6  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA6  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA6  220
C           BY THE USER.                                                QA6  230
C                                                                       QA6  240
C        METHOD                                                         QA6  250
C           EVALUATION IS DONE BY MEANS OF 6-POINT GENERALIZED GAUSSIAN-QA6  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA6  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 11.            QA6  280
C           FOR REFERENCE, SEE                                          QA6  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA6  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA6  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA6  320
C           ISS.83 (1963), PP.245-256.                                  QA6  330
C                                                                       QA6  340
C     ..................................................................QA6  350
C                                                                       QA6  360
      SUBROUTINE QA6(FCT,Y)                                             QA6  370
C                                                                       QA6  380
C                                                                       QA6  390
      X=15.12996                                                        QA6  400
      Y=.5317103E-6*FCT(X)                                              QA6  410
      X=9.124248                                                        QA6  420
      Y=Y+.0001714737*FCT(X)                                            QA6  430
      X=5.196153                                                        QA6  440
      Y=Y+.007810781*FCT(X)                                             QA6  450
      X=2.552590                                                        QA6  460
      Y=Y+.1032160*FCT(X)                                               QA6  470
      X=.8983028                                                        QA6  480
      Y=Y+.5209846*FCT(X)                                               QA6  490
      X=.09874701                                                       QA6  500
      Y=Y+1.140270*FCT(X)                                               QA6  510
      RETURN                                                            QA6  520
      END                                                               QA6  530
```

```
C                                                                       QA5   10
C     ..................................................................QA5   20
C                                                                       QA5   30
C        SUBROUTINE QA5                                                 QA5   40
C                                                                       QA5   50
C        PURPOSE                                                        QA5   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA5   70
C                              FROM 0 TO INFINITY).                     QA5   80
C                                                                       QA5   90
C        USAGE                                                          QA5  100
C           CALL QA5 (FCT,Y)                                            QA5  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA5  120
C                                                                       QA5  130
C        DESCRIPTION OF PARAMETERS                                      QA5  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QA5  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QA5  160
C                                                                       QA5  170
C        REMARKS                                                        QA5  180
C           NONE                                                        QA5  190
C                                                                       QA5  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA5  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA5  220
C           BY THE USER.                                                QA5  230
C                                                                       QA5  240
C        METHOD                                                         QA5  250
C           EVALUATION IS DONE BY MEANS OF 5-POINT GENERALIZED GAUSSIAN-QA5  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA5  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 9.             QA5  280
C           FOR REFERENCE, SEE                                          QA5  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA5  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA5  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA5  320
C           ISS.83 (1963), PP.245-256.                                  QA5  330
C                                                                       QA5  340
C     ..................................................................QA5  350
C                                                                       QA5  360
      SUBROUTINE QA5(FCT,Y)                                             QA5  370
C                                                                       QA5  380
C                                                                       QA5  390
      X=11.80719                                                        QA5  400
      Y=.1528087E-4*FCT(X)                                              QA5  410
      X=6.414730                                                        QA5  420
      Y=Y+.002687291*FCT(X)                                             QA5  430
      X=3.085937                                                        QA5  440
      Y=Y+.06774879*FCT(X)                                              QA5  450
      X=1.074562                                                        QA5  460
      Y=Y+.4802772*FCT(X)                                               QA5  470
      X=.1175813                                                        QA5  480
      Y=Y+1.221725*FCT(X)                                               QA5  490
      RETURN                                                            QA5  500
      END                                                               QA5  510
```

```
C                                                                       QA7   10
C     ..................................................................QA7   20
C                                                                       QA7   30
C        SUBROUTINE QA7                                                 QA7   40
C                                                                       QA7   50
C        PURPOSE                                                        QA7   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA7   70
C                              FROM 0 TO INFINITY).                     QA7   80
C                                                                       QA7   90
C        USAGE                                                          QA7  100
C           CALL QA7 (FCT,Y)                                            QA7  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA7  120
C                                                                       QA7  130
C        DESCRIPTION OF PARAMETERS                                      QA7  140
C           FCT    - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED.  QA7  150
C           Y      - THE RESULTING INTEGRAL VALUE.                      QA7  160
C                                                                       QA7  170
C        REMARKS                                                        QA7  180
C           NONE                                                        QA7  190
C                                                                       QA7  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA7  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA7  220
C           BY THE USER.                                                QA7  230
C                                                                       QA7  240
C        METHOD                                                         QA7  250
C           EVALUATION IS DONE BY MEANS OF 7-POINT GENERALIZED GAUSSIAN-QA7  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA7  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 13.            QA7  280
C           FOR REFERENCE, SEE                                          QA7  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA7  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA7  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA7  320
C           ISS.83 (1963), PP.245-256.                                  QA7  330
C                                                                       QA7  340
C     ..................................................................QA7  350
C                                                                       QA7  360
      SUBROUTINE QA7(FCT,Y)                                             QA7  370
C                                                                       QA7  380
C                                                                       QA7  390
      X=18.52828                                                        QA7  400
      Y=.1725718E-7*FCT(X)                                              QA7  410
      X=11.98999                                                        QA7  420
      Y=Y+.9432969E-5*FCT(X)                                            QA7  430
      X=7.554091                                                        QA7  440
      Y=Y+.0007101852*FCT(X)                                            QA7  450
      X=4.389793                                                        QA7  460
      Y=Y+.01570011*FCT(X)                                              QA7  470
      X=2.180592                                                        QA7  480
      Y=Y+.1370111*FCT(X)                                               QA7  490
      X=.7721379                                                        QA7  500
      Y=Y+.5462112*FCT(X)                                               QA7  510
      X=.08511544                                                       QA7  520
      Y=Y+1.072812*FCT(X)                                               QA7  530
      RETURN                                                            QA7  540
      END                                                               QA7  550
```

```
C                                                                       QA8   10
C     ..................................................................QA8   20
C                                                                       QA8   30
C        SUBROUTINE QA8                                                 QA8   40
C                                                                       QA8   50
C        PURPOSE                                                        QA8   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA8   70
C                              FROM 0 TO INFINITY).                     QA8   80
C                                                                       QA8   90
C        USAGE                                                          QA8  100
C           CALL QA8 (FCT,Y)                                            QA8  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA8  120
C                                                                       QA8  130
C        DESCRIPTION OF PARAMETERS                                      QA8  140
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QA8  150
C           Y       - THE RESULTING INTEGRAL VALUE.                     QA8  160
C                                                                       QA8  170
C        REMARKS                                                        QA8  180
C           NONE                                                        QA8  190
C                                                                       QA8  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA8  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA8  220
C           BY THE USER.                                                QA8  230
C                                                                       QA8  240
C        METHOD                                                         QA8  250
C           EVALUATION IS DONE BY MEANS OF 8-POINT GENERALIZED GAUSSIAN-QA8  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA8  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 15.            QA8  280
C           FOR REFERENCE, SEE                                          QA8  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA8  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA8  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA8  320
C           ISS.83 (1963), PP.245-256.                                  QA8  330
C                                                                       QA8  340
C     ..................................................................QA8  350
C                                                                       QA8  360
      SUBROUTINE QA8(FCT,Y)                                             QA8  370
C                                                                       QA8  380
C                                                                       QA8  390
      X=21.99427                                                        QA8  400
      Y=.5309615E-9*FCT(X)                                              QA8  410
      X=14.97262                                                        QA8  420
      Y=Y+.4641962E-6*FCT(X)                                            QA8  430
      X=10.09332                                                        QA8  440
      Y=Y+.5423720E-4*FCT(X)                                            QA8  450
      X=6.483145                                                        QA8  460
      Y=Y+.001864568*FCT(X)                                             QA8  470
      X=3.809476                                                        QA8  480
      Y=Y+.02576062*FCT(X)                                              QA8  490
      X=1.905114                                                        QA8  500
      Y=Y+.1676201*FCT(X)                                               QA8  510
      X=.6772491                                                        QA8  520
      Y=Y+.5612949*FCT(X)                                               QA8  530
      X=.07479188                                                       QA8  540
      Y=Y+1.015859*FCT(X)                                               QA8  550
      RETURN                                                            QA8  560
      END                                                               QA8  570
```

```
C                                                                       QA9   10
C     ..................................................................QA9   20
C                                                                       QA9   30
C        SUBROUTINE QA9                                                 QA9   40
C                                                                       QA9   50
C        PURPOSE                                                        QA9   60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA9   70
C                              FROM 0 TO INFINITY).                     QA9   80
C                                                                       QA9   90
C        USAGE                                                          QA9  100
C           CALL QA9 (FCT,Y)                                            QA9  110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA9  120
C                                                                       QA9  130
C        DESCRIPTION OF PARAMETERS                                      QA9  140
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QA9  150
C           Y       - THE RESULTING INTEGRAL VALUE.                     QA9  160
C                                                                       QA9  170
C        REMARKS                                                        QA9  180
C           NONE                                                        QA9  190
C                                                                       QA9  200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA9  210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA9  220
C           BY THE USER.                                                QA9  230
C                                                                       QA9  240
C        METHOD                                                         QA9  250
C           EVALUATION IS DONE BY MEANS OF 9-POINT GENERALIZED GAUSSIAN-QA9  260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA9  270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 17.            QA9  280
C           FOR REFERENCE, SEE                                          QA9  290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA9  300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA9  310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA9  320
C           ISS.83 (1963), PP.245-256.                                  QA9  330
C                                                                       QA9  340
C     ..................................................................QA9  350
C                                                                       QA9  360
      SUBROUTINE QA9(FCT,Y)                                             QA9  370
C                                                                       QA9  380
C                                                                       QA9  390
      X=25.48598                                                        QA9  400
      Y=.1565640E-10*FCT(X)                                             QA9  410
      X=18.04651                                                        QA9  420
      Y=Y+.2093441E-7*FCT(X)                                            QA9  430
      X=12.77183                                                        QA9  440
      Y=Y+.3621309E-5*FCT(X)                                            QA9  450
      X=8.769757                                                        QA9  460
      Y=Y+.00018362 25*FCT(X)                                           QA9  470
      X=5.694423                                                        QA9  480
      Y=Y+.003777045*FCT(X)                                             QA9  490
      X=3.369176                                                        QA9  500
      Y=Y+.03728008*FCT(X)                                              QA9  510
      X=1.692395                                                        QA9  520
      Y=Y+.1946035*FCT(X)                                               QA9  530
      X=.6032364                                                        QA9  540
      Y=Y+.5696146*FCT(X)                                               QA9  550
      X=.06670223                                                       QA9  560
      Y=Y+.9669914*FCT(X)                                               QA9  570
      RETURN                                                            QA9  580
      END                                                               QA9  590
```

```
C                                                                       DQA8  10
C     ..................................................................DQA8  20
C                                                                       DQA8  30
C        SUBROUTINE DQA8                                                DQA8  40
C                                                                       DQA8  50
C        PURPOSE                                                        DQA8  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   DQA8  70
C                              FROM 0 TO INFINITY).                     DQA8  80
C                                                                       DQA8  90
C        USAGE                                                          DQA8 100
C           CALL DQA8 (FCT,Y)                                           DQA8 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DQA8 120
C                                                                       DQA8 130
C        DESCRIPTION OF PARAMETERS                                      DQA8 140
C           FCT     - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION DQA8 150
C                     SUBPROGRAM USED.                                  DQA8 160
C           Y       - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.    DQA8 170
C                                                                       DQA8 180
C        REMARKS                                                        DQA8 190
C           NONE                                                        DQA8 200
C                                                                       DQA8 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DQA8 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DQA8 230
C           MUST BE FURNISHED BY THE USER.                              DQA8 240
C                                                                       DQA8 250
C        METHOD                                                         DQA8 260
C           EVALUATION IS DONE BY MEANS OF 8-POINT GENERALIZED GAUSS-   DQA8 270
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY       DQA8 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 15.            DQA8 290
C           FOR REFERENCE, SEE                                          DQA8 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DQA8 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DQA8 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DQA8 330
C           TR00.1100 (MARCH 1964), PP.15-16.                           DQA8 340
C                                                                       DQA8 350
C     ..................................................................DQA8 360
C                                                                       DQA8 370
      SUBROUTINE DQA8(FCT,Y)                                            DQA8 380
C                                                                       DQA8 390
C                                                                       DQA8 400
      DOUBLE PRECISION X,Y,FCT                                          DQA8 410
C                                                                       DQA8 420
      X=.2198427284096265 1D2                                           DQA8 430
      Y=.5309614948022364 5D-9*FCT(X)                                   DQA8 440
      X=.1497262708842639 3D2                                           DQA8 450
      Y=Y+.4641961689730421 3D-6*FCT(X)                                 DQA8 460
      X=.1009332367522134 3D2                                           DQA8 470
      Y=Y+.5423720185075763 0D-4*FCT(X)                                 DQA8 480
      X=.6483145428627170 4D1                                           DQA8 490
      Y=Y+.1864568017248361 1D-2*FCT(X)                                 DQA8 500
      X=.3809476361484907 1D1                                           DQA8 510
      Y=Y+.2576062307101994 7D-1*FCT(X)                                 DQA8 520
      X=.1905113635031428 4D1                                           DQA8 530
      Y=Y+.1676200827979716 6D0*FCT(X)                                  DQA8 540
      X=.6772490876492891 5D0                                           DQA8 550
      Y=Y+.5612949170570673 5D0*FCT(X)                                  DQA8 560
      X=.7479188259681827D-1                                            DQA8 570
      Y=Y+.1015859580333227 5D1*FCT(X)                                  DQA8 580
      RETURN                                                            DQA8 590
      END                                                               DQA8 600
```

```
C                                                                       QA10  10
C     ..................................................................QA10  20
C                                                                       QA10  30
C        SUBROUTINE QA10                                                QA10  40
C                                                                       QA10  50
C        PURPOSE                                                        QA10  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   QA10  70
C                              FROM 0 TO INFINITY).                     QA10  80
C                                                                       QA10  90
C        USAGE                                                          QA10 100
C           CALL QA10 (FCT,Y)                                           QA10 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                QA10 120
C                                                                       QA10 130
C        DESCRIPTION OF PARAMETERS                                      QA10 140
C           FCT     - THE NAME OF AN EXTERNAL FUNCTION SUBPROGRAM USED. QA10 150
C           Y       - THE RESULTING INTEGRAL VALUE.                     QA10 160
C                                                                       QA10 170
C        REMARKS                                                        QA10 180
C           NONE                                                        QA10 190
C                                                                       QA10 200
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QA10 210
C           THE EXTERNAL FUNCTION SUBPROGRAM FCT(X) MUST BE FURNISHED   QA10 220
C           BY THE USER.                                                QA10 230
C                                                                       QA10 240
C        METHOD                                                         QA10 250
C           EVALUATION IS DONE BY MEANS OF 10-POINT GENERALIZED GAUSS-  QA10 260
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY,      QA10 270
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 19.            QA10 280
C           FOR REFERENCE, SEE                                          QA10 290
C           CONCUS/CASSATT/JAEHNIG/MELBY, TABLES FOR THE EVALUATION OF  QA10 300
C           INTEGRAL(X**BETA*EXP(-X)*F(X), SUMMED OVER X FROM 0 TO      QA10 310
C           INFINITY) BY GAUSS-LAGUERRE QUADRATURE, MTAC, VOL.17,       QA10 320
C           ISS.83 (1963), PP.245-256.                                  QA10 330
C                                                                       QA10 340
C     ..................................................................QA10 350
C                                                                       QA10 360
      SUBROUTINE QA10(FCT,Y)                                            QA10 370
C                                                                       QA10 380
C                                                                       QA10 390
      X=29.02495                                                        QA10 400
      Y=.4458787E-12*FCT(X)                                             QA10 410
      X=21.19389                                                        QA10 420
      Y=Y+.8798682E-9*FCT(X)                                            QA10 430
      X=15.56116                                                        QA10 440
      Y=Y+.2172139E-6*FCT(X)                                            QA10 450
      X=11.20813                                                        QA10 460
      Y=Y+.1560511E-4*FCT(X)                                            QA10 470
      X=7.777439                                                        QA10 480
      Y=Y+.0004566773*FCT(X)                                            QA10 490
      X=5.084908                                                        QA10 500
      Y=Y+.006487547*FCT(X)                                             QA10 510
      X=3.022513                                                        QA10 520
      Y=Y+.04962104*FCT(X)                                              QA10 530
      X=1.522944                                                        QA10 540
      Y=Y+.2180344*FCT(X)                                               QA10 550
      X=.5438675                                                        QA10 560
      Y=Y+.5733510*FCT(X)                                               QA10 570
      X=.06019206                                                       QA10 580
      Y=Y+.9244873*FCT(X)                                               QA10 590
      RETURN                                                            QA10 600
      END                                                               QA10 610
```

```
C                                                                       DA12  10
C     ..................................................................DA12  20
C                                                                       DA12  30
C        SUBROUTINE DQA12                                               DA12  40
C                                                                       DA12  50
C        PURPOSE                                                        DA12  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   DA12  70
C                              FROM 0 TO INFINITY).                     DA12  80
C                                                                       DA12  90
C        USAGE                                                          DA12 100
C           CALL DQA12 (FCT,Y)                                          DA12 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DA12 120
C                                                                       DA12 130
C        DESCRIPTION OF PARAMETERS                                      DA12 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DA12 150
C                    SUBPROGRAM USED.                                   DA12 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DA12 170
C                                                                       DA12 180
C        REMARKS                                                        DA12 190
C           NONE                                                        DA12 200
C                                                                       DA12 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DA12 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DA12 230
C           MUST BE FURNISHED BY THE USER.                              DA12 240
C                                                                       DA12 250
C        METHOD                                                         DA12 260
C           EVALUATION IS DONE BY MEANS OF 12-POINT GENERALIZED GAUSS-  DA12 270
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY       DA12 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 23.            DA12 290
C           FOR REFERENCE, SEE                                          DA12 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DA12 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DA12 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DA12 330
C           TR00.1100 (MARCH 1964), PP.15-16.                           DA12 340
C                                                                       DA12 350
C     ..................................................................DA12 360
C                                                                       DA12 370
      SUBROUTINE DQA12(FCT,Y)                                           DA12 380
C                                                                       DA12 390
C                                                                       DA12 400
      DOUBLE PRECISION X,Y,FCT                                          DA12 410
C                                                                       DA12 420
      X=.36191360360615602D2                                            DA12 430
      Y=.33287369929782177D-15*FCT(X)                                   DA12 440
      X=.27661108779846090D2                                            DA12 450
      Y=Y+.13169240486156340D-11*FCT(X)                                 DA12 460
      X=.21396755936166109D2                                            DA12 470
      Y=Y+.60925085399751278D-9*FCT(X)                                  DA12 480
      X=.16432195087675313D2                                            DA12 490
      Y=Y+.80379423498828590D-7*FCT(X)                                  DA12 500
      X=.12390447963809471D2                                            DA12 510
      Y=Y+.43164914098046673D-5*FCT(X)                                  DA12 520
      X=.90754342309612031D1                                            DA12 530
      Y=Y+.11377383272808760D-3*FCT(X)                                  DA12 540
      X=.63699753880306349D1                                            DA12 550
      Y=Y+.16473849653768349D-2*FCT(X)                                  DA12 560
      X=.41984156448784132D1                                            DA12 570
      Y=Y+.14096711620145342D-1*FCT(X)                                  DA12 580
      X=.25098480972321280D1                                            DA12 590
      Y=Y+.74890941006461490D-1*FCT(X)                                  DA12 600
      X=.12695899401039615D1                                            DA12 610
      Y=Y+.25547924356911832D0*FCT(X)                                   DA12 620
      X=.45450668156378028D0                                            DA12 630
      Y=Y+.57235907069288860400*FCT(X)                                  DA12 640
      X=.50361889117293951D-1                                           DA12 650
      Y=Y+.85386232773739850D*FCT(X)                                    DA12 660
      RETURN                                                            DA12 670
      END                                                               DA12 680
```

```
C                                                                       DA16  10
C     ..................................................................DA16  20
C                                                                       DA16  30
C        SUBROUTINE DQA16                                               DA16  40
C                                                                       DA16  50
C        PURPOSE                                                        DA16  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   DA16  70
C                              FROM 0 TO INFINITY).                     DA16  80
C                                                                       DA16  90
C        USAGE                                                          DA16 100
C           CALL DQA16 (FCT,Y)                                          DA16 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DA16 120
C                                                                       DA16 130
C        DESCRIPTION OF PARAMETERS                                      DA16 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DA16 150
C                    SUBPROGRAM USED.                                   DA16 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DA16 170
C                                                                       DA16 180
C        REMARKS                                                        DA16 190
C           NONE                                                        DA16 200
C                                                                       DA16 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DA16 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DA16 230
C           MUST BE FURNISHED BY THE USER.                              DA16 240
C                                                                       DA16 250
C        METHOD                                                         DA16 260
C           EVALUATION IS DONE BY MEANS OF 16-POINT GENERALIZED GAUSS-  DA16 270
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY       DA16 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 31.            DA16 290
C           FOR REFERENCE, SEE                                          DA16 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DA16 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DA16 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DA16 330
C           TR00.1100 (MARCH 1964), PP.15-16.                           DA16 340
C                                                                       DA16 350
C     ..................................................................DA16 360
C                                                                       DA16 370
      SUBROUTINE DQA16(FCT,Y)                                           DA16 380
C                                                                       DA16 390
C                                                                       DA16 400
      DOUBLE PRECISION X,Y,FCT                                          DA16 410
C                                                                       DA16 420
      X=.50777223877537080D2                                            DA16 430
      Y=.14621352854768325D-21*FCT(X)                                   DA16 440
      X=.41081666525491202D2                                            DA16 450
      Y=Y+.18463473073036584D-17*FCT(X)                                 DA16 460
      X=.33781970488226166D2                                            DA16 470
      Y=Y+.23946880341856973D-14*FCT(X)                                 DA16 480
      X=.27831438211328676D2                                            DA16 490
      Y=Y+.84300204226528950D-12*FCT(X)                                 DA16 500
      X=.22821300693525208D2                                            DA16 510
      Y=Y+.11866582926793277D-9*FCT(X)                                  DA16 520
      X=.18537743178606694D2                                            DA16 530
      Y=Y+.81976643295417930-8*FCT(X)                                   DA16 540
      X=.14851431341801250D2                                            DA16 550
      Y=Y+.31483355850911881D-6*FCT(X)                                  DA16 560
      X=.11677033673975957D2                                            DA16 570
      Y=Y+.73011702591247520-5*FCT(X)                                   DA16 580
      X=.89550013377233900D1                                            DA16 590
      Y=Y+.10833168123639965D-3*FCT(X)                                  DA16 600
      X=.66422151797414440D1                                            DA16 610
      Y=Y+.10725367310559441D-2*FCT(X)                                  DA16 620
      X=.47067267076675872D1                                            DA16 630
      Y=Y+.73097806533088560-2*FCT(X)                                   DA16 640
      X=.31246010507021443D1                                            DA16 650
      Y=Y+.35106857663146861D-1*FCT(X)                                  DA16 660
      X=.18779315076960743D1                                            DA16 670
      Y=Y+.12091626191182523D0*FCT(X)                                   DA16 680
      X=.95355315539086550D0                                            DA16 690
      Y=Y+.30253946815328497D0*FCT(X)                                   DA16 700
      X=.34220015601094768D0                                            DA16 710
      Y=Y+.55491628460505980D0*FCT(X)                                   DA16 720
      X=.37962914575134550-1                                            DA16 730
      Y=Y+.75047670518560480*FCT(X)                                     DA16 740
      RETURN                                                            DA16 750
      END                                                               DA16 760
```

```
C                                                                       DA24  10
C     ...................................................................DA24  20
C                                                                       DA24  30
C        SUBROUTINE DQA24                                               DA24  40
C                                                                       DA24  50
C        PURPOSE                                                        DA24  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   DA24  70
C                              FROM 0 TO INFINITY).                     DA24  80
C                                                                       DA24  90
C        USAGE                                                          DA24 100
C           CALL DQA24 (FCT,Y)                                          DA24 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DA24 120
C                                                                       DA24 130
C        DESCRIPTION OF PARAMETERS                                      DA24 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DA24 150
C                    SUBPROGRAM USED.                                   DA24 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DA24 170
C                                                                       DA24 180
C        REMARKS                                                        DA24 190
C           NONE                                                        DA24 200
C                                                                       DA24 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DA24 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DA24 230
C           MUST BE FURNISHED BY THE USER.                              DA24 240
C                                                                       DA24 250
C        METHOD                                                         DA24 260
C           EVALUATION IS DONE BY MEANS OF 24-POINT GENERALIZED GAUSS-  DA24 270
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY       DA24 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 47.            DA24 290
C           FOR REFERENCE, SEE                                          DA24 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DA24 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DA24 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DA24 330
C           TR00.1100 (MARCH 1964), PP.15-16.                           DA24 340
C                                                                       DA24 350
C     ...................................................................DA24 360
C                                                                       DA24 370
      SUBROUTINE DQA24(FCT,Y)                                           DA24 380
C                                                                       DA24 390
C                                                                       DA24 400
      DOUBLE PRECISION X,Y,FCT                                          DA24 410
C                                                                       DA24 420
      X=.8055628081995041D2                                             DA24 430
      Y=.1587110292154799 4D-34*FCT(X)                                  DA24 440
      X=.6906860197530436 9D2                                           DA24 450
      Y=Y+.1196922538662775 7D-29*FCT(X)                                DA24 460
      X=.6020666696305722 3D2                                           DA24 470
      Y=Y+.7370072160301340D-26*FCT(X)                                  DA24 480
      X=.5279543252728363 0D2                                           DA24 490
      Y=Y+.1112915493780457 0D-22*FCT(X)                                DA24 500
      X=.4637697955754013 3D2                                           DA24 510
      Y=Y+.6376774647010276 9D-20*FCT(X)                                DA24 520
      X=.4071159818554310 7D2                                           DA24 530
      Y=Y+.1746031920233733 53D-17*FCT(X)                               DA24 540
      X=.3565370351632821 2D2                                           DA24 550
      Y=Y+.2630319245316817 0D-15*FCT(X)                                DA24 560
      X=.3110646470904656 5D2                                           DA24 570
      Y=Y+.2395179730958358 7D-13*FCT(X)                                DA24 580
      X=.2700140605647235 6D2                                           DA24 590
      Y=Y+.1409386516309177 8D-11*FCT(X)                                DA24 600
      X=.2328793282487991 7D2                                           DA24 610
      Y=Y+.5630593075676338 2D-10*FCT(X)                                DA24 620
      X=.1992742587524246 2D2                                           DA24 630
      Y=Y+.1586093499033076 5D-8*FCT(X)                                 DA24 640
      X=.1688967192852710 8D2                                           DA24 650
      Y=Y+.3245028271791539 7D-7*FCT(X)                                 DA24 660
      X=.1415058618728575 9D2                                           DA24 670
      Y=Y+.4937317987339501 0D-6*FCT(X)                                 DA24 680
      X=.1169069592605607 3D2                                           DA24 690
      Y=Y+.5694517383469696 2D-5*FCT(X)                                 DA24 700
      X=.9494095330026488 D1                                            DA24 710
      Y=Y+.5057198055496977 8D-4*FCT(X)                                 DA24 720
      X=.7547704680023454 D1                                            DA24 730
      Y=Y+.3503008636023456 6D-3*FCT(X)                                 DA24 740
      X=.5840733271323608 0D1                                           DA24 750
      Y=Y+.1912784639638830 6D-2*FCT(X)                                 DA24 760
      X=.4364283076935306 2D1                                           DA24 770
      Y=Y+.8306009823955105 D-2*FCT(X)                                  DA24 780
      X=.3111052455147713 0D1                                           DA24 790
      Y=Y+.2888992314996219 9D-1*FCT(X)                                 DA24 800
      X=.2075112909852380 6D1                                           DA24 810
      Y=Y+.8095935396920770 D-1*FCT(X)                                  DA24 820
      X=.1251740632362746 4D1                                           DA24 830
      Y=Y+.1836445941585703 6D0*FCT(X)                                  DA24 840
      X=.6372902787326687 9D0                                           DA24 850
      Y=Y+.3384089438912822 1D0*FCT(X)                                  DA24 860
      X=.2291023164926243 3D0                                           DA24 870
      Y=Y+.5079230853295182 0D0*FCT(X)                                  DA24 880
      X=.2543799658568935 90-1                                          DA24 890
      Y=Y+.6220020607559261 6D0*FCT(X)                                  DA24 900
      RETURN                                                            DA24 910
      END                                                               DA24 920
```

```
C                                                                       DA32  10
C     ...................................................................DA32  20
C                                                                       DA32  30
C        SUBROUTINE DQA32                                               DA32  40
C                                                                       DA32  50
C        PURPOSE                                                        DA32  60
C           TO COMPUTE INTEGRAL(EXP(-X)*FCT(X)/SQRT(X), SUMMED OVER X   DA32  70
C                              FROM 0 TO INFINITY).                     DA32  80
C                                                                       DA32  90
C        USAGE                                                          DA32 100
C           CALL DQA32 (FCT,Y)                                          DA32 110
C           PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                DA32 120
C                                                                       DA32 130
C        DESCRIPTION OF PARAMETERS                                      DA32 140
C           FCT    - THE NAME OF AN EXTERNAL DOUBLE PRECISION FUNCTION  DA32 150
C                    SUBPROGRAM USED.                                   DA32 160
C           Y      - THE RESULTING DOUBLE PRECISION INTEGRAL VALUE.     DA32 170
C                                                                       DA32 180
C        REMARKS                                                        DA32 190
C           NONE                                                        DA32 200
C                                                                       DA32 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DA32 220
C           THE EXTERNAL DOUBLE PRECISION FUNCTION SUBPROGRAM FCT(X)    DA32 230
C           MUST BE FURNISHED BY THE USER.                              DA32 240
C                                                                       DA32 250
C        METHOD                                                         DA32 260
C           EVALUATION IS DONE BY MEANS OF 32-POINT GENERALIZED GAUSS-  DA32 270
C           LAGUERRE QUADRATURE FORMULA, WHICH INTEGRATES EXACTLY       DA32 280
C           WHENEVER FCT(X) IS A POLYNOMIAL UP TO DEGREE 63.            DA32 290
C           FOR REFERENCE, SEE                                          DA32 300
C           SHAO/CHEN/FRANK, TABLES OF ZEROS AND GAUSSIAN WEIGHTS OF    DA32 310
C           CERTAIN ASSOCIATED LAGUERRE POLYNOMIALS AND THE RELATED     DA32 320
C           GENERALIZED HERMITE POLYNOMIALS, IBM TECHNICAL REPORT       DA32 330
C           TR00.1100 (MARCH 1964), PP.15-16.                           DA32 340
C                                                                       DA32 350
C     ...................................................................DA32 360
C                                                                       DA32 370
      SUBROUTINE DQA32(FCT,Y)                                           DA32 380
C                                                                       DA32 390
C                                                                       DA32 400
      DOUBLE PRECISION X,Y,FCT                                          DA32 410
C                                                                       DA32 420
      X=.1107992689470757 6D3                                           DA32 430
      Y=.1107141307171388 6D-47*FCT(X)                                  DA32 440
      X=.9791671642606276 D2                                            DA32 450
      Y=Y+.3359495980216318 4D-42*FCT(X)                                DA32 460
      X=.8785611994313352 D2                                            DA32 470
      Y=Y+.6842276022511481 0D-38*FCT(X)                                DA32 480
      X=.7933908652882320 D2                                            DA32 490
      Y=Y+.3114781249259527 6D-34*FCT(X)                                DA32 500
      X=.7186849935955142 D2                                            DA32 510
      Y=Y+.5099321798225998 5D-31*FCT(X)                                DA32 520
      X=.6518442637613578 D2                                            DA32 530
      Y=Y+.3858207190929933 7D-28*FCT(X)                                DA32 540
      X=.5912902793439195 1D2                                           DA32 550
      Y=Y+.1572359557785182 1D-25*FCT(X)                                DA32 560
      X=.5359723182614851 2D2                                           DA32 570
      Y=Y+.3823413766601285 7D-23*FCT(X)                                DA32 580
      X=.4851458386741604 8D2                                           DA32 590
      Y=Y+.5965725568559702 3D-21*FCT(X)                                DA32 600
      X=.4382588636990390 2D2                                           DA32 610
      Y=Y+.6304509133007562 8D-19*FCT(X)                                DA32 620
      X=.3948879712336812 7D2                                           DA32 630
      Y=Y+.4703769421351638 2D-17*FCT(X)                                DA32 640
      X=.3546996139617328 3D2                                           DA32 650
      Y=Y+.2560186782644876 1D-15*FCT(X)                                DA32 660
      X=.3174254379061660 6D2                                           DA32 670
      Y=Y+.1043724745318169 5D-13*FCT(X)                                DA32 680
      X=.2828458319497053 1D2                                           DA32 690
      Y=Y+.3256681461419440 7D-12*FCT(X)                                DA32 700
      X=.2507785654419805 3D2                                           DA32 710
      Y=Y+.7918355533895448 D-11*FCT(X)                                 DA32 720
      X=.2210707038220600 7D2                                           DA32 730
      Y=Y+.1523034350029090 3D-9*FCT(X)                                 DA32 740
      X=.1935927108726871 4D2                                           DA32 750
      Y=Y+.2347233484643098 7D-8*FCT(X)                                 DA32 760
      X=.1682340536295369 4D2                                           DA32 770
      Y=Y+.2930250632952218 7D-7*FCT(X)                                 DA32 780
      X=.1448998669078027 4D2                                           DA32 790
      Y=Y+.2991065873454494 1D-6*FCT(X)                                 DA32 800
      X=.1235083821771477 0D2                                           DA32 810
      Y=Y+.2516680502062369 2D-5*FCT(X)                                 DA32 820
      X=.1039889190555262 4D2                                           DA32 830
      Y=Y+.1757699846170071 8D-4*FCT(X)                                 DA32 840
      X=.8628029857405929 D1                                            DA32 850
      Y=Y+.1025185827157254 9D-3*FCT(X)                                 DA32 860
      X=.7032957798238936 D1                                            DA32 870
      Y=Y+.5019673970261249 7D-3*FCT(X)                                 DA32 880
      X=.5609103457496151 3D1                                           DA32 890
      Y=Y+.2072658199015155 3D-2*FCT(X)                                 DA32 900
      X=.4352534529330141 0D1                                           DA32 910
      Y=Y+.7245173957068918 D-2*FCT(X)                                  DA32 920
      X=.3259892256456941 9D1                                           DA32 930
      Y=Y+.2151208101975874 0D-1*FCT(X)                                 DA32 940
      X=.2328337668210397 0D1                                           DA32 950
      Y=Y+.5440625790737783 7D-1*FCT(X)                                 DA32 960
      X=.1555508231478938 0D1                                           DA32 970
      Y=Y+.1174799639281988 7D0*FCT(X)                                  DA32 980
      X=.9394832145007343 0D0                                           DA32 990
      Y=Y+.2169966986123736 8D0*FCT(X)                                  DA321000
      X=.4787564772774888 5D0                                           DA321010
      Y=Y+.3433716846981674 0D0*FCT(X)                                  DA321020
      X=.1722157241453955 8D0                                           DA321030
      Y=Y+.4659895721253560 9D0*FCT(X)                                  DA321040
      X=.1912751096844685 6D-1                                          DA321050
      Y=Y+.5427548498826079 6D0*FCT(X)                                  DA321060
      RETURN                                                            DA321070
      END                                                               DA321080
```

Numerical Differentiation

Subroutines DGT3 and DDGT3

These subroutines compute a vector $Z=(z_1,\ldots,z_n)$ of derivative values, given vectors $X=(x_1,\ldots,x_n)$ of argument values and $Y=(y_1,\ldots y_n)$ of corresponding function values. Except at the endpoints x_1 and x_n, z_i is the derivative at x_i of the Lagrangian interpolation polynomial of degree 2 relevant to the three successive points (x_{i-1}, y_{i-1}), (x_i, y_i) and (x_{i+1}, y_{i+1}).

1. Mathematical background

For $i=1,\ldots,n-2$ we must find a_i, b_i, and c_i such that

$$\bar{y}_i(x) = a_i x^2 + b_i x + c_i$$

passes through (x_i, y_i), (x_{i+1}, y_{i+1}), and (x_{i+2}, y_{i+2}).

The desired derivative values z_i are given by:

$$z_i = \begin{cases} y_1'(x_1) = 2a_1x_1 + b_1 & \text{if } i=1 \\ y_{i-1}'(x_i) = 2a_{i-1}x_i + b_{i-1} & \text{if } i=2,\ldots,n-1 \\ y_{n-2}'(x_n) = 2a_{n-2}x_n + b_{n-2} & \text{if } i=n \end{cases}$$

An easy computation yields:

$$z_i = \begin{cases} \dfrac{y_2-y_1}{x_2-x_1} + \dfrac{y_3-y_1}{x_3-x_1} - \dfrac{y_3-y_2}{x_3-x_2} & \text{if } i=1 \\[2ex] \dfrac{y_i-y_{i-1}}{x_i-x_{i-1}} + \dfrac{y_{i+1}-y_i}{x_{i+1}-x_i} - \dfrac{y_{i+1}-y_{i-1}}{x_{i+1}-x_{i-1}} & \text{if } i=2,\ldots,n-1 \\[2ex] \dfrac{y_n-y_{n-1}}{x_n-x_{n-1}} + \dfrac{y_n-y_{n-2}}{x_n-x_{n-2}} - \dfrac{y_{n-1}-y_{n-2}}{x_{n-1}-x_{n-2}} & \text{if } i=n \end{cases} \qquad (1)$$

Assuming that the vectors X and Y represent a portion of a three-times differentiable function, z_i involves a truncation error T_i where:

$$T_i = \begin{cases} \frac{1}{6}(x_1-x_2)(x_1-x_3)y'''(\xi_1) & \text{if } i=1 \\ \frac{1}{6}(x_i-x_{i-1})(x_i-x_{i+1})y'''(\xi_i) & \text{if } i=2,\ldots,n-1 \\ \frac{1}{6}(x_n-x_{n-2})(x_n-x_{n-1})y'''(\xi_n) & \text{if } i=n \end{cases}$$

and ξ_i is in the closed interval determined by the three argument values used in computing z_i, $i=1,\ldots,n$.

2. Programming considerations

The subroutines compute the z_i in serial order according to (1). If $n<3$, there is no computation, and the error parameter is set to -1. If two of the three arguments x_{j-2}, x_{j-i}, x_j are identical, $j=3,\ldots,n$, the computation is prematurely terminated, and the error parameter IER is set to j. However, $z_1,\ldots,z_{j-1}$ are returned if $j>3$. In this case z_{j-i} is computed treating x_{j-1} as the right-hand endpoint. If there is no error, IER is set to zero at the end of the computation. The output vector Z can have the same storage allocation as X or Y.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 64-68.

```
C                                                                       DGT3  10
C     ...................................................................DGT3  20
C                                                                       DGT3  30
C       SUBROUTINE DGT3                                                 DGT3  40
C                                                                       DGT3  50
C       PURPOSE                                                         DGT3  60
C          TO COMPUTE A VECTOR OF DERIVATIVE VALUES GIVEN VECTORS OF    DGT3  70
C          ARGUMENT VALUES AND CORRESPONDING FUNCTION VALUES.           DGT3  80
C                                                                       DGT3  90
C       USAGE                                                           DGT3 100
C          CALL DGT3(X,Y,Z,NDIM,IER)                                    DGT3 110
C                                                                       DGT3 120
C       DESCRIPTION OF PARAMETERS                                       DGT3 130
C          X     -  GIVEN VECTOR OF ARGUMENT VALUES (DIMENSION NDIM)    DGT3 140
C          Y     -  GIVEN VECTOR OF FUNCTION VALUES CORRESPONDING TO X  DGT3 150
C                   (DIMENSION NDIM)                                    DGT3 160
C          Z     -  RESULTING VECTOR OF DERIVATIVE VALUES (DIMENSION    DGT3 170
C                   NDIM)                                               DGT3 180
C          NDIM  -  DIMENSION OF VECTORS X,Y AND Z                      DGT3 190
C          IER   -  RESULTING ERROR PARAMETER                           DGT3 200
C                   IER  = -1  - NDIM IS LESS THAN 3                    DGT3 210
C                   IER  =  0  - NO ERROR                               DGT3 220
C                   IER POSITIVE  - X(IER) = X(IER-1) OR X(IER) =       DGT3 230
C                                   X(IER-2)                            DGT3 240
C                                                                       DGT3 250
C       REMARKS                                                         DGT3 260
C          (1)   IF IER = -1,2,3, THEN THERE IS NO COMPUTATION.         DGT3 270
C          (2)   IF IER =  4,...,N, THEN THE DERIVATIVE VALUES Z(1)     DGT3 280
C                ,..., Z(IER-1) HAVE BEEN COMPUTED.                     DGT3 290
C          (3)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS X OR Y. IF   DGT3 300
C                X OR Y IS DISTINCT FROM Z, THEN IT IS NOT DESTROYED.   DGT3 310
C                                                                       DGT3 320
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DGT3 330
C          NONE                                                         DGT3 340
C                                                                       DGT3 350
C                                                                       DGT3 360
C       METHOD                                                          DGT3 370
C          EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), Z(I) IS THE        DGT3 380
C          DERIVATIVE AT X(I) OF THE LAGRANGIAN INTERPOLATION           DGT3 390
C          POLYNOMIAL OF DEGREE 2 RELEVANT TO THE 3 SUCCESSIVE POINTS   DGT3 400
C          (X(I+K),Y(I+K)) K = -1,0,1. (SEE HILDEBRAND, F.B.,           DGT3 410
C          INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/  DGT3 420
C          TORONTO/LONDON, 1956, PP. 64-68.)                            DGT3 430
C                                                                       DGT3 440
C     ...................................................................DGT3 450
C                                                                       DGT3 460
      SUBROUTINE DGT3(X,Y,Z,NDIM,IER)                                   DGT3 470
C                                                                       DGT3 480
C                                                                       DGT3 490
      DIMENSION X(1),Y(1),Z(1)                                          DGT3 500
C                                                                       DGT3 510
C        TEST OF DIMENSION AND ERROR EXIT IN CASE NDIM IS LESS THAN 3   DGT3 520
      IER=-1                                                            DGT3 530
      IF(NDIM-3)8,1,1                                                   DGT3 540
C                                                                       DGT3 550
C        PREPARE DIFFERENTIATION LOOP                                   DGT3 560
    1 A=X(1)                                                            DGT3 570
      B=Y(1)                                                            DGT3 580
      I=2                                                               DGT3 590
      DY2=X(2)-A                                                        DGT3 600
      IF(DY2)2,9,2                                                      DGT3 610
    2 DY2=(Y(2)-B)/DY2                                                  DGT3 620
C                                                                       DGT3 630
C        START DIFFERENTIATION LOOP                                     DGT3 640
      DO 6 I=3,NDIM                                                     DGT3 650
      A=X(I)-A                                                          DGT3 660
      IF(A)3,9,3                                                        DGT3 670
    3 A=(Y(I)-B)/A                                                      DGT3 680
      B=X(I)-X(I-1)                                                     DGT3 690
      IF(B)4,9,4                                                        DGT3 700
    4 DY1=DY2                                                           DGT3 710
      DY2=(Y(I)-Y(I-1))/B                                               DGT3 720
      DY3=A                                                             DGT3 730
      A=X(I-1)                                                          DGT3 740
      B=Y(I-1)                                                          DGT3 750
      IF(I-3)5,5,6                                                      DGT3 760
    5 Z(1)=DY1+DY3-DY2                                                  DGT3 770
    6 Z(I-1)=DY1+DY2-DY3                                                DGT3 780
C        END DIFFERENTIATION LOOP                                       DGT3 790
C                                                                       DGT3 800
C        NORMAL EXIT                                                    DGT3 810
      IER=0                                                             DGT3 820
      I=NDIM                                                            DGT3 830
    7 Z(I)=DY2+DY3-DY1                                                  DGT3 840
    8 RETURN                                                            DGT3 850
C                                                                       DGT3 860
C        ERROR EXIT IN CASE OF IDENTICAL ARGUMENTS                      DGT3 870
    9 IER=I                                                             DGT3 880
      I=I-1                                                             DGT3 890
      IF(I-2)8,8,7                                                      DGT3 900
      END
```

```
C                                                                       DDGT  10
C     ...................................................................DDGT  20
C                                                                       DDGT  30
C        SUBROUTINE DDGT3                                               DDGT  40
C                                                                       DDGT  50
C        PURPOSE                                                        DDGT  60
C           TO COMPUTE A VECTOR OF DERIVATIVE VALUES GIVEN VECTORS OF   DDGT  70
C           ARGUMENT VALUES AND CORRESPONDING FUNCTION VALUES.          DDGT  80
C                                                                       DDGT  90
C        USAGE                                                          DDGT 100
C           CALL DDGT3(X,Y,Z,NDIM,IER)                                  DDGT 110
C                                                                       DDGT 120
C        DESCRIPTION OF PARAMETERS                                      DDGT 130
C           X      -  GIVEN VECTOR OF DOUBLE PRECISION ARGUMENT VALUES  DDGT 140
C                     (DIMENSION NDIM)                                  DDGT 150
C           Y      -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES  DDGT 160
C                     CORRESPONDING TO X (DIMENSION NDIM)               DDGT 170
C           Z      -  RESULTING VECTOR OF DOUBLE PRECISION DERIVATIVE   DDGT 180
C                     VALUES (DIMENSION NDIM)                           DDGT 190
C           NDIM   -  DIMENSION OF VECTORS X,Y AND Z                    DDGT 200
C           IER    -  RESULTING ERROR PARAMETER                         DDGT 210
C                     IER  = -1  - NDIM IS LESS THAN 3                  DDGT 220
C                     IER  =  0  - NO ERROR                             DDGT 230
C                     IER POSITIVE  - X(IER) = X(IER-1) OR X(IER) =     DDGT 240
C                                     X(IER-2)                          DDGT 250
C                                                                       DDGT 260
C        REMARKS                                                        DDGT 270
C           (1)   IF IER = -1,2,3, THEN THERE IS NO COMPUTATION.        DDGT 280
C           (2)   IF IER =  4,...,N, THEN THE DERIVATIVE VALUES Z(1)    DDGT 290
C                 ...., Z(IER-1) HAVE BEEN COMPUTED.                    DDGT 300
C           (3)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS X OR Y.  IF DDGT 310
C                 X OR Y IS DISTINCT FROM Z, THEN IT IS NOT DESTROYED.  DDGT 320
C                                                                       DDGT 330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DDGT 340
C           NONE                                                        DDGT 350
C                                                                       DDGT 360
C        METHOD                                                         DDGT 370
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), Z(I) IS THE       DDGT 380
C           DERIVATIVE AT X(I) OF THE LAGRANGIAN INTERPOLATION          DDGT 390
C           POLYNOMIAL OF DEGREE 2 RELEVANT TO THE 3 SUCCESSIVE POINTS  DDGT 400
C           (X(I+K),Y(I+K)) K = -1,0,1. (SEE HILDEBRAND, F.B.,          DDGT 410
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ DDGT 420
C           TORONTO/LONDON, 1956, PP. 64-68.)                           DDGT 430
C                                                                       DDGT 440
C     ...................................................................DDGT 450
C                                                                       DDGT 460
      SUBROUTINE DDGT3(X,Y,Z,NDIM,IER)                                  DDGT 470
C                                                                       DDGT 480
C                                                                       DDGT 490
      DIMENSION X(1),Y(1),Z(1)                                          DDGT 500
      DOUBLE PRECISION X,Y,Z,DY1,DY2,DY3,A,B                            DDGT 510
C                                                                       DDGT 520
C        TEST OF DIMENSION AND ERROR EXIT IN CASE NDIM IS LESS THAN 3   DDGT 530
      IER=-1                                                            DDGT 540
      IF(NDIM-3)8,1,1                                                   DDGT 550
C                                                                       DDGT 560
C        PREPARE DIFFERENTIATION LOOP                                   DDGT 570
    1 A=X(1)                                                            DDGT 580
      B=Y(1)                                                            DDGT 590
      I=2                                                               DDGT 600
      DY2=X(2)-A                                                        DDGT 610
      IF(DY2)2,9,2                                                      DDGT 620
    2 DY2=(Y(2)-B)/DY2                                                  DDGT 630
C                                                                       DDGT 640
C        START DIFFERENTIATION LOOP                                     DDGT 650
      DO 6 I=3,NDIM                                                     DDGT 660
      A=X(I)-A                                                          DDGT 670
      IF(A)3,9,3                                                        DDGT 680
    3 A=(Y(I)-B)/A                                                      DDGT 690
      B=X(I)-X(I-1)                                                     DDGT 700
      IF(B)4,9,4                                                        DDGT 710
    4 DY1=DY2                                                           DDGT 720
      DY2=(Y(I)-Y(I-1))/B                                               DDGT 730
      DY3=A                                                             DDGT 740
      A=X(I-1)                                                          DDGT 750
      B=Y(I-1)                                                          DDGT 760
      IF(I-3)5,5,6                                                      DDGT 770
    5 Z(1)=DY1+DY3-DY2                                                  DDGT 780
    6 Z(I-1)=DY1+DY2-DY3                                                DDGT 790
C        END OF DIFFERENTIATION LOOP                                    DDGT 800
C                                                                       DDGT 810
C        NORMAL EXIT                                                    DDGT 820
      IER=0                                                             DDGT 830
      I=NDIM                                                            DDGT 840
    7 Z(I)=DY2+DY3-DY1                                                  DDGT 850
    8 RETURN                                                            DDGT 860
C                                                                       DDGT 870
C        ERROR EXIT IN CASE OF IDENTICAL ARGUMENTS                      DDGT 880
    9 IER=I                                                             DDGT 890
      I=I-1                                                             DDGT 900
      IF(I-2)8,8,7                                                      DDGT 910
      END                                                               DDGT 920
```

Subroutines DET3 and DDET3

These subroutines compute a vector $Z=(z_1,\ldots,z_n)$ of derivative values, given a vector $Y=(y_1,\ldots,y_n)$ of function values whose entries y_i correspond to n equidistantly spaced argument values x_i, with $x_i - x_{i-1} = h$ for $i=2,\ldots,n$. Except at the endpoints x_1 and x_n, z_i is the derivative at x_i of the Lagrangian interpolation polynomial of degree 2 relevant to the three successive points (x_{i-1}, y_{i-1}), (x_i, y_i), and (x_{i+1}, y_{i+1}).

1. Mathematical background

The procedure is that described for subroutines DGT3 and DDGT3, but here we have the additional relation $x_i - x_{i-1} = h$, a constant, for $i=2,\ldots,n$. This leads to the following expression for the z_i:

$$z_i = \begin{cases} \frac{1}{2h}(-y_3+4y_2-3y_1) & \text{if } i=1 \\ \frac{1}{2h}(y_{i+1}-y_{i-1}) & \text{if } i=2,\ldots,n-1 \\ \frac{1}{2h}(3y_n-4y_{n-1}+y_{n-2}) & \text{if } i=n \end{cases} \qquad (1)$$

Assuming that the vector Y represents the function values of a portion of a three-times differentiable function, z_i involves a truncation error T_i where:

$$T_i = \begin{cases} \frac{h^2}{3}y'''(\xi_1),\ \xi_1 \in [x_1, x_3] & \text{if } i=1 \\ \frac{-h^2}{6}y'''(\xi_i),\ \xi_i \in [x_{i-1}, x_{i+1}] & \text{if } i=2,\ldots,n-1 \\ \frac{h^2}{3}y'''(\xi_n),\ \xi_n \in [x_{n-2}, x_n] & \text{if } i=n \end{cases}$$

In addition to these truncation errors, roundoff errors may be of considerable magnitude. Supposing that each of the ordinates y_i can be in error by $\pm\epsilon$, $\epsilon > 0$, the magnitude $|R_i|$ of the corresponding error R_i in the calculation of z_i can be as large as:

$$|R_i| = \begin{cases} \frac{4\epsilon}{|h|} & \text{if } i=1, n \\ \frac{\epsilon}{|h|} & \text{if } i=2,\ldots,n-1 \end{cases}$$

Since small truncation errors generally require small $|h|$, while small roundoff errors generally require large $|h|$, it is reasonable to choose h so that $|T_i| \approx |R_i|$.

If $M = \sup y'''(\xi)$, where $\xi \in [x_1, x_n]$, and if we regard only the inner points $x_2, \ldots, x_{n-1}$, we find that

$$h_{\text{optimum}} \approx \pm 1.8 \sqrt[3]{\epsilon/M},$$

and the magnitude $|E_i|$ of the total possible error E_i in z_i is given by:

$$|E_i| \approx \begin{cases} 3.3 \sqrt[3]{\epsilon^2 M} & \text{if } i=1, n \\ 1.1 \sqrt[3]{\epsilon^2 M} & \text{if } i=2, \ldots, n-1 \end{cases}$$

2. Programming considerations

The subroutines compute the z_i in serial order according to (1). If $n < 3$ or if $h = 0$, there is no computation, and the error parameter IER is set to -1 or 1 respectively. Otherwise IER is set to zero at the end of the computation. The output vector Z can have the same storage allocation as Y.

For reference see F. B. Hildebrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 82-84.

```
C                                                                       DET3  10
C     ..................................................................DET3  20
C                                                                       DET3  30
C        SUBROUTINE DET3                                                DET3  40
C                                                                       DET3  50
C        PURPOSE                                                        DET3  60
C           TO COMPUTE A VECTOR OF DERIVATIVE VALUES GIVEN A VECTOR OF  DET3  70
C           FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO EQUIDISTANTLY   DET3  80
C           SPACED ARGUMENT VALUES.                                     DET3  90
C                                                                       DET3 100
C        USAGE                                                          DET3 110
C           CALL DET3(H,Y,Z,NDIM,IER)                                   DET3 120
C                                                                       DET3 130
C        DESCRIPTION OF PARAMETERS                                      DET3 140
C           H      -  CONSTANT DIFFERENCE BETWEEN SUCCESSIVE ARGUMENT   DET3 150
C                     VALUES (H IS POSITIVE IF THE ARGUMENT VALUES      DET3 160
C                     INCREASE AND NEGATIVE OTHERWISE)                  DET3 170
C           Y      -  GIVEN VECTOR OF FUNCTION VALUES (DIMENSION NDIM)  DET3 180
C           Z      -  RESULTING VECTOR OF DERIVATIVE VALUES (DIMENSION  DET3 190
C                     NDIM)                                             DET3 200
C           NDIM   -  DIMENSION OF VECTORS Y AND Z                      DET3 210
C           IER    -  RESULTING ERROR PARAMETER                         DET3 220
C                     IER = -1  - NDIM IS LESS THAN 3                   DET3 230
C                     IER =  0  - NO ERROR                              DET3 240
C                     IER =  1  - H = 0                                 DET3 250
C                                                                       DET3 260
C        REMARKS                                                        DET3 270
C           (1)    IF IER = -1,1, THEN THERE IS NO COMPUTATION.         DET3 280
C           (2)    Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y. IF Y IS DET3 290
C                  DISTINCT FROM Z, THEN IT IS NOT DESTROYED.           DET3 300
C                                                                       DET3 310
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DET3 320
C           NONE                                                        DET3 330
C                                                                       DET3 340
C        METHOD                                                         DET3 350
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    DET3 360
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), Z(I) IS THE       DET3 370
C           DERIVATIVE AT X(I) OF THE LAGRANGIAN INTERPOLATION          DET3 380
C           POLYNOMIAL OF DEGREE 2 RELEVANT TO THE 3 SUCCESSIVE POINTS  DET3 390
C           (X(I+K),Y(I+K)) K = -1,0,1.  (SEE HILDEBRAND, F.B.,         DET3 400
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC-GRAW-HILL, NEW YORK/ DET3 410
C           TORONTO/LONDON, 1956, PP.82-84.)                            DET3 420
C                                                                       DET3 430
C     ..................................................................DET3 440
C                                                                       DET3 450
      SUBROUTINE DET3(H,Y,Z,NDIM,IER)                                   DET3 460
C                                                                       DET3 470
C                                                                       DET3 480
      DIMENSION Y(1),Z(1)                                               DET3 490
C                                                                       DET3 500
C        TEST OF DIMENSION                                              DET3 510
      IF(NDIM-3)4,1,1                                                   DET3 520
C                                                                       DET3 530
C        TEST OF STEPSIZE                                               DET3 540
    1 IF(H)2,5,2                                                        DET3 550
C                                                                       DET3 560
C        PREPARE DIFFERENTIATION LOOP                                   DET3 570
    2 HH=.5/H                                                           DET3 580
      YY=Y(NDIM-2)                                                      DET3 590
      B=Y(2)+Y(2)                                                       DET3 600
      B=HH*(B+B-Y(3)-Y(1)-Y(1)-Y(1))                                    DET3 610
C                                                                       DET3 620
C        START DIFFERENTIATION LOOP                                     DET3 630
      DO 3 I=3,NDIM                                                     DET3 640
      A=B                                                               DET3 650
      B=HH*(Y(I)-Y(I-2))                                                DET3 660
    3 Z(I-2)=A                                                          DET3 670
C        END OF DIFFERENTIATION LOOP                                    DET3 680
C                                                                       DET3 690
C        NORMAL EXIT                                                    DET3 700
      IER=0                                                             DET3 710
      A=Y(NDIM-1)+Y(NDIM-1)                                             DET3 720
      Z(NDIM)=HH*(Y(NDIM)+Y(NDIM)+Y(NDIM)-A-A+YY)                       DET3 730
      Z(NDIM-1)=B                                                       DET3 740
      RETURN                                                            DET3 750
C                                                                       DET3 760
C        ERROR EXIT IN CASE NDIM IS LESS THAN 3                         DET3 770
    4 IER=-1                                                            DET3 780
      RETURN                                                            DET3 790
C                                                                       DET3 800
C        ERROR EXIT IN CASE OF ZERO STEPSIZE                            DET3 810
    5 IER=1                                                             DET3 820
      RETURN                                                            DET3 830
      END                                                               DET3 840
```

```
C                                                                       DDT3  10
C     ..................................................................DDT3  20
C                                                                       DDT3  30
C        SUBROUTINE DDET3                                               DDT3  40
C                                                                       DDT3  50
C        PURPOSE                                                        DDT3  60
C           TO COMPUTE A VECTOR OF DERIVATIVE VALUES GIVEN A VECTOR OF  DDT3  70
C           FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO EQUIDISTANTLY   DDT3  80
C           SPACED ARGUMENT VALUES.                                     DDT3  90
C                                                                       DDT3 100
C        USAGE                                                          DDT3 110
C           CALL DDET3(H,Y,Z,NDIM,IER)                                  DDT3 120
C                                                                       DDT3 130
C        DESCRIPTION OF PARAMETERS                                      DDT3 140
C           H      -  DOUBLE PRECISION CONSTANT DIFFERENCE BETWEEN      DDT3 150
C                     SUCCESSIVE ARGUMENT VALUES (H IS POSITIVE IF THE  DDT3 160
C                     ARGUMENT VALUES INCREASE AND NEGATIVE OTHERWISE)  DDT3 170
C           Y      -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES  DDT3 180
C                     (DIMENSION NDIM)                                  DDT3 190
C           Z      -  RESULTING VECTOR OF DOUBLE PRECISION DERIVATIVE   DDT3 200
C                     VALUES (DIMENSION NDIM)                           DDT3 210
C           NDIM   -  DIMENSION OF VECTORS Y AND Z                      DDT3 220
C           IER    -  RESULTING ERROR PARAMETER                         DDT3 230
C                     IER = -1  - NDIM IS LESS THAN 3                   DDT3 240
C                     IER =  0  - NO ERROR                              DDT3 250
C                     IER =  1  - H = 0                                 DDT3 260
C                                                                       DDT3 270
C        REMARKS                                                        DDT3 280
C           (1)    IF IER = -1,1, THEN THERE IS NO COMPUTATION.         DDT3 290
C           (2)    Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y. IF Y IS DDT3 300
C                  DISTINCT FROM Z, THEN IT IS NOT DESTROYED.           DDT3 310
C                                                                       DDT3 320
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DDT3 330
C           NONE                                                        DDT3 340
C                                                                       DDT3 350
C        METHOD                                                         DDT3 360
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    DDT3 370
C           EXCEPT AT THE ENDPOINTS X(1) AND X(NDIM), Z(I) IS THE       DDT3 380
C           DERIVATIVE AT X(I) OF THE LAGRANGIAN INTERPOLATION          DDT3 390
C           POLYNOMIAL OF DEGREE 2 RELEVANT TO THE 3 SUCCESSIVE POINTS  DDT3 400
C           (X(I+K),Y(I+K)) K = -1,0,1.  (SEE HILDEBRAND, F.B.,         DDT3 410
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC-GRAW-HILL, NEW YORK/ DDT3 420
C           TORONTO/LONDON, 1956, PP.82-84.)                            DDT3 430
C                                                                       DDT3 440
C     ..................................................................DDT3 450
C                                                                       DDT3 460
      SUBROUTINE DDET3(H,Y,Z,NDIM,IER)                                  DDT3 470
C                                                                       DDT3 480
C                                                                       DDT3 490
      DIMENSION Y(1),Z(1)                                               DDT3 500
      DOUBLE PRECISION H,Y,Z,HH,YY,A,B                                  DDT3 510
C                                                                       DDT3 520
C        TEST OF DIMENSION                                              DDT3 530
      IF(NDIM-3)4,1,1                                                   DDT3 540
C                                                                       DDT3 550
C        TEST OF STEPSIZE                                               DDT3 560
    1 IF(H)2,5,2                                                        DDT3 570
C                                                                       DDT3 580
C        PREPARE DIFFERENTIATION LOOP                                   DDT3 590
    2 HH=.5D0/H                                                         DDT3 600
      YY=Y(NDIM-2)                                                      DDT3 610
      B=Y(2)+Y(2)                                                       DDT3 620
      B=HH*(B+B-Y(3)-Y(1)-Y(1)-Y(1))                                    DDT3 630
C                                                                       DDT3 640
C        START DIFFERENTIATION LOOP                                     DDT3 650
      DO 3 I=3,NDIM                                                     DDT3 660
      A=B                                                               DDT3 670
      B=HH*(Y(I)-Y(I-2))                                                DDT3 680
    3 Z(I-2)=A                                                          DDT3 690
C        END OF DIFFERENTIATION LOOP                                    DDT3 700
C                                                                       DDT3 710
C        NORMAL EXIT                                                    DDT3 720
      IER=0                                                             DDT3 730
      A=Y(NDIM-1)+Y(NDIM-1)                                             DDT3 740
      Z(NDIM)=HH*(Y(NDIM)+Y(NDIM)+Y(NDIM)-A-A+YY)                       DDT3 750
      Z(NDIM-1)=B                                                       DDT3 760
      RETURN                                                            DDT3 770
C                                                                       DDT3 780
C        ERROR EXIT IN CASE NDIM IS LESS THAN 3                         DDT3 790
    4 IER=-1                                                            DDT3 800
      RETURN                                                            DDT3 810
C                                                                       DDT3 820
C        ERROR EXIT IN CASE OF ZERO STEPSIZE                            DDT3 830
    5 IER=1                                                             DDT3 840
      RETURN                                                            DDT3 850
      END                                                               DDT3 860
```

Subroutines DET5 and DDET5

These subroutines compute a vector $Z=(z_1,\ldots,z_n)$ of derivative values, given a vector $Y=(y_1,\ldots,y_n)$ of function values whose entries y_i correspond to n equidistantly spaced argument values x_i, with $x_i - x_{i-1} = h$ for $i=2,\ldots,n$. Except at the points x_1, x_2, x_{n-1}, and x_n, z_i is the derivative at x_i of the Lagrangian interpolation polynomial of degree 4 relevant to the five successive points (x_{i+k}, y_{i+k}), $k=-2,-1,\ldots,2$.

1. Mathematical background

For $i=1,\ldots,n-4$ we must find a_i, b_i, c_i, d_i, and e_i such that

$$\bar{y}_i(x) = a_i x^4 + b_i x^3 + c_i x^2 + d_i x + e_i$$

passes through (x_{i+k}, y_{i+k}) for $k=0,\ldots,4$.

The desired derivative values z_i are given by:

$$z_i = \begin{cases} \bar{y}_1'(x_i) = 4a_1x_i^3 + 3b_1x_i^2 + 2c_1x_i + d_1 & \text{if } i=1,2 \\ \bar{y}_{i-2}'(x_i) = 4a_{i-2}x_i^3 + 3b_{i-2}x_i^2 + 2c_{i-2}x_i + d_{i-2} & \text{if } i=3,\ldots,n-2 \\ \bar{y}_{n-4}'(x_i) = 4a_{n-4}x_i^3 + 3b_{n-4}x_i^2 + 2c_{n-4}x_i + d_{n-4} & \text{if } i=n-1,n \end{cases}$$

Using the fact that $x_i - x_{i-1} = h$, a constant, for $i=2,\ldots,n$, we get:

$$z_i = \begin{cases} \frac{1}{12h}(-25y_1 + 48y_2 - 36y_3 + 16y_4 - 3y_5) & \text{if } i=1 \\ \frac{1}{12h}(-3y_1 - 10y_2 + 18y_3 - 6y_4 + y_5) & \text{if } i=2 \\ \frac{1}{12h}(y_{i-2} - 8y_{i-1} + 8y_{i+1} - y_{i+2}) & \text{if } i=3,\ldots,n-2 \\ \frac{1}{12h}(-y_{n-4} + 6y_{n-3} - 18y_{n-2} + 10_{n-1} + 3y_n) & \text{if } i=n-1 \\ \frac{1}{12h}(3y_{n-4} - 16y_{n-3} + 36y_{n-2} - 48y_{n-1} + 25y_n) & \text{if } i=n \end{cases} \quad (1)$$

Assuming that the vector Y represents the function values of a portion of a five-times differentiable function, z_i involves a truncation error T_i where:

$$T_i = \begin{cases} \frac{h^4}{5}y^{v}(\xi_1), & \xi_1 \epsilon [x_1, x_5] \quad \text{if } i=1 \\ -\frac{h^4}{20}y^{v}(\xi_2), & \xi_2 \epsilon [x_1, x_5] \quad \text{if } i=2 \\ \frac{h^4}{30}y^{v}(\xi_i), & \xi_i \epsilon [x_{i-2}, x_{i+2}] \quad \text{if } i=3,\ldots,n-2 \\ -\frac{h^4}{20}y^{v}(\xi_{n-1}), & \xi_{n-i} \epsilon [x_{n-4}, x_n] \quad \text{if } i=n-1 \\ \frac{h^4}{5}y^{v}(\xi_n), & \xi_n \epsilon [x_{n-4}, x_n] \quad \text{if } i=n \end{cases}$$

In addition to the truncation errors, roundoff errors may be of considerable magnitude. Supposing that the ordinates y_i can be in error by $\pm\epsilon$, $\epsilon > 0$, the magnitude $|R_i|$ of the corresponding error R_i in the computation of z_i can be as large as:

$$|R_i| = \begin{cases} \frac{32\epsilon}{3|h|} & \text{if } i=1,n \\ \frac{19\epsilon}{6|h|} & \text{if } i=2,n-1 \\ \frac{3\epsilon}{2|h|} & \text{if } i=3,\ldots,n-2 \end{cases}$$

Since small truncation errors generally require small $|h|$, while small roundoff errors generally require large $|h|$, it is reasonable to choose h so that $|T_i| \approx |R_i|$.

If $M = \sup_{\xi \in [x_1, x_n]} y^{V}(\xi)$, and if we regard only the inner points $x_3, \ldots, x_{n-2}$, we find that

$$h_{optimum} \approx 2.1 \sqrt[5]{\epsilon/M}$$

and the magnitude $|E_i|$ of the total possible error E_i in z_i is given by:

$$|E_i| \approx \begin{cases} 9 \sqrt[5]{\epsilon^4 M} & \text{if } i=1, n \\ 2.5 \sqrt[5]{\epsilon^4 M} & \text{if } i=2, n-1 \\ 1.4 \sqrt[5]{\epsilon^4 M} & \text{if } i=3, \ldots, n-2 \end{cases}$$

2. Programming considerations

The subroutines compute the z_i in serial order according to (1). If $n < 5$ or if $h = 0$, there is no computation, and the error parameter IER is set to -1 or 1, respectively. Otherwise IER is set to zero at the end of the computation. The output vector Z may have the same storage allocation as Y.

For reference see F.B. Hilderbrand, Introduction to Numerical Analysis. McGraw-Hill, New York/Toronto/London, 1956, pp. 82-84.

```
C                                                                       DET5  10
C     ..................................................................DET5  20
C                                                                       DET5  30
C        SUBROUTINE DET5                                                DET5  40
C                                                                       DET5  50
C        PURPOSE                                                        DET5  60
C           TO COMPUTE A VECTOR OF DERIVATIVE VALUES GIVEN A VECTOR OF  DET5  70
C           FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO EQUIDISTANTLY   DET5  80
C           SPACED ARGUMENT VALUES.                                     DET5  90
C                                                                       DET5 100
C        USAGE                                                          DET5 110
C           CALL DET5(H,Y,Z,NDIM,IER)                                   DET5 120
C                                                                       DET5 130
C        DESCRIPTION OF PARAMETERS                                      DET5 140
C           H      -  CONSTANT DIFFERENCE BETWEEN SUCCESSIVE ARGUMENT   DET5 150
C                     VALUES (H IS POSITIVE IF THE ARGUMENT VALUES      DET5 160
C                     INCREASE AND NEGATIVE OTHERWISE)                  DET5 170
C           Y      -  GIVEN VECTOR OF FUNCTION VALUES (DIMENSION NDIM)  DET5 180
C           Z      -  RESULTING VECTOR OF DERIVATIVE VALUES (DIMENSION  DET5 190
C                     NDIM)                                             DET5 200
C           NDIM   -  DIMENSION OF VECTORS Y AND Z                      DET5 210
C           IER    -  RESULTING ERROR PARAMETER                         DET5 220
C                     IER = -1  - NDIM IS LESS THAN 5                   DET5 230
C                     IER =  0  - NO ERROR                              DET5 240
C                     IER =  1  - H = 0                                 DET5 250
C                                                                       DET5 260
C        REMARKS                                                        DET5 270
C           (1)   IF IER = -1,1, THEN THERE IS NO COMPUTATION.          DET5 280
C           (2)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y IS DET5 290
C                 DISTINCT FROM Z, THEN IT IS NOT DESTROYED.            DET5 300
C                                                                       DET5 310
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DET5 320
C           NONE                                                        DET5 330
C                                                                       DET5 340
C        METHOD                                                         DET5 350
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    DET5 360
C           EXCEPT AT THE POINTS X(1),X(2),X(NDIM-1) AND X(NDIM), Z(I)  DET5 370
C           IS THE DERIVATIVE AT X(I) OF THE LAGRANGIAN INTERPOLATION   DET5 380
C           POLYNOMIAL OF DEGREE 4 RELEVANT TO THE 5 SUCCESSIVE POINTS  DET5 390
C           (X(I+K),Y(I+K)) K = -2,-1,...,2.  (SEE HILDEBRAND, F.B.,    DET5 400
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ DET5 410
C           TORONTO/LONDON, 1956, PP. 82-84.)                           DET5 420
C                                                                       DET5 430
C     ..................................................................DET5 440
C                                                                       DET5 450
      SUBROUTINE DET5(H,Y,Z,NDIM,IER)                                   DET5 460
C                                                                       DET5 470
C                                                                       DET5 480
      DIMENSION Y(1),Z(1)                                               DET5 490
C                                                                       DET5 500
C        TEST OF DIMENSION                                              DET5 510
      IF(NDIM-5)4,1,1                                                   DET5 520
C                                                                       DET5 530
C        TEST OF STEPSIZE                                               DET5 540
    1 IF(H)2,5,2                                                        DET5 550
C                                                                       DET5 560
C                                                                       DET5 570
C        PREPARE DIFFERENTIATION LOOP                                   DET5 580
    2 HH=.08333333/H                                                    DET5 590
      YY=Y(NDIM-4)                                                      DET5 600
      B=HH*(-25.*Y(1)+48.*Y(2)-36.*Y(3)+16.*Y(4)-3.*Y(5))               DET5 610
      C=HH*(-3.*Y(1)-10.*Y(2)+18.*Y(3)-6.*Y(4)+Y(5))                    DET5 620
C                                                                       DET5 630
C        START DIFFERENTIATION LOOP                                     DET5 640
      DO 3 I=5,NDIM                                                     DET5 650
      A=B                                                               DET5 660
      B=C                                                               DET5 670
      C=HH*(Y(I-4)-Y(I)+8.*(Y(I-1)-Y(I-3)))                             DET5 680
    3 Z(I-4)=A                                                          DET5 690
C        END OF DIFFERENTIATION LOOP                                    DET5 700
C                                                                       DET5 710
C        NORMAL EXIT                                                    DET5 720
      IER=0                                                             DET5 730
      A=HH*(-YY+6.*Y(NDIM-3)-18.*Y(NDIM-2)+10.*Y(NDIM-1)+3.*Y(NDIM))    DET5 740
     OZ(NDIM)=HH*(3.*YY-16.*Y(NDIM-3)+36.*Y(NDIM-2)-48.*Y(NDIM-1)       DET5 750
     1            +25.*Y(NDIM))                                         DET5 760
      Z(NDIM-1)=A                                                       DET5 770
      Z(NDIM-2)=C                                                       DET5 780
      Z(NDIM-3)=B                                                       DET5 790
      RETURN                                                            DET5 800
C                                                                       DET5 810
C        ERROR EXIT IN CASE NDIM IS LESS THAN 5                         DET5 820
    4 IER=-1                                                            DET5 830
      RETURN                                                            DET5 840
C                                                                       DET5 850
C        ERROR EXIT IN CASE OF ZERO STEPSIZE                            DET5 860
    5 IER=1                                                             DET5 870
      RETURN                                                            DET5 880
      END
```

```
C                                                                       DDT5  10
C     ..................................................................DDT5  20
C                                                                       DDT5  30
C        SUBROUTINE DDET5                                               DDT5  40
C                                                                       DDT5  50
C        PURPOSE                                                        DDT5  60
C           TO COMPUTE A VECTOR OF DERIVATIVE VALUES GIVEN A VECTOR OF  DDT5  70
C           FUNCTION VALUES WHOSE ENTRIES CORRESPOND TO EQUIDISTANTLY   DDT5  80
C           SPACED ARGUMENT VALUES.                                     DDT5  90
C                                                                       DDT5 100
C        USAGE                                                          DDT5 110
C           CALL DDET5(H,Y,Z,NDIM,IER)                                  DDT5 120
C                                                                       DDT5 130
C        DESCRIPTION OF PARAMETERS                                      DDT5 140
C           H      -  DOUBLE PRECISION CONSTANT DIFFERENCE BETWEEN      DDT5 150
C                     SUCCESSIVE ARGUMENT VALUES (H IS POSITIVE IF THE  DDT5 160
C                     ARGUMENT VALUES INCREASE AND NEGATIVE OTHERWISE)  DDT5 170
C           Y      -  GIVEN VECTOR OF DOUBLE PRECISION FUNCTION VALUES  DDT5 180
C                     (DIMENSION NDIM)                                  DDT5 190
C           Z      -  RESULTING VECTOR OF DOUBLE PRECISION DERIVATIVE   DDT5 200
C                     VALUES (DIMENSION NDIM)                           DDT5 210
C           NDIM   -  DIMENSION OF VECTORS Y AND Z                      DDT5 220
C           IER    -  RESULTING ERROR PARAMETER                         DDT5 230
C                     IER = -1  - NDIM IS LESS THAN 5                   DDT5 240
C                     IER =  0  - NO ERROR                              DDT5 250
C                     IER =  1  - H = 0                                 DDT5 260
C                                                                       DDT5 270
C        REMARKS                                                        DDT5 280
C           (1)   IF IER = -1,1, THEN THERE IS NO COMPUTATION.          DDT5 290
C           (2)   Z CAN HAVE THE SAME STORAGE ALLOCATION AS Y.  IF Y IS DDT5 300
C                 DISTINCT FROM Z, THEN IT IS NOT DESTROYED.            DDT5 310
C                                                                       DDT5 320
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DDT5 330
C           NONE                                                        DDT5 340
C                                                                       DDT5 350
C        METHOD                                                         DDT5 360
C           IF X IS THE (SUPPRESSED) VECTOR OF ARGUMENT VALUES, THEN    DDT5 370
C           EXCEPT AT THE POINTS X(1),X(2),X(NDIM-1) AND X(NDIM), Z(I)  DDT5 380
C           IS THE DERIVATIVE AT X(I) OF THE LAGRANGIAN INTERPOLATION   DDT5 390
C           POLYNOMIAL OF DEGREE 4 RELEVANT TO THE 5 SUCCESSIVE POINTS  DDT5 400
C           (X(I+K),Y(I+K)) K = -2,-1,...,2.  (SEE HILDEBRAND, F.B.,    DDT5 410
C           INTRODUCTION TO NUMERICAL ANALYSIS, MC GRAW-HILL, NEW YORK/ DDT5 420
C           TORONTO/LONDON, 1956, PP. 82-84.)                           DDT5 430
C                                                                       DDT5 440
C     ..................................................................DDT5 450
C                                                                       DDT5 460
      SUBROUTINE DDET5(H,Y,Z,NDIM,IER)                                  DDT5 470
C                                                                       DDT5 480
C                                                                       DDT5 490
      DIMENSION Y(1),Z(1)                                               DDT5 500
      DOUBLE PRECISION H,Y,Z,HH,YY,A,B,C                                DDT5 510
C                                                                       DDT5 520
C        TEST OF DIMENSION                                              DDT5 530
      IF(NDIM-5)4,1,1                                                   DDT5 540
C                                                                       DDT5 550
C        TEST OF STEPSIZE                                               DDT5 560
    1 IF(H)2,5,2                                                        DDT5 570
C                                                                       DDT5 580
C        PREPARE DIFFERENTIATION LOOP                                   DDT5 590
    2 HH=.083333333333333D0/H                                           DDT5 600
      YY=Y(NDIM-4)                                                      DDT5 610
      B=HH*(-25.D0*Y(1)+48.D0*Y(2)-36.D0*Y(3)+16.D0*Y(4)-3.D0*Y(5))     DDT5 620
      C=HH*(-3.D0*Y(1)-10.D0*Y(2)+18.D0*Y(3)-6.D0*Y(4)+Y(5))            DDT5 630
C                                                                       DDT5 640
C        START DIFFERENTIATION LOOP                                     DDT5 650
      DO 3 I=5,NDIM                                                     DDT5 660
      A=B                                                               DDT5 670
      B=C                                                               DDT5 680
      C=HH*(Y(I-4)-Y(I)+8.D0*(Y(I-1)-Y(I-3)))                           DDT5 690
    3 Z(I-4)=A                                                          DDT5 700
C        END OF DIFFERENTIATION LOOP                                    DDT5 710
C                                                                       DDT5 720
C        NORMAL EXIT                                                    DDT5 730
      IER=0                                                             DDT5 740
      OA=HH*(-YY+6.D0*Y(NDIM-3)-18.D0*Y(NDIM-2)+10.D0*Y(NDIM-1)         DDT5 750
     1      +3.D0*Y(NDIM))                                              DDT5 760
      OZ(NDIM)=HH*(3.D0*YY-16.D0*Y(NDIM-3)+36.D0*Y(NDIM-2)              DDT5 770
     1            -48.D0*Y(NDIM-1)+25.D0*Y(NDIM))                       DDT5 780
      Z(NDIM-1)=A                                                       DDT5 790
      Z(NDIM-2)=C                                                       DDT5 800
      Z(NDIM-3)=B                                                       DDT5 810
      RETURN                                                            DDT5 820
C                                                                       DDT5 830
C        ERROR EXIT IN CASE NDIM IS LESS THAN 5                         DDT5 840
    4 IER=-1                                                            DDT5 850
      RETURN                                                            DDT5 860
C                                                                       DDT5 870
C        ERROR EXIT IN CASE OF ZERO STEPSIZE                            DDT5 880
    5 IER=1                                                             DDT5 890
      RETURN                                                            DDT5 900
      END                                                               DDT5 910
```

Subroutines DCAR and DDCAR

Suppose that $y = y(t)$ is eleven-times differentiable in a neighborhood of radius $R > 0$ of the point x. These subroutines compute an approximation Z to $y'(x)$ by applying Richardson's and Romberg's extrapolation method to successively computed central divided differences, using function values in a closed interval of radius $|h|$ about the point x, $0 < |h| < R$.

1. Mathematical background

Suppose, first, that $y = y(t)$ is analytic at x; that is, y has a Taylor series expansion about the point x with radius of convergence $R > 0$. Let h be such that $0 < |h| < R$. For each positive integer n a step size h_1 with $0 < h_1 \le |h|$ is computed as described below, and a sequence h_k of increments is generated, where

$$h_k = \frac{n-k+1}{n} h_1 \qquad \text{for } k=2,\ldots,n.$$

From the sequence $(x-h_k,\ x+h_k)$ of point pairs $(k=1,\ldots,n)$, the sequence of central divided differences

$$T_{0,k} = \frac{y(x+h_k) - y(x-h_k)}{2h_k} \qquad \text{for } k=1,\ldots,n \qquad (1)$$

is computed, which forms the first column of the triangular Romberg scheme. The central divided differences $T_{0,k}$ represent the slopes of the secants s_k in Figure 26.

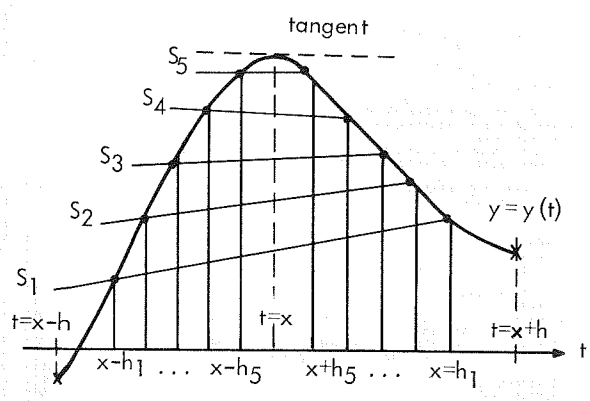


Figure 26. A sequence of secants for a given function $y = y(t)$ and a given argument $t = x$ for the case $n = 5$, $h > 0$

From the Taylor series expansions of $y(x+h_k)$ and $y(x-h_k)$ it follows that

$$T_{0,k} = y'(x) + \frac{h_k^2}{3!} y'''(x) + \frac{h_k^4}{5!} y^{v}(x) + \ldots \text{ for } k=1,\ldots,n$$

so that, as an approximation to $y'(x)$, $T_{0,k}$ involves a truncation error of order h_k^2.

Knowing the two divided differences $T_{0,k}$ and $T_{0,k+1}$, we are able to generate the extrapolated value

$$T_{1,k} = T_{0,k+1} + \frac{T_{0,k+1} - T_{0,k}}{a_{1,k} - 1}$$

where (2)

$$a_{1,k} = \left(1 + \frac{1}{n-k}\right)^2$$

$T_{1,k}$ is a better approximation to $y'(x)$ since

$$T_{1,k} = y'(x) - \frac{1}{5!}\frac{1}{a_{1,k}} h_k^4 y^{v}(x) - \frac{1}{7!}\frac{1}{a_{1,k}}\left(1+\frac{1}{a_{1,k}}\right) h_k^6 y^{vii}(x) - \ldots,$$

which involves a truncation error of order h_k^4.

If we also know $T_{0,k+2}$, then we can generate $T_{1,k+1}$ using formula (2), and further, we can compute the extrapolated value

$$T_{2,k} = T_{1,k+1} + \frac{T_{1,k+1} - T_{1,k}}{a_{2,k} - 1}$$

where (3)

$$a_{2,k} = \left(1 + \frac{2}{n-(k+1)}\right)^2$$

which involves a truncation error of order h_k^6.

Generally, the order of the truncation error is increased by 2 with each new extrapolation step; in particular, $T_{i,j}$ will involve a truncation error of order

$$h_j^{2i+2}, \quad i = 0,\ldots,n-1, \quad j = 1,\ldots,n.$$

Figure 27 shows the arrangement of the T-values in the triangular Romberg scheme. The T-values are computed following the upward diagonals, using the general formula:

$$T_{m,k-m} = T_{m-1,k-m+1} + \frac{T_{m-1,k-m+1} - T_{m-1,k-m}}{\frac{m}{n-(k+1)}\left(2+\frac{m}{n-(k+1)}\right)}$$

$$\text{for } m = 1, \ldots, k-1 \text{ for fixed } k,\ k = 2, \ldots, n \tag{4}$$

Truncation error		$O(h_k^2)$	$O(h_k^4)$	$O(h_k^6)$	$O(h_k^8)$	$O(h_k^{10})$
Stepsize	k \ m	0	1	2	3	4
h_1	1	$T_{0,1}$	$T_{1,1}$	$T_{2,1}$	$T_{3,1}$	$T_{4,1}$
$h_2=0.8h_1$	2	$T_{0,2}$	$T_{1,2}$	$T_{2,2}$	$T_{3,2}$	
$h_2=0.6h_1$	3	$T_{0,3}$	$T_{1,3}$	$T_{2,3}$		
$h_4=0.4h_1$	4	$T_{0,4}$	$T_{1,4}$			
$h_5=0.2h_1$	5	$T_{0,5}$				

Figure 27. The triangular Romberg scheme of T-values for the case n = 5

Numerical experience shows that the accuracy of the results depends heavily on roundoff errors in the central divided differences $T_{0,k}$. Therefore, the choice of the absolutely smallest step size, h_n, is based on the following considerations.
Let

$$v = \begin{cases} 1 & \text{in single-precision computation} \\ 3 & \text{in double-precision computation} \end{cases},$$

$$h_0 = \min\left(n \cdot 10^{-v}, |h|\right)$$

Set

$$Y = \frac{1}{2}\left(y(x+h_0) + y(x-h_0)\right)$$

and

$$T = \frac{1}{2}\left(y(x+h_0) - y(x-h_0)\right)/h_0$$

Y and T are approximations to y(x) and y'(x), respectively.

Assuming that the errors in the function values y(t) for $t \in [x-|h|, x+|h|]$ are bounded by

$$\epsilon = \begin{cases} Y \cdot 10^{-D} & \text{if } |Y| > 1 \\ 10^{-D} & \text{if } |Y| \le 1 \end{cases}$$

formula (1) shows that the roundoff error in the computation of $T_{0,n}$ is bounded by

$$R(T_{0,n}) = \frac{\epsilon}{h_n} = \begin{cases} \dfrac{Y \cdot 10^{-D}}{h_n} & \text{if } |Y| > 1 \\ \dfrac{10^{-D}}{h_n} & \text{if } |Y| \le 1 \end{cases}$$

where D is the number of significant digits in the floating-point representation of numbers. Suppose, also, that we are willing to tolerate a roundoff error

$$R'(T_{0,n}) = \begin{cases} T \cdot 10^{-D+v} & \text{if } |T| > 1 \\ 10^{-D+v} & \text{if } |T| \le 1 \end{cases}$$

Then we must have $R(T_{0,n}) \le R'(T_{0,n})$, which is satisfied when

$$h_n = \frac{\max(1, |Y|)}{\max(1, |T|)} \cdot 10^{-v} \tag{5}$$

Finally we set

$$h_1 = \min(n \cdot h_n, |h|) \tag{6}$$

guaranteeing that the evaluation of the function y = y(t) is restricted to the closed interval $[x-|h|, x+|h|]$.

2. Programming considerations

Numerical experience shows that, because of increasing roundoff errors, it is generally fruitless to perform more than five extrapolations. Thus, the subroutines use n = 5, and consequently it is only necessary that y = y(t) be eleven-times differentiable, rather than analytic. It is easy to see that in the case n = 5, y = y(t) must be evaluated at twelve points in the closed interval $[x-|h|, x+|h|]$.

The parameter lists of both subroutines contain the decision parameter IH. If the user wants the maximum step size to equal $|h|$, he must set IH = 0. In all other cases (IH $\neq$ 0) the procedure determines the maximum step size by itself, using formulas (5) and (6) with n = 5. Step size h_1 and all the other step sizes h_k (k = 2, . . . , 5) are successively stored in the single storage location HH.

As previously explained, the computation of the T-values is performed along the upward diagonals of the triangular Romberg scheme. Therefore, only a one-dimensional internal storage vector,

named AUX, with five storage locations is necessary. Figure 28 shows the storage administration and the sequence of computations (numbers in parentheses).

AUX(1)	$T_{0,5}$(11)				
AUX(2)	$T_{0,4}$(7)	$T_{1,4}$(12)			
AUX(3)	$T_{0,3}$(4)	$T_{1,3}$(8)	$T_{2,3}$(13)		
AUX(4)	$T_{0,2}$(2)	$T_{1,2}$(5)	$T_{2,2}$(9)	$T_{3,2}$(14)	
AUX(5)	$T_{0,1}$(1)	$T_{1,1}$(3)	$T_{2,1}$(6)	$T_{3,1}$(10)	$T_{4,1}$(15)

Figure 28. Storage administration and order of computation

Each extrapolation loop, the computation of the elements on an upward diagonal, is terminated as soon as the absolute values of the differences between adjacent diagonal elements stop decreasing, showing the influence of roundoff errors. The computed T-value that differs least in absolute value from its immediately preceding diagonal neighbor is the desired value Z.

3. Accuracy of results

Tests of the single-precision (double-precision) subroutine on the functions $y(t) = t^2$, $1/t$, e^t, $\tan t$, and $\sin t$ showed that the accuracy obtained using the step size determined by the subroutine was 3-4 (11-12) significant digits near poles, 5-6(13-14) decimal places near points with horizontal tangents, and 5-6(13-14) significant digits otherwise.

For reference see S. Fillipi and H. Engels, "Altes und Neues zur numerischen Differentiation," Elektronische Datenverarbeitung, iss. 2(1966) pp. 57-65.

```
C                                                                       DCAR  10
C     ...................................................................DCAR  20
C                                                                       DCAR  30
C        SUBROUTINE DCAR                                                DCAR  40
C                                                                       DCAR  50
C     PURPOSE                                                           DCAR  60
C        TO COMPUTE, AT A GIVEN POINT X, AN APPROXIMATION Z TO THE      DCAR  70
C        DERIVATIVE OF AN ANALYTICALLY GIVEN FUNCTION FCT THAT IS 11-   DCAR  80
C        TIMES DIFFERENTIABLE IN A DOMAIN CONTAINING A CLOSED, 2-SIDED  DCAR  90
C        SYMMETRIC INTERVAL OF RADIUS ABSOLUTE H ABOUT X, USING FUNCTIONDCAR 100
C        VALUES ONLY ON THAT CLOSED INTERVAL.                           DCAR 110
C                                                                       DCAR 120
C     USAGE                                                             DCAR 130
C           CALL DCAR (X,H,IH,FCT,Z)                                    DCAR 140
C        PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                   DCAR 150
C                                                                       DCAR 160
C     DESCRIPTION OF PARAMETERS                                         DCAR 170
C        X   - THE POINT AT WHICH THE DERIVATIVE IS TO BE COMPUTED      DCAR 180
C        H   - THE NUMBER WHOSE ABSOLUTE VALUE DEFINES THE CLOSED,      DCAR 190
C              SYMMETRIC 2-SIDED INTERVAL ABOUT X (SEE PURPOSE)         DCAR 200
C        IH  - INPUT PARAMETER (SEE REMARKS AND METHOD)                 DCAR 210
C              IH NON-ZERO - THE SUBROUTINE GENERATES THE INTERNAL      DCAR 220
C                            VALUE HH                                   DCAR 230
C              IH    =   0 - THE INTERNAL VALUE HH IS SET TO ABSOLUTE H DCAR 240
C        FCT - THE NAME OF THE EXTERNAL FUNCTION SUBPROGRAM THAT WILL   DCAR 250
C              GENERATE THE NECESSARY FUNCTION VALUES                   DCAR 260
C        Z   - RESULTING DERIVATIVE VALUE                               DCAR 270
C                                                                       DCAR 280
C     REMARKS                                                           DCAR 290
C        (1)  IF H = 0, THEN THERE IS NO COMPUTATION.                   DCAR 300
C        (2)  THE INTERNAL VALUE HH, WHICH IS DETERMINED ACCORDING TO   DCAR 310
C              IH, IS THE MAXIMUM STEP-SIZE USED IN THE COMPUTATION OF  DCAR 320
C              THE CENTRAL DIVIDED DIFFERENCES (SEE METHOD.)  IF IH IS  DCAR 330
C              NON-ZERO, THEN THE SUBROUTINE GENERATES HH ACCORDING TO  DCAR 340
C              CRITERIA THAT BALANCE ROUND-OFF AND TRUNCATION ERROR.  HH DCAR 350
C              IS ALWAYS LESS THAN OR EQUAL TO ABSOLUTE H IN ABSOLUTE   DCAR 360
C              VALUE, SO THAT ALL COMPUTATION OCCURS WITHIN A RADIUS    DCAR 370
C              ABSOLUTE H OF X.                                         DCAR 380
C                                                                       DCAR 390
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     DCAR 400
C        THE EXTERNAL FUNCTION SUBPROGRAM FCT(T) MUST BE FURNISHED BY   DCAR 410
C        THE USER.                                                      DCAR 420
C                                                                       DCAR 430
C     METHOD                                                            DCAR 440
C        THE COMPUTATION OF Z IS BASED ON RICHARDSON'S AND ROMBERG'S    DCAR 450
C        EXTRAPOLATION METHOD AS APPLIED TO THE SEQUENCE OF CENTRAL     DCAR 460
C        DIVIDED DIFFERENCES ASSOCIATED WITH THE POINT PAIRS            DCAR 470
C        (X-(K*HH)/5,X+(K*HH)/5) K=1,...,5.  (SEE FILLIPI, S. AND       DCAR 480
C        ENGELS, H., ALTES UND NEUES ZUR NUMERISCHEN DIFFERENTIATION,   DCAR 490
C        ELECTRONISCHE DATENVERARBEITUNG, ISS. 2 (1966), PP. 57-65.)    DCAR 500
C                                                                       DCAR 510
C     ...................................................................DCAR 520
C                                                                       DCAR 530
      SUBROUTINE DCAR(X,H,IH,FCT,Z)                                     DCAR 540
C                                                                       DCAR 550
C                                                                       DCAR 560
      DIMENSION AUX(5)                                                  DCAR 570
C                                                                       DCAR 580
C        NO ACTION IN CASE OF ZERO INTERVAL LENGTH                      DCAR 590
      IF(H)1,17,1                                                       DCAR 600
C                                                                       DCAR 610
C        GENERATE STEPSIZE HH FOR DIVIDED DIFFERENCES                   DCAR 620
    1 C=ABS(H)                                                          DCAR 630
      IF(IH)2,9,2                                                       DCAR 640
    2 HH=.5                                                             DCAR 650
      IF(C-HH)3,4,4                                                     DCAR 660
    3 HH=C                                                              DCAR 670
    4 A=FCT(X+HH)                                                       DCAR 680
      B=FCT(X-HH)                                                       DCAR 690
      Z=ABS((A-B)/(HH+HH))                                              DCAR 700
      A=.5*ABS(A+B)                                                     DCAR 710
      HH=.5                                                             DCAR 720
      IF(A-1.)6,6,5                                                     DCAR 730
    5 HH=HH*A                                                           DCAR 740
    6 IF(Z-1.)8,8,7                                                     DCAR 750
    7 HH=HH/Z                                                           DCAR 760
    8 IF(HH-C)10,10,9                                                   DCAR 770
    9 HH=C                                                              DCAR 780
C                                                                       DCAR 790
C        INITIALIZE DIFFERENTIATION LOOP                                DCAR 800
   10 Z=(FCT(X+HH)-FCT(X-HH))/(HH+HH)                                   DCAR 810
      J=5                                                               DCAR 820
      JJ=J-1                                                            DCAR 830
      AUX(J)=Z                                                          DCAR 840
      DH=HH/FLOAT(J)                                                    DCAR 850
      DZ=1.E75                                                          DCAR 860
C                                                                       DCAR 870
C        START DIFFERENTIATION LOOP                                     DCAR 880
   11 J=J-1                                                             DCAR 890
      C=J                                                               DCAR 900
      HH=C*DH                                                           DCAR 910
      AUX(J)=(FCT(X+HH)-FCT(X-HH))/(HH+HH)                              DCAR 920
C                                                                       DCAR 930
C        INITIALIZE EXTRAPOLATION LOOP                                  DCAR 940
      D2=1.E75                                                          DCAR 950
      B=0.                                                              DCAR 960
      A=1./C                                                            DCAR 970
C                                                                       DCAR 980
C        START EXTRAPOLATION LOOP                                       DCAR 990
      DO 12 I=J,JJ                                                      DCAR1000
      D1=D2                                                             DCAR1010
      B=B+A                                                             DCAR1020
      HH=(AUX(I)-AUX(I+1))/(B*(2.+B))                                   DCAR1030
      AUX(I+1)=AUX(I)+HH                                                DCAR1040
C                                                                       DCAR1050
C        TEST ON OSCILLATING INCREMENTS                                 DCAR1060
      D2=ABS(HH)                                                        DCAR1070
      IF(D2-D1)12,13,13                                                 DCAR1080
   12 CONTINUE                                                          DCAR1090
C        END OF EXTRAPOLATION LOOP                                      DCAR1100
C                                                                       DCAR1110
C        UPDATE RESULT VALUE Z                                          DCAR1120
      I=JJ+1                                                            DCAR1130
      GO TO 14                                                          DCAR1140
   13 D2=D1                                                             DCAR1150
      JJ=I                                                              DCAR1160
   14 IF(D2-DZ)15,16,16                                                 DCAR1170
   15 DZ=D2                                                             DCAR1180
      Z=AUX(I)                                                          DCAR1190
   16 IF(J-1)17,17,11                                                   DCAR1200
C        END OF DIFFERENTIATION LOOP                                    DCAR1210
C                                                                       DCAR1220
   17 RETURN                                                            DCAR1230
      END                                                               DCAR1240
```

```
C                                                                       DDCA  10
C     ..................................................................DDCA  20
C                                                                       DDCA  30
C     SUBROUTINE DDCAR                                                  DDCA  40
C                                                                       DDCA  50
C     PURPOSE                                                           DDCA  60
C        TO COMPUTE, AT A GIVEN POINT X, AN APPROXIMATION Z TO THE      DDCA  70
C        DERIVATIVE OF AN ANALYTICALLY GIVEN FUNCTION FCT THAT IS 11-   DDCA  80
C        TIMES DIFFERENTIABLE IN A DOMAIN CONTAINING A CLOSED, 2-SIDED  DDCA  90
C        SYMMETRIC INTERVAL OF RADIUS ABSOLUTE H ABOUT X, USING FUNCTIONDDCA 100
C        VALUES ONLY ON THAT CLOSED INTERVAL.                           DDCA 110
C                                                                       DDCA 120
C     USAGE                                                             DDCA 130
C        CALL DDCAR(X,H,IH,FCT,Z)                                       DDCA 140
C        PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT                   DDCA 150
C                                                                       DDCA 160
C     DESCRIPTION OF PARAMETERS                                         DDCA 170
C        X   - THE POINT AT WHICH THE DERIVATIVE IS TO BE COMPUTED      DDCA 180
C              X IS IN DOUBLE PRECISION.                                DDCA 190
C        H   - THE NUMBER WHOSE ABSOLUTE VALUE DEFINES THE CLOSED,      DDCA 200
C              SYMMETRIC 2-SIDED INTERVAL ABOUT X (SEE PURPOSE)         DDCA 210
C              H IS IN SINGLE PRECISION                                 DDCA 220
C        IH  - INPUT PARAMETER (SEE REMARKS AND METHOD)                 DDCA 230
C              IH NON-ZERO - THE SUBROUTINE GENERATES THE INTERNAL      DDCA 240
C                            VALUE HH                                   DDCA 250
C              IH    =   0 - THE INTERNAL VALUE HH IS SET TO ABSOLUTE H DDCA 260
C        FCT - THE NAME OF THE EXTERNAL DOUBLE PRECISION FUNCTION       DDCA 270
C              SUBPROGRAM THAT WILL GENERATE THE NECESSARY FUNCTION     DDCA 280
C              VALUES.                                                  DDCA 290
C        Z   - RESULTING DERIVATIVE VALUE - DOUBLE PRECISION            DDCA 300
C                                                                       DDCA 310
C     REMARKS                                                           DDCA 320
C        (1)  IF H = 0, THEN THERE IS NO COMPUTATION.                   DDCA 330
C        (2)  THE INTERNAL VALUE HH, WHICH IS DETERMINED ACCORDING TO   DDCA 340
C             IH, IS THE MAXIMUM STEP-SIZE USED IN THE COMPUTATION OF   DDCA 350
C             THE CENTRAL DIVIDED DIFFERENCES (SEE METHOD.)  IF IH IS   DDCA 360
C             NON-ZERO, THEN THE SUBROUTINE GENERATES HH ACCORDING TO   DDCA 370
C             CRITERIA THAT BALANCE ROUND-OFF AND TRUNCATION ERROR.  HH DDCA 380
C             IS ALWAYS LESS THAN OR EQUAL TO ABSOLUTE H IN ABSOLUTE    DDCA 390
C             VALUE, SO THAT ALL COMPUTATION OCCURS WITHIN A RADIUS     DDCA 400
C             ABSOLUTE H OF X.                                          DDCA 410
C                                                                       DDCA 420
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     DDCA 430
C        THE EXTERNAL FUNCTION SUBPROGRAM FCT(T) MUST BE FURNISHED BY   DDCA 440
C        THE USER. FCT(T) IS IN DOUBLE PRECISION                        DDCA 450
C                                                                       DDCA 460
C     METHOD                                                            DDCA 470
C        THE COMPUTATION OF Z IS BASED ON RICHARDSON'S AND ROMBERG'S    DDCA 480
C        EXTRAPOLATION METHOD AS APPLIED TO THE SEQUENCE OF CENTRAL     DDCA 490
C        DIVIDED DIFFERENCES ASSOCIATED WITH THE POINT PAIRS            DDCA 500
C        (X-(K*HH)/5,X+(K*HH)/5) K=1,...,5.  (SEE FILLIPI, S. AND       DDCA 510
C        ENGELS, H., ALTES UND NEUES ZUR NUMERISCHEN DIFFERENTIATION,   DDCA 520
C        ELECTRONISCHE DATENVERARBEITUNG, ISS. 2 (1966), PP. 57-65.)    DDCA 530
C                                                                       DDCA 540
C     ..................................................................DDCA 550
C                                                                       DDCA 560
      SUBROUTINE DDCAR(X,H,IH,FCT,Z)                                    DDCA 570
C                                                                       DDCA 580
C                                                                       DDCA 590
      DIMENSION AUX(5)                                                  DDCA 600
      DOUBLE PRECISION X,FCT,Z,AUX,A,B,C,DH,HH                          DDCA 610
C                                                                       DDCA 620
C        NO ACTION IN CASE OF ZERO INTERVAL LENGTH                      DDCA 630
      IF(H)1,17,1                                                       DDCA 640
C                                                                       DDCA 650
C        GENERATE STEPSIZE HH FOR DIVIDED DIFFERENCES                   DDCA 660
    1 C=ABS(H)                                                          DDCA 670
      IF(IH)2,9,2                                                       DDCA 680
    2 HH=.5D-2                                                          DDCA 690
      IF(C-HH)3,4,4                                                     DDCA 700
    3 HH=C                                                              DDCA 710
    4 A=FCT(X+HH)                                                       DDCA 720
      B=FCT(X-HH)                                                       DDCA 730
      Z=DABS((A-B)/(HH+HH))                                             DDCA 740
      A=.5D0*DABS(A+B)                                                  DDCA 750
      HH=.5D-2                                                          DDCA 760
      IF(A-1.D0)6,6,5                                                   DDCA 770
    5 HH=HH*A                                                           DDCA 780
    6 IF(Z-1.D0)8,8,7                                                   DDCA 790
    7 HH=HH/Z                                                           DDCA 800
    8 IF(HH-C)10,10,9                                                   DDCA 810
    9 HH=C                                                              DDCA 820
C                                                                       DDCA 830
C        INITIALIZE DIFFERENTIATION LOOP                                DDCA 840
   10 Z=(FCT(X+HH)-FCT(X-HH))/(HH+HH)                                   DDCA 850
      J=5                                                               DDCA 860
      JJ=J-1                                                            DDCA 870
      AUX(J)=Z                                                          DDCA 880
      DH=HH/DFLOAT(J)                                                   DDCA 890
      DZ=1.E75                                                          DDCA 900
C                                                                       DDCA 910
C        START DIFFERENTIATION LOOP                                     DDCA 920
   11 J=J-1                                                             DDCA 930
      C=J                                                               DDCA 940
      HH=C*DH                                                           DDCA 950
      AUX(J)=(FCT(X+HH)-FCT(X-HH))/(HH+HH)                              DDCA 960
C                                                                       DDCA 970
C        INITIALIZE EXTRAPOLATION LOOP                                  DDCA 980
      D2=1.E75                                                          DDCA 990
      B=0.D0                                                            DDCA1000
      A=1.D0/C                                                          DDCA1010
C                                                                       DDCA1020
C        START EXTRAPOLATION LOOP                                       DDCA1030
      DO 12 I=J,JJ                                                      DDCA1040
      D1=D2                                                             DDCA1050
      B=B+A                                                             DDCA1060
      HH=(AUX(I)-AUX(I+1))/(8*(2.D0+B))                                 DDCA1070
      AUX(I+1)=AUX(I)+HH                                                DDCA1080
C                                                                       DDCA1090
C        TEST ON OSCILLATING INCREMENTS                                 DDCA1100
      D2=DABS(HH)                                                       DDCA1110
      IF(D2-D1)12,13,13                                                 DDCA1120
   12 CONTINUE                                                          DDCA1130
C        END OF EXTRAPOLATION LOOP                                      DDCA1140
C                                                                       DDCA1150
C        UPDATE RESULT VALUE Z                                          DDCA1160
      I=JJ+1                                                            DDCA1170
      GO TO 14                                                          DDCA1180
   13 D2=D1                                                             DDCA1190
      JJ=I                                                              DDCA1200
   14 IF(D2-DZ)15,16,16                                                 DDCA1210
   15 DZ=D2                                                             DDCA1220
      Z=AUX(I)                                                          DDCA1230
   16 IF(J-1)17,17,11                                                   DDCA1240
C        END OF DIFFERENTIATION LOOP                                    DDCA1250
C                                                                       DDCA1260
   17 RETURN                                                            DDCA1270
      END                                                               DDCA1280
```

Subroutines DBAR and DDBAR

Suppose that $y = y(t)$ is eleven-times differentiable in a neighborhood of radius $R > 0$ of the point x. These subroutines compute an approximation Z to $y'(x)$ by applying Richardson's and Romberg's extrapolation method to successively computed one-sided divided differences, using function values in a closed interval $[x, x+h]$, $0 < |h| < R$.

1. Mathematical background

Suppose, first, that $y = y(t)$ is analytic at x; that is, y has a Taylor series expansion about the point x with radius of convergence $R > 0$. Let h be such that $0 < |h| < R$. For each positive integer n, a step size h_1 with $0 < |h_1| \leq |h|$ is computed as described below, and a sequence h_k of increments is generated, where

$$h_k = \frac{n-k+1}{n} h_1$$

for $k = 2, \ldots, n$.

From the sequence $(x, x+h_k)$ of point pairs $(k = 1, \ldots, n)$, the sequence of one-sided divided differences

$$T_{0,k} = \frac{y(x+h_k) - y(x)}{h_k} \quad \text{for } k = 1, \ldots, n \qquad (1)$$

is computed, which forms the first column of the triangular Romberg scheme. These one-sided divided differences $T_{0,k}$ represent the slopes of the secants s_k in Figure 29, in the case $h > 0$.

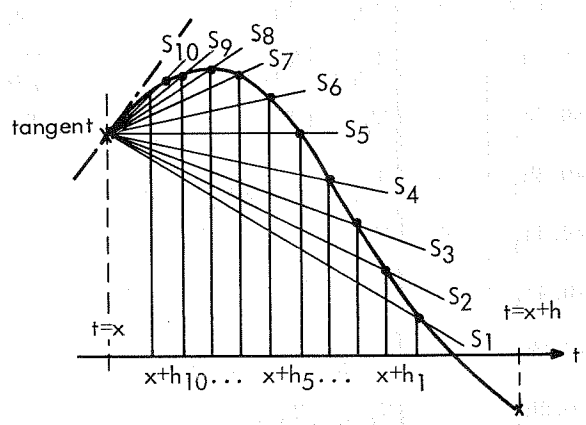


Figure 29. A sequence of secants for a given function $y = y(t)$ and a given argument $t = x$ for the case $n = 10$, $h > 0$

From the Taylor series expansion of $y(x+h_k)$ it follows that

$$T_{0,k} = y'(x) + \frac{h_k}{2!} y''(x) + \frac{h_k^2}{3!} y'''(x) + \ldots \text{ for } k = 1, \ldots, n$$

so that, as an approximation to $y'(x)$, $T_{0,k}$ involves a truncation error of order h_k. Knowing the two divided differences $T_{0,k}$ and $T_{0,k+1}$, we are able to generate the extrapolated value

$$T_{1,k} = T_{0,k+1} + \frac{T_{0,k+1} - T_{0,k}}{a_{1,k} - 1}$$

where (2)

$$a_{1,k} = \left(1 + \frac{1}{n-k}\right)$$

$T_{1,k}$ is a better approximation to $y'(x)$ since

$$T_{1,k} = y'(x) = \frac{1}{3!} \frac{1}{a_{i,k} - 1} h_k^2 y'''(x)$$

$$- \frac{1}{4!} \frac{1}{a_{i,k}} \left(1 + \frac{1}{a_{i,k}}\right) h_k^3 y^{iv}(x)$$

..., which involves a truncation error of order h_k^2.

If we also know $T_{0,k+2}$, then we can generate $T_{1,k+1}$ using formula (2), and further, we can compute the extrapolated value

$$T_{2,k} = T_{1,k+1} + \frac{T_{1,k+1} - T_{1,k}}{a_{2,k} - 1}$$

where

$$a_{2,k} = \left(1 + \frac{2}{n-(k+1)}\right)$$

which involves a truncation error of order h_k^3.

Generally, the order of the truncation error is increased by 1 with each new extrapolation step; in particular, $T_{i,j}$ will involve a truncation error of order

$$h_j^{i+1}, \; i = 0, \ldots, n-1, \; j = 1, \ldots, n.$$

Figure 30 shows the arrangement of the T-values in the triangular Romberg scheme. The T-values are computed following the upward diagonals, using the general formula

$$T_{m,k-m} = T_{m-1,k-m+1} + \frac{T_{m-1,k-m+1} - T_{m-1,k-m}}{\frac{m}{n-k+1}}$$

for $m = 1, \ldots, k-1$ for fixed k, $k = 2, \ldots, n$ (3)

Truncation error		$O(h_k)$	$O(h_k^2)$	$O(h_k^3)$	$O(h_k^4)$	$O(h_k^5)$	$O(h_k^6)$	$O(h_k^7)$	$O(h_k^8)$	$O(h_k^9)$	$O(h_k^{10})$
Stepsize	k \ m	0	1	2	3	4	5	6	7	8	9
h_1	1	$T_{0,1}$	$T_{1,1}$	$T_{2,1}$	$T_{3,1}$	$T_{4,1}$	$T_{5,1}$	$T_{6,1}$	$T_{7,1}$	$T_{8,1}$	$T_{9,1}$
$h_2=0.9h_1$	2	$T_{0,2}$	$T_{1,2}$	$T_{2,2}$	$T_{3,2}$	$T_{4,2}$	$T_{5,2}$	$T_{6,2}$	$T_{7,2}$	$T_{8,2}$	
$h_3=0.8h_1$	3	$T_{0,3}$	$T_{1,3}$	$T_{2,3}$	$T_{3,3}$	$T_{4,3}$	$T_{5,3}$	$T_{6,3}$	$T_{7,3}$		
$h_4=0.7h_1$	4	$T_{0,4}$	$T_{1,4}$	$T_{2,4}$	$T_{3,4}$	$T_{4,4}$	$T_{5,4}$	$T_{6,4}$			
$h_5=0.6h_1$	5	$T_{0,5}$	$T_{1,5}$	$T_{2,5}$	$T_{3,5}$	$T_{4,5}$	$T_{5,5}$				
$h_6=0.5h_1$	6	$T_{0,6}$	$T_{1,6}$	$T_{2,6}$	$T_{3,6}$	$T_{4,6}$					
$h_7=0.4h_1$	7	$T_{0,7}$	$T_{1,7}$	$T_{2,7}$	$T_{3,7}$						
$h_8=0.3h_1$	8	$T_{0,8}$	$T_{1,8}$	$T_{2,8}$							
$h_9=0.2h_1$	9	$T_{0,9}$	$T_{1,9}$								
$h_{10}=0.1h_1$	10	$T_{0,10}$									

Figure 30. The triangular Romberg scheme of T-values for the case n = 10

Numerical experience shows that the accuracy of the results depends heavily on roundoff errors in the one-sided divided differences $T_{0,k}$. Therefore, the choice of the absolutely smallest step size, h_n, is based on the following considerations.

Let

$$v = \begin{cases} 1 \text{ in single-precision computation} \\ 3 \text{ in double-precision computation} \end{cases}$$

Set

$$n_0 = \operatorname{sgn}(h) \cdot \min\left(\frac{n}{2} \cdot 10^{-v}, |h|\right)$$

$$T = (y(x+h_0) - y(x))/h_0.$$

T is an approximation to $y'(x)$.

Assuming that the errors in the function values $y(t)$ for $t \in [x, x+h]$ are bounded by

$$\epsilon = \begin{cases} |y(x)| \cdot 10^{-D} & \text{if } |y(x)| > 1 \\ 10^{-D} & \text{if } |y(x)| \le 1 \end{cases}$$

formula (1) shows that the roundoff error in the computation of $T_{0,n}$ is bounded by

$$R(T_{0,n}) = \frac{2\epsilon}{|h_n|} = \begin{cases} \dfrac{2\,|y(x)|\,10^{-D}}{|h_n|} & \text{if } |y(x)| > 1 \\ 2\,\dfrac{10^{-D}}{|h_n|} & \text{if } |y(x)| \le 1 \end{cases}$$

where D is the number of significant digits in the floating-point representation of numbers. Suppose, also, that we are willing to tolerate a roundoff error

$$R'(T_{0,n}) = \begin{cases} 2T \cdot 10^{-D+v} & \text{if } |T| > 1 \\ 2 \cdot 10^{-D+v} & \text{if } |T| \le 1 \end{cases}$$

Then we must have $R(T_{0,n}) \le R'(T_{0,n})$, which is satisfied when

$$h_n = \frac{\max(1, |y(x)|)}{\max(1, |T|)} \cdot 10^{-v} \tag{4}$$

Finally, we set

$$h_1 = \operatorname{sgn}(h) \cdot \min(n \cdot |h_n|, |h|) \tag{5}$$

guaranteeing that the evaluation of the function $y = y(t)$ is restricted to the closed interval $[x, x+h]$.

2. Programming considerations

Numerical experience shows that, because of increasing roundoff errors, it is generally fruitless to perform more than ten extrapolations. Thus, the subroutines use $n = 10$, and consequently, it is only necessary that $y = y(t)$ be eleven-times differentiable, rather than analytic. It is easy to see that in the case $n = 10$, $y = y(t)$ must be evaluated at twelve points in the closed interval $[x, x+h]$.

The parameter lists of both subroutines contain the decision parameter IH. If the user wants the maximum step size to equal $|h|$, he must set IH = 0. In all other cases (IH $\neq$ 0) the procedure determines the maximum step size by itself, using formulas (4) and (5) with $n = 10$. Step size h_1 and all the other step sizes h_k ($k = 2, \ldots, 10$) are successively stored in the single storage location HH.

As previously explained, the computation of the T-values is performed along the upward diagonals of the triangular Romberg scheme. Therefore, only a one-dimensional internal storage vector, named AUX, with ten storage locations is necessary. Figure 31 shows the storage administration and the sequence of computations (numbers in parentheses).

AUX(1)	$T_{0,10}$(46)						
AUX(2)	$T_{0,9}$(37)	$T_{1,9}$(47)					
AUX(3)	$T_{0,8}$(29)	$T_{1,8}$(38)	$T_{2,8}$(48)				
AUX(4)	$T_{0,7}$(22)	$T_{1,7}$(30)	$T_{2,7}$(39)	...			
AUX(5)	$T_{0,6}$(16)	$T_{1,6}$(23)	$T_{2,6}$(31)	...			
AUX(6)	$T_{0,5}$(11)	$T_{1,5}$(17)	$T_{2,5}$(24)	...			
AUX(7)	$T_{0,4}$(7)	$T_{1,4}$(12)	$T_{2,4}$(18)	...			
AUX(8)	$T_{0,3}$(4)	$T_{1,3}$(8)	$T_{2,3}$(13)	...	$T_{7,3}$(53)		
AUX(9)	$T_{0,2}$(2)	$T_{1,2}$(5)	$T_{2,2}$(9)	...	$T_{7,2}$(44)	$T_{8,2}$(54)	
AUX(10)	$T_{0,1}$(1)	$T_{1,1}$(3)	$T_{2,1}$(6)	...	$T_{7,1}$(36)	$T_{8,1}$(45)	$T_{9,1}$(55)

Figure 31. Storage administration and sequence of calculations

Each extrapolation loop, the computation of the elements on an upward diagonal, is terminated as soon as the absolute values of the differences between adjacent diagonal elements stop decreasing, showing the influence of roundoff errors. The computed T-value that differs least in absolute value from its immediately preceding diagonal neighbor is the desired value Z.

3. Accuracy of results

Tests of the single-precision (double-precision) subroutine on the functions $y(t) = t^2$, $1/t$, e^t, $\tan t$, and $\sin t$ showed that the accuracy obtained using the step size determined by the subroutine was 2-4 (10-12) significant digits near poles, 4-6(13-14) decimal places near points with horizontal tangents, and 4-6(11-13) significant digits otherwise.

For reference see S. Fillipi and H. Engles, "Altes und Neues zur numerischen Differentiation," Elektronische Datenverarbeitung, iss. 2 (1966) pp. 57-65.

```
C
C     ..................................................................
C
C     SUBROUTINE DBAR
C
C     PURPOSE
C        TO COMPUTE, AT A GIVEN POINT X, AN APPROXIMATION Z TO THE
C        DERIVATIVE OF AN ANALYTICALLY GIVEN FUNCTION FCT THAT IS 11-
C        TIMES DIFFERENTIABLE IN A DOMAIN CONTAINING A CLOSED INTERVAL -
C        THE SET OF T BETWEEN X AND X+H (H POSITIVE OR NEGATIVE) - USING
C        FUNCTION VALUES ONLY ON THAT INTERVAL.
C
C     USAGE
C        CALL DBAR(X,H,IH,FCT,Z)
C        PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C     DESCRIPTION OF PARAMETERS
C        X   - THE POINT AT WHICH THE DERIVATIVE IS TO BE COMPUTED
C        H   - THE NUMBER THAT DEFINES THE CLOSED INTERVAL WHOSE END-
C              POINTS ARE X AND X+H (SEE PURPOSE)
C        IH  - INPUT PARAMETER (SEE REMARKS AND METHOD)
C              IH NON-ZERO - THE SUBROUTINE GENERATES THE INTERNAL
C                            VALUE HH
C              IH     =    0 - THE INTERNAL VALUE HH IS SET TO H
C        FCT - THE NAME OF THE EXTERNAL FUNCTION SUBPROGRAM THAT WILL
C              GENERATE THE NECESSARY FUNCTION VALUES
C        Z   - RESULTING DERIVATIVE VALUE
C
C     REMARKS
C        (1)  IF H = 0, THEN THERE IS NO COMPUTATION..
C        (2)  THE (MAGNITUDE OF THE) INTERNAL VALUE HH, WHICH IS DETER-
C             MINED ACCORDING TO IH, IS THE MAXIMUM STEP-SIZE USED IN
C             THE COMPUTATION OF THE ONE-SIDED DIVIDED DIFFERENCES (SEE
C             METHOD.)  IF IH IS NON-ZERO, THEN THE SUBROUTINE GENERATES
C             HH ACCORDING TO CRITERIA THAT BALANCE ROUND-OFF AND TRUN-
C             CATION ERROR.  HH ALWAYS HAS THE SAME SIGN AS H AND IT IS
C             ALWAYS LESS THAN OR EQUAL TO THE MAGNITUDE OF H IN AB-
C             SOLUTE VALUE, SO THAT ALL COMPUTATION OCCURS IN THE CLOSED
C             INTERVAL DETERMINED BY H.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        THE EXTERNAL FUNCTION SUBPROGRAM FCT(T) MUST BE FURNISHED BY
C        THE USER.
C
C     METHOD
C        THE COMPUTATION OF Z IS BASED ON RICHARDSON'S AND ROMBERG'S
C        EXTRAPOLATION METHOD AS APPLIED TO THE SEQUENCE OF ONE-SIDED
C        DIVIDED DIFFERENCES ASSOCIATED WITH THE POINT PAIRS
C        (X,X+(K*HH)/10)K=1,...,10.  (SEE FILLIPI, S. AND ENGELS, H.,
C        ALTES UND NEUES ZUR NUMERISCHEN DIFFERENTIATION, ELECTRONISCHE
C        DATENVERARBEITUNG, ISS. 2 (1966), PP. 57-65.)
C     ..................................................................
C
      SUBROUTINE DBAR(X,H,IH,FCT,Z)
C
C
      DIMENSION AUX(10)
C
C        NO ACTION IN CASE OF ZERO INTERVAL LENGTH
      IF(H)1,17,1
C
C        GENERATE STEPSIZE HH FOR DIVIDED DIFFERENCES
    1 C=ABS(H)
      B=H
      D=X
      D=FCT(D)
      IF(IH)2,9,2
    2 HH=.5
      IF(C-HH)3,4,4
    3 HH=B
    4 HH=SIGN(HH,B)
      Z=ABS((FCT(X+HH)-D)/HH)
      A=ABS(D)
      HH=1.
      IF(A-1.)6,6,5
    5 HH=HH*A
    6 IF(Z-1.)8,8,7
    7 HH=HH/Z
    8 IF(HH-C)10,10,9
    9 HH=B
   10 HH=SIGN(HH,B)
C
C        INITIALIZE DIFFERENTIATION LOOP
      Z=(FCT(X+HH)-D)/HH
      J=10
      JJ=J-1
      AUX(J)=Z
      DH=HH/FLOAT(J)
      DZ=1.E75
C
C        START DIFFERENTIATION LOOP
   11 J=J-1
      C=J
      HH=C*DH
      AUX(J)=(FCT(X+HH)-D)/HH
C
C        INITIALIZE EXTRAPOLATION LOOP
      D2=1.E75
      B=0.
      A=1./C
C
C        START EXTRAPOLATION LOOP
      DO 12 I=J,JJ
      D1=D2
      B=B+A
      HH=(AUX(I)-AUX(I+1))/B
      AUX(I+1)=AUX(I)+HH
C
C        TEST ON OSCILLATING INCREMENTS
      D2=ABS(HH)
      IF(D2-D1)12,13,13
   12 CONTINUE
C        END OF EXTRAPOLATION LOOP
C
C        UPDATE RESULT VALUE Z
      I=JJ+1
      GO TO 14
   13 D2=D1
      JJ=I
   14 IF(D2-DZ)15,16,16
   15 DZ=D2
      Z=AUX(I)
   16 IF(J-1)17,17,11
C        END OF DIFFERENTIATION LOOP
C
   17 RETURN
      END
```

```
C
C     ..................................................................
C
C     SUBROUTINE DDBAR
C
C     PURPOSE
C        TO COMPUTE, AT A GIVEN POINT X, AN APPROXIMATION Z TO THE
C        DERIVATIVE OF AN ANALYTICALLY GIVEN FUNCTION FCT THAT IS 11-
C        TIMES DIFFERENTIABLE IN A DOMAIN CONTAINING A CLOSED INTERVAL -
C        THE SET OF T BETWEEN X AND X+H (H POSITIVE OR NEGATIVE) - USING
C        FUNCTION VALUES ONLY ON THAT INTERVAL.
C
C     USAGE
C        CALL DDBAR(X,H,IH,FCT,Z,)
C        PARAMETER FCT REQUIRES AN EXTERNAL STATEMENT
C
C     DESCRIPTION OF PARAMETERS
C        X   - THE POINT AT WHICH THE DERIVATIVE IS TO BE COMPUTED
C              X IS IN DOUBLE PRECISION
C        H   - THE NUMBER THAT DEFINES THE CLOSED INTERVAL WHOSE END-
C              POINTS ARE X AND X+H (SEE PURPOSE)
C              H IS IN SINGLE PRECISION
C        IH  - INPUT PARAMETER (SEE REMARKS AND METHOD)
C              IH NON-ZERO - THE SUBROUTINE GENERATES THE INTERNAL
C                            VALUE HH
C              IH     =    0 - THE INTERNAL VALUE HH IS SET TO H
C        FCT - THE NAME OF THE EXTERNAL DOUBLE PRECISION FUNCTION
C              SUBPROGRAM THAT WILL GENERATE THE NECESSARY FUNCTION
C              VALUES.
C        Z   - RESULTING DERIVATIVE VALUE - DOUBLE PRECISION
C
C     REMARKS
C        (1)  IF H = 0, THEN THERE IS NO COMPUTATION.
C        (2)  THE (MAGNITUDE OF THE) INTERNAL VALUE HH, WHICH IS DETER-
C             MINED ACCORDING TO IH, IS THE MAXIMUM STEP-SIZE USED IN
C             THE COMPUTATION OF THE ONE-SIDED DIVIDED DIFFERENCES (SEE
C             METHOD.)  IF IH IS NON-ZERO, THEN THE SUBROUTINE GENERATES
C             HH ACCORDING TO CRITERIA THAT BALANCE ROUND-OFF AND TRUN-
C             CATION ERROR.  HH ALWAYS HAS THE SAME SIGN AS H AND IT IS
C             ALWAYS LESS THAN OR EQUAL TO THE MAGNITUDE OF H IN AB-
C             SOLUTE VALUE, SO THAT ALL COMPUTATION OCCURS IN THE CLOSED
C             INTERVAL DETERMINED BY H.
C
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C        THE EXTERNAL FUNCTION SUBPROGRAM FCT(T) MUST BE FURNISHED BY
C        THE USER. FCT(T) IS IN DOUBLE PRECISION
C
C     METHOD
C        THE COMPUTATION OF Z IS BASED ON RICHARDSON'S AND ROMBERG'S
C        EXTRAPOLATION METHOD AS APPLIED TO THE SEQUENCE OF ONE-SIDED
C        DIVIDED DIFFERENCES ASSOCIATED WITH THE POINT PAIRS
C        (X,X+(K*HH)/10)K=1,...,10.  (SEE FILLIPI, S. AND ENGELS, H.,
C        ALTES UND NEUES ZUR NUMERISCHEN DIFFERENTIATION, ELECTRONISCHE
C        DATENVERARBEITUNG, ISS. 2 (1966), PP. 57-65.)
C     ..................................................................
C
      SUBROUTINE DDBAR(X,H,IH,FCT,Z)
C
C
      DIMENSION AUX(10)
      DOUBLE PRECISION X,FCT,Z,AUX,A,B,C,D,DH,HH
C
C        NO ACTION IN CASE OF ZERO INTERVAL LENGTH
      IF(H)1,17,1
C
C        GENERATE STEPSIZE HH FOR DIVIDED DIFFERENCES
    1 C=ABS(H)
      B=H
      D=X
      D=FCT(D)
      IF(IH)2,9,2
    2 HH=.5D-2
      IF(C-HH)3,4,4
    3 HH=B
    4 HH=DSIGN(HH,B)
      Z=DABS((FCT(X+HH)-D)/HH)
      A=DABS(D)
      HH=1.D-2
      IF(A-1.D0)6,6,5
    5 HH=HH*A
    6 IF(Z-1.D0)8,8,7
    7 HH=HH/Z
    8 IF(HH-C)10,10,9
    9 HH=B
   10 HH=DSIGN(HH,B)
C
C        INITIALIZE DIFFERENTIATION LOOP
      Z=(FCT(X+HH)-D)/HH
      J=10
      JJ=J-1
      AUX(J)=Z
      DH=HH/DFLOAT(J)
      DZ=1.E75
C
C        START DIFFERENTIATION LOOP
   11 J=J-1
      C=J
      HH=C*DH
      AUX(J)=(FCT(X+HH)-D)/HH
C
C        INITIALIZE EXTRAPOLATION LOOP
      D2=1.E75
      B=0.D0
      A=1.D0/C
C
C        START EXTRAPOLATION LOOP
      DO 12 I=J,JJ
      D1=D2
      B=B+A
      HH=(AUX(I)-AUX(I+1))/B
      AUX(I+1)=AUX(I)+HH
C
C        TEST ON OSCILLATING INCREMENTS
      D2=DABS(HH)
      IF(D2-D1)12,13,13
   12 CONTINUE
C        END OF EXTRAPOLATION LOOP
C
C        UPDATE RESULT VALUE Z
      I=JJ+1
      GO TO 14
   13 D2=D1
      JJ=I
   14 IF(D2-DZ)15,16,16
   15 DZ=D2
      Z=AUX(I)
   16 IF(J-1)17,17,11
C        END OF DIFFERENTIATION LOOP
C
   17 RETURN
      END
```

Ordinary Differential Equations

Subroutine RK1

This subroutine integrates a given function using the Runge-Kutta technique and produces the final computed value of the integral.

The ordinary differential equation:

$$\frac{dy}{dx} = f(x,y) \tag{1}$$

with initial condition $y(x_0) = y_0$ is solved numerically using a fourth-order Runge-Kutta integration process. This is a single-step method in which the value of y at $x = x_n$ is used to compute $y_{n+1} = y(x_{n+1})$ and earlier values y_{n-1}, y_{n-2}, etc. are not used.

The relevant formulas are:

$$y_{n+1} = y_n + 1/6\,[k_0 + 2k_1 + 2k_2 + k_3] \tag{2}$$

where, for step size h:

$$\begin{cases} k_0 = hf(x_n, y_n) \\ k_1 = hf(x_n + h/2,\ y_n + k_0/2) \\ k_2 = hf(x_n + h/2,\ y_n + k_1/2) \\ k_3 = hf(x_n + h,\ y_n + k_2) \end{cases} \tag{3}$$

```
C                                                                       RK1   10
C     ...................................................................RK1   20
C                                                                       RK1   30
C        SUBROUTINE RK1                                                 RK1   40
C                                                                       RK1   50
C        PURPOSE                                                        RK1   60
C           INTEGRATES A FIRST ORDER DIFFERENTIAL EQUATION              RK1   70
C           DY/DX=FUN(X,Y) UP TO A SPECIFIED FINAL VALUE                RK1   80
C                                                                       RK1   90
C        USAGE                                                          RK1  100
C           CALL RK1(FUN,HI,XI,YI,XF,YF,ANSX,ANSY,IER)                  RK1  110
C                                                                       RK1  120
C        DESCRIPTION OF PARAMETERS                                      RK1  130
C           FUN -USER-SUPPLIED FUNCTION SUBPROGRAM WITH ARGUMENTS X,Y   RK1  140
C                WHICH GIVES DY/DX                                      RK1  150
C           HI  -THE STEP SIZE                                          RK1  160
C           XI  -INITIAL VALUE OF X                                     RK1  170
C           YI  -INITIAL VALUE OF Y WHERE YI=Y(XI)                      RK1  180
C           XF  -FINAL VALUE OF X                                       RK1  190
C           YF  -FINAL VALUE OF Y                                       RK1  200
C           ANSX-RESULTANT FINAL VALUE OF X                             RK1  210
C           ANSY-RESULTANT FINAL VALUE OF Y                             RK1  220
C                EITHER ANSX WILL EQUAL XF OR ANSY WILL EQUAL YF        RK1  230
C                DEPENDING ON WHICH IS REACHED FIRST                    RK1  240
C           IER -ERROR CODE                                             RK1  250
C                IER=0 NO ERROR                                         RK1  260
C                IER=1 STEP SIZE IS ZERO                                RK1  270
C                                                                       RK1  280
C        REMARKS                                                        RK1  290
C           IF XI IS GREATER THAN XF, ANSX=XI AND ANSY=YI               RK1  300
C           IF H IS ZERO, IER IS SET TO ONE, ANSX IS SET TO XI, AND     RK1  310
C           ANSY IS SET TO ZERO                                         RK1  320
C                                                                       RK1  330
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RK1  340
C           FUN IS A TWO ARGUMENT FUNCTION SUBPROGRAM FURNISHED BY THE  RK1  350
C           USER.  DY/DX=FUN (X,Y)                                      RK1  360
C           CALLING PROGRAM MUST HAVE FORTRAN EXTERNAL STATEMENT        RK1  370
C           CONTAINING NAMES OF FUNCTION SUBPROGRAMS LISTED IN CALL TO  RK1  380
C           RK1                                                         RK1  390
C                                                                       RK1  400
C        METHOD                                                         RK1  410
C           USES FOURTH ORDER RUNGE-KUTTA INTEGRATION PROCESS ON A      RK1  420
C           RECURSIVE BASIS AS SHOWN IN F.B. HILDEBRAND, 'INTRODUCTION  RK1  430
C           TO NUMERICAL ANALYSIS',MCGRAW-HILL,1956. PROCESS IS         RK1  440
C           TERMINATED AND FINAL VALUE ADJUSTED WHEN EITHER XF OR YF    RK1  450
C           IS REACHED.                                                 RK1  460
C                                                                       RK1  470
C     ...................................................................RK1  480
C                                                                       RK1  490
      SUBROUTINE RK1(FUN,HI,XI,YI,XF,YF,ANSX,ANSY,IER)                  RK1  500
C                                                                       RK1  510
C     ...................................................................RK1  520
C                                                                       RK1  530
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  RK1  540
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      RK1  550
C        STATEMENT WHICH FOLLOWS.                                       RK1  560
C                                                                       RK1  570
C     DOUBLE PRECISION H1,XI,YI,XF,YF,ANSX,ANSY,H,XN,YN,HNEW,XN1,YN1,   RK1  580
C    1                XX,YY,XNEW,YNEW,H2,T1,T2,T3,T4,FUN               RK1  590
C                                                                       RK1  600
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    RK1  610
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      RK1  620
C        ROUTINE.                                                       RK1  630
C                                                                       RK1  640
C        USER FUNCTION SUBPROGRAM, FUN, MUST BE IN DOUBLE PRECISION.    RK1  650
C                                                                       RK1  660
C        ...................................................................RK1  670
C                                                                       RK1  680
C     IF XF IS LESS THAN OR EQUAL TO XI, RETURN XI,YI AS ANSWER         RK1  690
C                                                                       RK1  700
      IER=0                                                             RK1  705
      IF(XF-XI) 11,11,12                                                RK1  710
   11 ANSX=XI                                                           RK1  720
      ANSY=YI                                                           RK1  730
      RETURN                                                            RK1  740
C                                                                       RK1  750
C     TEST INTERVAL VALUE                                               RK1  760
C                                                                       RK1  770
   12 H=HI                                                              RK1  780
      IF(H) 16,14,20                                                    RK1  790
   14 IER=1                                                             RK1  800
      ANSX=XI                                                           RK1  810
      ANSY=0.0                                                          RK1  820
      RETURN                                                            RK1  830
   16 H=-H                                                              RK1  840
C                                                                       RK1  850
C     SET XN=INITIAL X,YN=INITIAL Y                                     RK1  860
C                                                                       RK1  870
   20 XN=XI                                                             RK1  880
      YN=YI                                                             RK1  890
C                                                                       RK1  900
C     INTEGRATE ONE TIME STEP                                           RK1  910
C                                                                       RK1  920
      HNEW=H                                                            RK1  930
      JUMP=1                                                            RK1  940
      GO TO 170                                                         RK1  950
   25 XN1=XX                                                            RK1  960
      YN1=YY                                                            RK1  970
C                                                                       RK1  980
C     COMPARE XN1 (=X(N+1)) TO X FINAL AND BRANCH ACCORDINGLY           RK1  990
C                                                                       RK1 1000
      IF(XN1-XF)50,30,40                                                RK1 1010
C                                                                       RK1 1020
C     XN1=XF, RETURN (XF,YN1) AS ANSWER                                 RK1 1030
C                                                                       RK1 1040
   30 ANSX=XF                                                           RK1 1050
      ANSY=YN1                                                          RK1 1060
      GO TO 160                                                         RK1 1070
C                                                                       RK1 1080
C     XN1 GREATER THAN XF, SET NEW STEP SIZE AND INTEGRATE ONE STEP     RK1 1090
C     RETURN RESULTS OF INTEGRATION AS ANSWER                           RK1 1100
C                                                                       RK1 1110
   40 HNEW=XF-XN                                                        RK1 1120
      JUMP=2                                                            RK1 1130
      GO TO 170                                                         RK1 1140
   45 ANSX=XX                                                           RK1 1150
      ANSY=YY                                                           RK1 1160
      GO TO 160                                                         RK1 1170
C                                                                       RK1 1180
C     XN1 LESS THAN X FINAL, CHECK IF (XN,YN1) SPAN Y FINAL             RK1 1190
C                                                                       RK1 1200
C                                                                       RK1 1210
   50 IF((YN1-YF)*(YF-YN))60,70,110                                     RK1 1220
C                                                                       RK1 1230
C     YN1 AND YN DO NOT SPAN YF. SET (XN,YN) AS (XN1,YN1) AND REPEAT    RK1 1240
C                                                                       RK1 1250
   60 YN=YN1                                                            RK1 1260
      XN=XN1                                                            RK1 1270
      GO TO 170                                                         RK1 1280
C                                                                       RK1 1290
C     EITHER YN OR YN1 =YF. CHECK WHICH AND SET PROPER (X,Y) AS ANSWER  RK1 1300
C                                                                       RK1 1310
   70 IF(YN1-YF)80,100,60                                               RK1 1320
   80 ANSY=YN                                                           RK1 1330
      ANSX=XN                                                           RK1 1340
      GO TO 160                                                         RK1 1350
  100 ANSY=YN1                                                          RK1 1360
      ANSX=XN1                                                          RK1 1370
      GO TO 160                                                         RK1 1380
C                                                                       RK1 1390
C     YN AND YN1 SPAN YF. TRY TO FIND X VALUE ASSOCIATED WITH YF        RK1 1400
C                                                                       RK1 1410
  110 DO 140 I=1,10                                                     RK1 1420
C                                                                       RK1 1430
C     INTERPOLATE TO FIND NEW TIME STEP AND INTEGRATE ONE STEP          RK1 1440
C     TRY TEN INTERPOLATIONS AT MOST                                    RK1 1450
C                                                                       RK1 1460
      HNEW=((YF-YN )/(YN1-YN))*(XN1-XN)                                 RK1 1470
      JUMP=3                                                            RK1 1480
      GO TO 170                                                         RK1 1490
  115 XNEW=XX                                                           RK1 1500
      YNEW=YY                                                           RK1 1510
C                                                                       RK1 1520
C     COMPARE COMPUTED Y VALUE WITH YF AND BRANCH                       RK1 1530
C                                                                       RK1 1540
      IF(YNEW-YF)120,150,130                                            RK1 1550
C                                                                       RK1 1560
C     ADVANCE, YF IS BETWEEN YNEW AND YN1                               RK1 1570
C                                                                       RK1 1580
  120 YN=YNEW                                                           RK1 1590
      XN=XNEW                                                           RK1 1600
      GO TO 140                                                         RK1 1610
C                                                                       RK1 1620
C     ADVANCE, YF IS BETWEEN YN AND YNEW                                RK1 1630
C                                                                       RK1 1640
  130 YN1=YNEW                                                          RK1 1650
      XN1=XNEW                                                          RK1 1660
  140 CONTINUE                                                          RK1 1670
C                                                                       RK1 1680
C     RETURN (XNEW,YF) AS ANSWER                                        RK1 1690
C                                                                       RK1 1700
  150 ANSX=XNEW                                                         RK1 1710
      ANSY=YF                                                           RK1 1720
  160 RETURN                                                            RK1 1730
C                                                                       RK1 1740
  170 H2=HNEW/2.0                                                       RK1 1750
      T1=HNEW*FUN(XN,YN)                                                RK1 1760
      T2=HNEW*FUN(XN+H2,YN+T1/2.0)                                      RK1 1770
      T3=HNEW*FUN(XN+H2,YN+T2/2.0)                                      RK1 1780
      T4=HNEW*FUN(XN+HNEW,YN+T3)                                        RK1 1790
      YY=YN+(T1+2.0*T2+2.0*T3+T4)/6.0                                   RK1 1800
      XX=XN+HNEW                                                        RK1 1810
      GO TO (25,45,115), JUMP                                           RK1 1820
C                                                                       RK1 1830
      END                                                               RK1 1840
```

Subroutine RK2

This subroutine integrates a given function using the Runge-Kutta technique and produces tabulated values of the computed integral.

The ordinary differential equation:

$$\frac{dy}{dx} = f(x, y) \tag{1}$$

with initial condition $y(x_0) = y_0$ is solved numerically using a fourth-order Runge-Kutta integration process. This is a single-step method in which the value of y at $x = x_n$ is used to compute $y_{n+1} = y(x_{n+1})$ and earlier values y_{n-1}, y_{n-2}, etc. are not used

The relevant formulas are:

$$y_{n+1} = y_n + 1/6\,[k_0 + 2k_1 + 2k_2 + k_3] \tag{2}$$

where, for step size h:

$$\begin{cases} k_0 = hf(x_n, y_n) \\ k_1 = hf(x_n + h/2,\ y_n + k_0/2) \\ k_2 = hf(x_n + h/2,\ y_n + k_1/2) \\ k_3 = hf(x_n + h,\ y_n + k_2) \end{cases} \tag{3}$$

```
C                                                                       RK2   10
C     ...................................................................RK2   20
C                                                                       RK2   30
C        SUBROUTINE RK2                                                 RK2   40
C                                                                       RK2   50
C        PURPOSE                                                        RK2   60
C           INTEGRATES A FIRST ORDER DIFFERENTIAL EQUATION              RK2   70
C           DY/DX=FUN(X,Y) AND PRODUCES A TABLE OF INTEGRATED VALUES    RK2   80
C                                                                       RK2   90
C        USAGE                                                          RK2  100
C           CALL RK2(FUN,H,XI,YI,K,N,VEC)                               RK2  110
C                                                                       RK2  120
C        DESCRIPTION OF PARAMETERS                                      RK2  130
C           FUN-USER-SUPPLIED FUNCTION SUBPROGRAM WITH ARGUMENTS X,Y    RK2  140
C               WHICH GIVES DY/DX                                       RK2  150
C           H  -STEP SIZE                                               RK2  160
C           XI -INITIAL VALUE OF X                                      RK2  170
C           YI -INITIAL VALUE OF Y WHERE YI=Y(XI)                       RK2  180
C           K  -THE INTERVAL AT WHICH COMPUTED VALUES ARE TO BE STORED  RK2  190
C           N  -THE NUMBER OF VALUES TO BE STORED                       RK2  200
C           VEC-THE RESULTANT VECTOR OF LENGTH N IN WHICH COMPUTED      RK2  210
C               VALUES OF Y ARE TO BE STORED                            RK2  220
C                                                                       RK2  230
C        REMARKS                                                        RK2  240
C           NONE                                                        RK2  250
C                                                                       RK2  260
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RK2  270
C           FUN - USER-SUPPLIED FUNCTION SUBPROGRAM FOR DY/DX           RK2  280
C           CALLING PROGRAM MUST HAVE FORTRAN EXTERNAL STATEMENT        RK2  290
C           CONTAINING NAMES OF FUNCTION SUBPROGRAMS LISTED IN CALL TO  RK2  300
C           RK2                                                         RK2  310
C                                                                       RK2  320
C        METHOD                                                         RK2  330
C           FOURTH ORDER RUNGE-KUTTA INTEGRATION ON A RECURSIVE BASIS ASRK2  340
C           SHOWN IN F.B. HILDEBRAND, 'INTRODUCTION TO NUMERICAL        RK2  350
C           ANALYSIS', MCGRAW-HILL, NEW YORK, 1956                      RK2  360
C                                                                       RK2  370
C     ...................................................................RK2  380
C                                                                       RK2  390
      SUBROUTINE RK2(FUN,H,XI,YI,K,N,VEC)                               RK2  400
C                                                                       RK2  410
C        ................................................................RK2  420
C                                                                       RK2  430
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  RK2  440
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      RK2  450
C        STATEMENT WHICH FOLLOWS.                                       RK2  460
C                                                                       RK2  470
C     DOUBLE PRECISION H,XI,YI,VEC,H2,Y,X,T1,T2,T3,T4,FUN               RK2  480
C                                                                       RK2  490
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    RK2  500
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      RK2  510
C        ROUTINE.                                                       RK2  520
C                                                                       RK2  530
C        USER FUNCTION SUBPROGRAM, FUN, MUST BE IN DOUBLE PRECISION.    RK2  540
C                                                                       RK2  550
C        ................................................................RK2  560
C                                                                       RK2  570
      DIMENSION VEC(1)                                                  RK2  580
      H2=H/2.                                                           RK2  590
      Y=YI                                                              RK2  600
      X=XI                                                              RK2  610
      DO 2 I=1,N                                                        RK2  620
      DO 1 J=1,K                                                        RK2  630
      T1=H*FUN(X,Y)                                                     RK2  640
      T2=H*FUN(X+H2,Y+T1/2.)                                            RK2  650
      T3=H*FUN(X+H2,Y+T2/2.)                                            RK2  660
      T4=H*FUN(X+H,Y+T3)                                                RK2  670
      Y= Y+(T1+2.*T2+2.*T3+T4)/6.                                       RK2  680
    1 X=X+H                                                             RK2  690
    2 VEC(I)=Y                                                          RK2  700
      RETURN                                                            RK2  710
      END                                                               RK2  720
```

Subroutines RKGS and DRKGS

These subroutines use the Runge-Kutta method for the solution of initial-value problems.

The purpose of the Runge-Kutta method is to obtain an approximate solution of a system of first-order ordinary differential equations with given initial values. It is a fourth-order integration procedure which is stable and self-starting; that is, only the functional values at a single previous point are required to obtain the functional values ahead. For this reason it is easy to change the step size h at any step in the calculations. On the other hand, each Runge-Kutta step requires the evaluation of the right-hand side of the system four times, which is a great disadvantage compared with other methods of the same order of accuracy, especially predictor-corrector methods. Another disadvantage of the method is that neither the truncation errors nor estimates of them are obtained in the calculation procedure. Therefore, control of accuracy and adjustment of the step size h is done by comparison of the results due to double and single step size 2h and h.

Given the system of first-order ordinary differential equations:

$$y_1' = \frac{dy_1}{dx} = f_1\,(x,\ y_1,\ y_2,\ \ldots,\ y_n)$$

$$y_2' = \frac{dy_2}{dx} = f_2\,(x,\ y_1,\ y_2,\ \ldots,\ y_n)$$

$$\cdots\cdots\cdots\cdots$$

$$y_n' = \frac{dy_n}{dx} = f_n\,(x,\ y_1,\ y_2,\ \ldots,\ y_n)$$

and the initial values:

$$y_1(x_0) = y_{1,0},\ y_2(x_0) = y_{2,0},\ \ldots,\ y_n(x_0) = y_{n,0}$$

and using the following vector notations:

$$Y(x) = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \cdot \\ \cdot \\ \cdot \\ y_n(x) \end{pmatrix},\quad F(x, Y) = \begin{pmatrix} f_1(x, Y) \\ f_2(x, Y) \\ \cdot \\ \cdot \\ \cdot \\ f_n(x, Y) \end{pmatrix},\quad Y_0 = \begin{pmatrix} y_{1,0} \\ y_{2,0} \\ \cdot \\ \cdot \\ \cdot \\ y_{n,0} \end{pmatrix}$$

where Y, F and Y_0 are column vectors, the given problem appears as follows:

$$Y' = \frac{dY}{dx} = F(x, Y) \text{ with } Y(x_0) = Y_0$$

With respect to storage requirements and compensation of accumulated roundoff errors, Gill's modification of the classical Runge-Kutta formulas is preferred. Thus, starting at x_0 with $Y(x_0) = Y_0$ and vector $Q_0 = 0$, the resulting vector $Y_4 = Y(x_0 + h)$ is computed by the following formulas:

$$K_1 = h\,.\ F(x_0,\ Y_0) \quad ;\ Y_1 = Y_0 + \frac{1}{2}(K_1 - 2Q_0) \quad ;$$

$$Q_1 = Q_0 + 3\left[\frac{1}{2}(K_1 - 2Q_0)\right] - \frac{1}{2}K_1$$

$$K_2 = h\,.\ F(x_0 + \frac{h}{2},\ Y_1) \ ;\ Y_2 = Y_1 + (1 - \sqrt{\frac{1}{2}})(K_2 - Q_1) \quad ;$$

$$Q_2 = Q_1 + 3\left[(1 - \sqrt{\frac{1}{2}})(K_2 - Q_1)\right] - (1 - \sqrt{\frac{1}{2}})K_2 \tag{1}$$

$$K_3 = h\,.\ F(x_0 + \frac{h}{2},\ Y_2) \ ;\ Y_3 = Y_2 + (1 + \sqrt{\frac{1}{2}})(K_3 - Q_2) \ ;$$

$$Q_3 = Q_2 + 3\left[(1 + \sqrt{\frac{1}{2}})(K_3 - Q_2)\right] - (1 + \sqrt{\frac{1}{2}})K_3$$

$$K_4 = h\,.\ F(x_0 + h,\ Y_3) \ ;\ Y_4 = Y_3 + \frac{1}{6}(K_4 - 2Q_3) \quad ;$$

$$Q_4 = Q_3 + 3\left[\frac{1}{6}(K_4 - 2Q_3)\right] - \frac{1}{2}K_4$$

where K_1, K_2, K_3, K_4, Y_1, Y_2, Y_3, Y_4, Q_1, Q_2, Q_3, Q_4 are all column vectors with n components. If the procedure were carried out with infinite precision (that is, no rounding errors), vector Q_4 defined above would be zero. In practice this is not true, and Q_4 represents approximately three times the roundoff error in Y_4 accumulated during one step. To compensate for this accumulated roundoff, Q_4 is used as Q_0 for the next step. Also $(x_0 + h)$ and Y_4 serve as x_0 and Y_0 respectively at the next step.

For initial control of accuracy, an approximation for $Y(x_0 + 2h)$ called $Y^{(2)}(x_0 + 2h)$ is computed using the step size 2h, and then an approximation called $Y^{(1)}(x_0 + 2h)$, using two times the step size h. From these two approximations, a test value δ for accuracy is generated in the following way:

$$\delta = \frac{1}{15}\sum_{i=1}^{n} a_i \cdot \left| y_i^{(1)} - y_i^{(2)} \right| \tag{2}$$

where the coefficients a_i are error-weights specified in the input of the procedure.

Test value δ is an approximate measure for the local truncation error at point x_0+2h. If δ is greater than a given tolerance ϵ_2, increment h is halved and the procedure starts again at the point x_0. If δ is less than ϵ_2, the results $Y^{(1)}(x_0+h)$ and $Y^{(1)}(x_0+2h)$

are assumed to be correct. They are then handed, together with $x_0 + h$ and $x_0 + 2h$ and the derivatives at these points -- that is, the values of $F[x_0 + h, Y^{(1)}(x_0+h)]$ and $F[x_0+2h, Y^{(1)}(x_0+2h)]$ respectively -- to a user-supplied output subroutine.

If δ is less than $\epsilon_1 = \epsilon_2/50$, the next step is carried out with the doubled increment. However, care is taken in the procedure that the increment never becomes greater than the increment h specified as an input parameter, and further that all points $x_0 + j \cdot h$ (where $j = 1, 2, \ldots$) which are situated between the lower and upper bound of the integration interval are included in the output. Finally, the increment of the last step of the procedure is chosen in such a way that the upper bound of the integration interval is reached exactly.

The entire input of the procedure is:

1. Lower and upper bound of the integration interval, initial increment of the independent variable, upper bound ϵ_2 of the local truncation error
2. Initial values of the dependent variables and weights for the local truncation errors in each component of the dependent variables
3. The number of differential equations in the system
4. As external subroutine subprograms, the computation of the right-hand side of the system of differential equations; for flexibility in output, an output subroutine
5. An auxiliary storage array named AUX with 8 rows and n columns

Output is done in the following way. If a set of approximations to the dependent variables Y(x) is found to be of sufficient accuracy, it is handed -- together with x, the derivative $F[x, Y(x)]$, the number of bisections of the initial increment, the number of differential equations, the lower and upper bound of the interval, the initial step size, error bound ϵ_2, and a parameter for terminating subroutine RKGS or DRKGS -- to the output subroutine. Because of this output subroutine, the user has the opportunity to choose his own output format, to handle the output values as he wants, to change the upper error bound, and to terminate subroutine RKGS or DRKGS at any output point. In particular, the user is able to drop the output of some intermediate points, printing only the result values at the special points $x_0 + n \cdot h$ ($n = 0, 1, 2, \ldots$). The user may also perform intermediate computation using the integration results before continuing the process.

For better understanding of the flowchart and of the FORTRAN program, Figure 32 shows the allocation of special intermediate result vectors within the storage array AUX.

For reference see A. Ralston/H. S. Wilf, Mathematical Methods for Digital Computers, Wiley, New York/London, 1960, pp. 110-120.

AUX

function vector Y(x)	1. row (AUX (1) in flowchart)
derivative vector F(x, Y(x))	2. row (AUX (2) in flowchart)
vector of accumulated roundoff at point x	3. row (AUX (3) in flowchart)
function vector Y(x+2h) for testing purposes	4. row (AUX (4) in flowchart)
function vector Y(x+h)	5. row (AUX (5) in flowchart)
vector of accumulated roundoff at point x+h	6. row (AUX (6) in flowchart)
derivative vector F(x+h, Y(x+h))	7. row (AUX (7) in flowchart)
vector of error weights multiplied by 1/15	8. row (AUX (8) in flowchart)

Figure 32. Storage allocation in auxiliary storage array AUX (RKGS-DRKGS)

```
C                                                                       RKGS  10
C     ..................................................................RKGS  20
C                                                                       RKGS  30
C        SUBROUTINE RKGS                                                RKGS  40
C                                                                       RKGS  50
C        PURPOSE                                                        RKGS  60
C           TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY DIFFERENTIAL      RKGS  70
C           EQUATIONS WITH GIVEN INITIAL VALUES.                        RKGS  80
C                                                                       RKGS  90
C        USAGE                                                          RKGS 100
C           CALL RKGS (PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)              RKGS 110
C           PARAMETERS FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.      RKGS 120
C                                                                       RKGS 130
C        DESCRIPTION OF PARAMETERS                                      RKGS 140
C           PRMT    - AN INPUT AND OUTPUT VECTOR WITH DIMENSION GREATER RKGS 150
C                     OR EQUAL TO 5, WHICH SPECIFIES THE PARAMETERS OF  RKGS 160
C                     THE INTERVAL AND OF ACCURACY AND WHICH SERVES FOR RKGS 170
C                     COMMUNICATION BETWEEN OUTPUT SUBROUTINE (FURNISHED RKGS 180
C                     BY THE USER) AND SUBROUTINE RKGS. EXCEPT PRMT(5)  RKGS 190
C                     THE COMPONENTS ARE NOT DESTROYED BY SUBROUTINE    RKGS 200
C                     RKGS AND THEY ARE                                 RKGS 210
C           PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),               RKGS 220
C           PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),               RKGS 230
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE      RKGS 240
C                     (INPUT),                                          RKGS 250
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS    RKGS 260
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.      RKGS 270
C                     IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE    RKGS 280
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.RKGS 290
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS       RKGS 300
C                     OUTPUT SUBROUTINE.                                RKGS 310
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE RKGS INITIALIZES    RKGS 320
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE         RKGS 330
C                     SUBROUTINE RKGS AT ANY OUTPUT POINT, HE HAS TO    RKGS 340
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE RKGS 350
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE       RKGS 360
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER      RKGS 370
C                     THAN 5. HOWEVER SUBROUTINE RKGS DOES NOT REQUIRE  RKGS 380
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL  RKGS 390
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM     RKGS 400
C                     (CALLING RKGS) WHICH ARE OBTAINED BY SPECIAL      RKGS 410
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP. RKGS 420
C           Y       - INPUT VECTOR OF INITIAL VALUES.  (DESTROYED)      RKGS 430
C                     LATERON Y IS THE RESULTING VECTOR OF DEPENDENT    RKGS 440
C                     VARIABLES COMPUTED AT INTERMEDIATE POINTS X.      RKGS 450
C           DERY    - INPUT VECTOR OF ERROR WEIGHTS.  (DESTROYED)       RKGS 460
C                     THE SUM OF ITS COMPONENTS MUST BE EQUAL TO 1.     RKGS 470
C                     LATERON DERY IS THE VECTOR OF DERIVATIVES, WHICH  RKGS 480
C                     BELONG TO FUNCTION VALUES Y AT A POINT X.         RKGS 490
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF     RKGS 500
C                     EQUATIONS IN THE SYSTEM.                          RKGS 510
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF    RKGS 520
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS RKGS 530
C                     GREATER THAN 10, SUBROUTINE RKGS RETURNS WITH     RKGS 540
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM. ERROR    RKGS 550
C                     MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE        RKGS 560
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-RKGS 570
C                     PRMT(1)) RESPECTIVELY.                            RKGS 580
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. THIS     RKGS 590
C                     SUBROUTINE COMPUTES THE RIGHT HAND SIDES DERY OF  RKGS 600
C                     THE SYSTEM TO GIVEN VALUES X AND Y. ITS PARAMETER RKGS 610
C                     LIST MUST BE X,Y,DERY. SUBROUTINE FCT SHOULD      RKGS 620
C                     NOT DESTROY X AND Y.                              RKGS 630
C           OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.   RKGS 640
C                     ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.RKGS 650
C                     NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,   RKGS 660
C                     PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY         RKGS 670
C                     SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,RKGS 680
C                     SUBROUTINE RKGS IS TERMINATED.                    RKGS 690
C           AUX     - AN AUXILIARY STORAGE ARRAY WITH 8 ROWS AND NDIM   RKGS 700
C                     COLUMNS.                                          RKGS 710
C                                                                       RKGS 720
C        REMARKS                                                        RKGS 730
C           THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF RKGS 740
C           (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE    RKGS 750
```

```
C              NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE   RKGS 760
C              IHLF=11),                                               RKGS 770
C           (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN      RKGS 780
C              (ERROR MESSAGES IHLF=12 OR IHLF=13),                    RKGS 790
C           (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,      RKGS 800
C           (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.       RKGS 810
C                                                                      RKGS 820
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                 RKGS 830
C           THE EXTERNAL SUBROUTINES FCT(X,Y,DERY) AND                 RKGS 840
C           OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.RKGS 850
C                                                                      RKGS 860
C        METHOD                                                        RKGS 870
C           EVALUATION IS DONE BY MEANS OF FOURTH ORDER RUNGE-KUTTA    RKGS 880
C           FORMULAE IN THE MODIFICATION DUE TO GILL. ACCURACY IS      RKGS 890
C           TESTED COMPARING THE RESULTS OF THE PROCEDURE WITH SINGLE  RKGS 900
C           AND DOUBLE INCREMENT.                                      RKGS 910
C           SUBROUTINE RKGS AUTOMATICALLY ADJUSTS THE INCREMENT DURING RKGS 920
C           THE WHOLE COMPUTATION BY HALVING OR DOUBLING. IF MORE THAN RKGS 930
C           10 BISECTIONS OF THE INCREMENT ARE NECESSARY TO GET        RKGS 940
C           SATISFACTORY ACCURACY, THE SUBROUTINE RETURNS WITH         RKGS 950
C           ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.                   RKGS 960
C           TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE    RKGS 970
C           MUST BE FURNISHED BY THE USER.                             RKGS 980
C           FOR REFERENCE, SEE                                         RKGS 990
C           RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL COMPUTERS,  RKGS1000
C           WILEY, NEW YORK/LONDON, 1960, PP.110-120.                  RKGS1010
C                                                                      RKGS1020
C     ..................................................................RKGS1030
C                                                                      RKGS1040
      SUBROUTINE RKGS(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)              RKGS1050
C                                                                      RKGS1060
C                                                                      RKGS1070
      DIMENSION Y(1),DERY(1),AUX(8,1),A(4),B(4),C(4),PRMT(1)           RKGS1080
      DO 1 I=1,NDIM                                                    RKGS1090
    1 AUX(8,I)=.06666667*DERY(I)                                       RKGS1100
      X=PRMT(1)                                                        RKGS1110
      XEND=PRMT(2)                                                     RKGS1120
      H=PRMT(3)                                                        RKGS1130
      PRMT(5)=0.                                                       RKGS1140
      CALL FCT(X,Y,DERY)                                               RKGS1150
C                                                                      RKGS1160
C     ERROR TEST                                                       RKGS1170
      IF(H*(XEND-X))38,37,2                                            RKGS1180
C                                                                      RKGS1190
C     PREPARATIONS FOR RUNGE-KUTTA METHOD                              RKGS1200
    2 A(1)=.5                                                          RKGS1210
      A(2)=.2928932                                                    RKGS1220
      A(3)=1.707107                                                    RKGS1230
      A(4)=.1666667                                                    RKGS1240
      B(1)=2.                                                          RKGS1250
      B(2)=1.                                                          RKGS1260
      B(3)=1.                                                          RKGS1270
      B(4)=2.                                                          RKGS1280
      C(1)=.5                                                          RKGS1290
      C(2)=.2928932                                                    RKGS1300
      C(3)=1.707107                                                    RKGS1310
      C(4)=.5                                                          RKGS1320
C                                                                      RKGS1330
C     PREPARATIONS OF FIRST RUNGE-KUTTA STEP                           RKGS1340
      DO 3 I=1,NDIM                                                    RKGS1350
      AUX(1,I)=Y(I)                                                    RKGS1360
      AUX(2,I)=DERY(I)                                                 RKGS1370
      AUX(3,I)=0.                                                      RKGS1380
    3 AUX(6,I)=0.                                                      RKGS1390
      IREC=0                                                           RKGS1400
      H=H+H                                                            RKGS1410
      IHLF=-1                                                          RKGS1420
      ISTEP=0                                                          RKGS1430
      IEND=0                                                           RKGS1440
C                                                                      RKGS1450
C                                                                      RKGS1460
C     START OF A RUNGE-KUTTA STEP                                      RKGS1470
    4 IF((X+H-XEND)*H)7,6,5                                            RKGS1480
    5 H=XEND-X                                                         RKGS1490
    6 IEND=1                                                           RKGS1500
C                                                                      RKGS1510
C     RECORDING OF INITIAL VALUES OF THIS STEP                         RKGS1520
    7 CALL OUTP(X,Y,DERY,IREC,NDIM,PRMT)                               RKGS1530
      IF(PRMT(5))40,8,40                                               RKGS1540
    8 ITEST=0                                                          RKGS1550
    9 ISTEP=ISTEP+1                                                    RKGS1560
C                                                                      RKGS1570
C                                                                      RKGS1580
C     START OF INNERMOST RUNGE-KUTTA LOOP                              RKGS1590
      J=1                                                              RKGS1600
   10 AJ=A(J)                                                          RKGS1610
      BJ=B(J)                                                          RKGS1620
      CJ=C(J)                                                          RKGS1630
      DO 11 I=1,NDIM                                                   RKGS1640
      R1=H*DERY(I)                                                     RKGS1650
      R2=AJ*(R1-BJ*AUX(6,I))                                           RKGS1660
      Y(I)=Y(I)+R2                                                     RKGS1670
      R2=R2+R2+R2                                                      RKGS1680
   11 AUX(6,I)=AUX(6,I)+R2-CJ*R1                                       RKGS1690
      IF(J-4)12,15,15                                                  RKGS1700
   12 J=J+1                                                            RKGS1710
      IF(J-3)13,14,13                                                  RKGS1720
   13 X=X+.5*H                                                         RKGS1730
   14 CALL FCT(X,Y,DERY)                                               RKGS1740
      GOTO 10                                                          RKGS1750
C     END OF INNERMOST RUNGE-KUTTA LOOP                                RKGS1760
C                                                                      RKGS1770
C                                                                      RKGS1780
C     TEST OF ACCURACY                                                 RKGS1790
   15 IF(ITEST)16,16,20                                                RKGS1800
C                                                                      RKGS1810
C     IN CASE ITEST=0 THERE IS NO POSSIBILITY FOR TESTING OF ACCURACY  RKGS1820
   16 DO 17 I=1,NDIM                                                   RKGS1830
   17 AUX(4,I)=Y(I)                                                    RKGS1840
      ITEST=1                                                          RKGS1850
      ISTEP=ISTEP+ISTEP-2                                              RKGS1860
   18 IHLF=IHLF+1                                                      RKGS1870
      X=X-H                                                            RKGS1880
      H=.5*H                                                           RKGS1890
      DO 19 I=1,NDIM                                                   RKGS1900
      Y(I)=AUX(1,I)                                                    RKGS1910
      DERY(I)=AUX(2,I)                                                 RKGS1920
   19 AUX(6,I)=AUX(3,I)                                                RKGS1930
      GOTO 9                                                           RKGS1940
C                                                                      RKGS1950
C     IN CASE ITEST=1 TESTING OF ACCURACY IS POSSIBLE                  RKGS1960
   20 IMOD=ISTEP/2                                                     RKGS1970
      IF(ISTEP-IMOD-IMOD)21,23,21                                      RKGS1980
   21 CALL FCT(X,Y,DERY)                                               RKGS1990
      DO 22 I=1,NDIM                                                   RKGS2000
      AUX(5,I)=Y(I)                                                    RKGS2010
   22 AUX(7,I)=DERY(I)                                                 RKGS2020
      GOTO 9                                                           RKGS2030
C                                                                      RKGS2040
C     COMPUTATION OF TEST VALUE DELT                                   RKGS2050
   23 DELT=0.                                                          RKGS2060
      DO 24 I=1,NDIM                                                   RKGS2070
   24 DELT=DELT+AUX(8,I)*ABS(AUX(4,I)-Y(I))                            RKGS2080
      IF(DELT-PRMT(4))28,28,25                                         RKGS2090
C                                                                      RKGS2100
C     ERROR IS TOO GREAT                                               RKGS2110
   25 IF(IHLF-10)26,36,36                                              RKGS2120
   26 DO 27 I=1,NDIM                                                   RKGS2130
   27 AUX(4,I)=AUX(5,I)                                                RKGS2140
      ISTEP=ISTEP+ISTEP-4                                              RKGS2150
      X=X-H                                                            RKGS2160
      IEND=0                                                           RKGS2170
      GOTO 18                                                          RKGS2180
C                                                                      RKGS2190
C     RESULT VALUES ARE GOOD                                           RKGS2200
   28 CALL FCT(X,Y,DERY)                                               RKGS2210
      DO 29 I=1,NDIM                                                   RKGS2220
      AUX(1,I)=Y(I)                                                    RKGS2230
      AUX(2,I)=DERY(I)                                                 RKGS2240
      AUX(3,I)=AUX(6,I)                                                RKGS2250
      Y(I)=AUX(5,I)                                                    RKGS2260
   29 DERY(I)=AUX(7,I)                                                 RKGS2270
      CALL OUTP(X-H,Y,DERY,IHLF,NDIM,PRMT)                             RKGS2280
      IF(PRMT(5))40,30,40                                              RKGS2290
   30 DO 31 I=1,NDIM                                                   RKGS2300
      Y(I)=AUX(1,I)                                                    RKGS2310
   31 DERY(I)=AUX(2,I)                                                 RKGS2320
      IREC=IHLF                                                        RKGS2330
      IF(IEND)32,32,39                                                 RKGS2340
C                                                                      RKGS2350
C     INCREMENT GETS DOUBLED                                           RKGS2360
   32 IHLF=IHLF-1                                                      RKGS2370
      ISTEP=ISTEP/2                                                    RKGS2380
      H=H+H                                                            RKGS2390
      IF(IHLF)4,33,33                                                  RKGS2400
   33 IMOD=ISTEP/2                                                     RKGS2410
      IF(ISTEP-IMOD-IMOD)4,34,4                                        RKGS2420
   34 IF(DELT-.02*PRMT(4))35,35,4                                      RKGS2430
   35 IHLF=IHLF-1                                                      RKGS2440
      ISTEP=ISTEP/2                                                    RKGS2450
      H=H+H                                                            RKGS2460
      GOTO 4                                                           RKGS2470
C                                                                      RKGS2480
C                                                                      RKGS2490
C     RETURNS TO CALLING PROGRAM                                       RKGS2500
   36 IHLF=11                                                          RKGS2510
      CALL FCT(X,Y,DERY)                                               RKGS2520
      GOTO 39                                                          RKGS2530
   37 IHLF=12                                                          RKGS2540
      GOTO 39                                                          RKGS2550
   38 IHLF=13                                                          RKGS2560
   39 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                               RKGS2570
   40 RETURN                                                           RKGS2580
      END                                                              RKGS2590
```

```
C                                                                      DRKG  10
C     ..................................................................DRKG  20
C                                                                      DRKG  30
C        SUBROUTINE DRKGS                                              DRKG  40
C                                                                      DRKG  50
C        PURPOSE                                                       DRKG  60
C           TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY DIFFERENTIAL     DRKG  70
C           EQUATIONS WITH GIVEN INITIAL VALUES.                       DRKG  80
C                                                                      DRKG  90
C        USAGE                                                         DRKG 100
C           CALL DRKGS (PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)            DRKG 110
C           PARAMETERS FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.     DRKG 120
C                                                                      DRKG 130
C        DESCRIPTION OF PARAMETERS                                     DRKG 140
C           PRMT   - DOUBLE PRECISION INPUT AND OUTPUT VECTOR WITH     DRKG 150
C                    DIMENSION GREATER THAN OR EQUAL TO 5, WHICH       DRKG 160
C                    SPECIFIES THE PARAMETERS OF THE INTERVAL AND OF   DRKG 170
C                    ACCURACY AND WHICH SERVES FOR COMMUNICATION BETWEENDRKG 180
C                    OUTPUT SUBROUTINE (FURNISHED BY THE USER) AND     DRKG 190
C                    SUBROUTINE DRKGS. EXCEPT PRMT(5) THE COMPONENTS   DRKG 200
C                    ARE NOT DESTROYED BY SUBROUTINE DRKGS AND THEY ARE DRKG 210
C           PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),              DRKG 220
C           PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),              DRKG 230
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE     DRKG 240
C                    (INPUT),                                          DRKG 250
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS   DRKG 260
C                    GREATER THAN PRMT(4), INCREMENT GETS HALVED.      DRKG 270
C                    IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE    DRKG 280
C                    ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.DRKG 290
C                    THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS       DRKG 300
C                    OUTPUT SUBROUTINE.                                DRKG 310
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE DRKGS INITIALIZES  DRKG 320
C                    PRMT(5)=0. IF THE USER WANTS TO TERMINATE         DRKG 330
C                    SUBROUTINE DRKGS AT ANY OUTPUT POINT, HE HAS TO   DRKG 340
C                    CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE DRKG 350
C                    OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE       DRKG 360
C                    FEASIBLE IF ITS DIMENSION IS DEFINED GREATER      DRKG 370
C                    THAN 5. HOWEVER SUBROUTINE DRKGS DOES NOT REQUIRE DRKG 380
C                    AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL  DRKG 390
C                    FOR HANDING RESULT VALUES TO THE MAIN PROGRAM     DRKG 400
C                    (CALLING DRKGS) WHICH ARE OBTAINED BY SPECIAL     DRKG 410
C                    MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.DRKG 420
C           Y      - DOUBLE PRECISION INPUT VECTOR OF INITIAL VALUES  DRKG 430
C                    (DESTROYED). LATERON Y IS THE RESULTING VECTOR OF DRKG 440
C                    DEPENDENT VARIABLES COMPUTED AT INTERMEDIATE      DRKG 450
C                    POINTS X.                                         DRKG 460
C           DERY   - DOUBLE PRECISION INPUT VECTOR OF ERROR WEIGHTS   DRKG 470
C                    (DESTROYED). THE SUM OF ITS COMPONENTS MUST BE    DRKG 480
C                    EQUAL TO 1. LATERON DERY IS THE VECTOR OF         DRKG 490
C                    DERIVATIVES, WHICH BELONG TO FUNCTION VALUES Y AT DRKG 500
C                    INTERMEDIATE POINTS X.                            DRKG 510
C           NDIM   - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF     DRKG 520
C                    EQUATIONS IN THE SYSTEM.                          DRKG 530
C           IHLF   - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF    DRKG 540
C                    BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS DRKG 550
C                    GREATER THAN 10, SUBROUTINE DRKGS RETURNS WITH    DRKG 560
C                    ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM. ERROR    DRKG 570
C                    MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE        DRKG 580
C                    PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-DRKG 590
C                    PRMT(1)) RESPECTIVELY.                            DRKG 600
C           FCT    - THE NAME OF AN EXTERNAL SUBROUTINE USED. THIS    DRKG 610
C                    SUBROUTINE COMPUTES THE RIGHT HAND SIDES DERY OF  DRKG 620
C                    THE SYSTEM TO GIVEN VALUES X AND Y. ITS PARAMETER DRKG 630
C                    LIST MUST BE X,Y,DERY. SUBROUTINE FCT SHOULD      DRKG 640
C                    NOT DESTROY X AND Y.                              DRKG 650
```

```
C          OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.   DRKG 660
C                    ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.DRKG 670
C                    NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,    DRKG 680
C                    PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY          DRKG 690
C                    SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,DRKG 700
C                    SUBROUTINE DRKGS IS TERMINATED.                    DRKG 710
C          AUX     - DOUBLE PRECISION AUXILIARY STORAGE ARRAY WITH 8    DRKG 720
C                    ROWS AND NDIM COLUMNS.                             DRKG 730
C                                                                       DRKG 740
C       REMARKS                                                         DRKG 750
C          THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF  DRKG 760
C          (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE     DRKG 770
C              NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE    DRKG 780
C              IHLF=11),                                                DRKG 790
C          (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN        DRKG 800
C              (ERROR MESSAGES IHLF=12 OR IHLF=13),                     DRKG 810
C          (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,        DRKG 820
C          (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.         DRKG 830
C                                                                       DRKG 840
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DRKG 850
C          THE EXTERNAL SUBROUTINES FCT(X,Y,DERY) AND                   DRKG 860
C          OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.DRKG 870
C                                                                       DRKG 880
C       METHOD                                                          DRKG 890
C          EVALUATION IS DONE BY MEANS OF FOURTH ORDER RUNGE-KUTTA      DRKG 900
C          FORMULAE IN THE MODIFICATION DUE TO GILL. ACCURACY IS        DRKG 910
C          TESTED COMPARING THE RESULTS OF THE PROCEDURE WITH SINGLE    DRKG 920
C          AND DOUBLE INCREMENT.                                        DRKG 930
C          SUBROUTINE DRKGS AUTOMATICALLY ADJUSTS THE INCREMENT DURING DRKG 940
C          THE WHOLE COMPUTATION BY HALVING OR DOUBLING. IF MORE THAN   DRKG 950
C          10 BISECTIONS OF THE INCREMENT ARE NECESSARY TO GET          DRKG 960
C          SATISFACTORY ACCURACY, THE SUBROUTINE RETURNS WITH           DRKG 970
C          ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.                     DRKG 980
C          TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE      DRKG 990
C          MUST BE FURNISHED BY THE USER.                               DRKG1000
C          FOR REFERENCE, SEE                                           DRKG1010
C          RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL COMPUTERS,    DRKG1020
C          WILEY, NEW YORK/LONDON, 1960, PP.110-120.                    DRKG1030
C                                                                       DRKG1040
C     ..................................................................DRKG1050
C                                                                       DRKG1060
      SUBROUTINE DRKGS(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)              DRKG1070
C                                                                       DRKG1080
C                                                                       DRKG1090
      DIMENSION Y(1),DERY(1),AUX(8,1),A(4),B(4),C(4),PRMT(1)            DRKG1100
      DOUBLE PRECISION PRMT,Y,DERY,AUX,A,B,C,X,XEND,H,AJ,BJ,CJ,R1,R2,   DRKG1110
     1DELT                                                              DRKG1120
      DO 1 I=1,NDIM                                                     DRKG1130
    1 AUX(8,I)=.06666666666666666700*DERY(I)                            DRKG1140
      X=PRMT(1)                                                         DRKG1150
      XEND=PRMT(2)                                                      DRKG1160
      H=PRMT(3)                                                         DRKG1170
      PRMT(5)=0.D0                                                      DRKG1180
      CALL FCT(X,Y,DERY)                                                DRKG1190
C                                                                       DRKG1200
C     ERROR TEST                                                        DRKG1210
      IF(H*(XEND-X))38,37,2                                             DRKG1220
C                                                                       DRKG1230
C     PREPARATIONS FOR RUNGE-KUTTA METHOD                               DRKG1240
    2 A(1)=.5D0                                                         DRKG1250
      A(2)=.29289321881345248D0                                         DRKG1260
      A(3)=1.7071067811865475D0                                         DRKG1270
      A(4)=.16666666666666667D0                                         DRKG1280
      B(1)=2.D0                                                         DRKG1290
      B(2)=1.D0                                                         DRKG1300
      B(3)=1.D0                                                         DRKG1310
      B(4)=2.D0                                                         DRKG1320
      C(1)=.5D0                                                         DRKG1330
      C(2)=.29289321881345248D0                                         DRKG1340
      C(3)=1.7071067811865475D0                                         DRKG1350
      C(4)=.5D0                                                         DRKG1360
C                                                                       DRKG1370
C     PREPARATIONS OF FIRST RUNGE-KUTTA STEP                            DRKG1380
      DO 3 I=1,NDIM                                                     DRKG1390
      AUX(1,I)=Y(I)                                                     DRKG1400
      AUX(2,I)=DERY(I)                                                  DRKG1410
      AUX(3,I)=0.D0                                                     DRKG1420
    3 AUX(6,I)=0.D0                                                     DRKG1430
      IREC=0                                                            DRKG1440
      H=H+H                                                             DRKG1450
      IHLF=-1                                                           DRKG1460
      ISTEP=0                                                           DRKG1470
      IEND=0                                                            DRKG1480
C                                                                       DRKG1490
C                                                                       DRKG1500
C     START OF A RUNGE-KUTTA STEP                                       DRKG1510
    4 IF((X+H-XEND)*H)7,6,5                                             DRKG1520
    5 H=XEND-X                                                          DRKG1530
    6 IEND=1                                                            DRKG1540
C                                                                       DRKG1550
C     RECORDING OF INITIAL VALUES OF THIS STEP                          DRKG1560
    7 CALL OUTP(X,Y,DERY,IREC,NDIM,PRMT)                                DRKG1570
      IF(PRMT(5))40,8,40                                                DRKG1580
    8 ITEST=0                                                           DRKG1590
    9 ISTEP=ISTEP+1                                                     DRKG1600
C                                                                       DRKG1610
C                                                                       DRKG1620
C     START OF INNERMOST RUNGE-KUTTA LOOP                               DRKG1630
      J=1                                                               DRKG1640
   10 AJ=A(J)                                                           DRKG1650
      BJ=B(J)                                                           DRKG1660
      CJ=C(J)                                                           DRKG1670
      DO 11 I=1,NDIM                                                    DRKG1680
      R1=H*DERY(I)                                                      DRKG1690
      R2=AJ*(R1-BJ*AUX(6,I))                                            DRKG1700
      Y(I)=Y(I)+R2                                                      DRKG1710
      R2=R2+R2+R2                                                       DRKG1720
   11 AUX(6,I)=AUX(6,I)+R2-CJ*R1                                        DRKG1730
      IF(J-4)12,15,15                                                   DRKG1740
   12 J=J+1                                                             DRKG1750
      IF(J-3)13,14,13                                                   DRKG1760
   13 X=X+.5D0*H                                                        DRKG1770
   14 CALL FCT(X,Y,DERY)                                                DRKG1780
      GOTO 10                                                           DRKG1790
C     END OF INNERMOST RUNGE-KUTTA LOOP                                 DRKG1800
C                                                                       DRKG1810
C                                                                       DRKG1820
C     TEST OF ACCURACY                                                  DRKG1830
   15 IF(ITEST)16,16,20                                                 DRKG1840
C                                                                       DRKG1850
C     IN CASE ITEST=0 THERE IS NO POSSIBILITY FOR TESTING OF ACCURACY   DRKG1860
   16 DO 17 I=1,NDIM                                                    DRKG1870
   17 AUX(4,I)=Y(I)                                                     DRKG1880
      ITEST=1                                                           DRKG1890
      ISTEP=ISTEP+ISTEP-2                                               DRKG1900
   18 IHLF=IHLF+1                                                       DRKG1910
      X=X-H                                                             DRKG1920
      H=.5D0*H                                                          DRKG1930
      DO 19 I=1,NDIM                                                    DRKG1940
      Y(I)=AUX(1,I)                                                     DRKG1950
      DERY(I)=AUX(2,I)                                                  DRKG1960
   19 AUX(6,I)=AUX(3,I)                                                 DRKG1970
      GOTO 9                                                            DRKG1980
C                                                                       DRKG1990
C     IN CASE ITEST=1 TESTING OF ACCURACY IS POSSIBLE                   DRKG2000
   20 IMOD=ISTEP/2                                                      DRKG2010
      IF(ISTEP-IMOD-IMOD)21,23,21                                       DRKG2020
   21 CALL FCT(X,Y,DERY)                                                DRKG2030
      DO 22 I=1,NDIM                                                    DRKG2040
      AUX(5,I)=Y(I)                                                     DRKG2050
   22 AUX(7,I)=DERY(I)                                                  DRKG2060
      GOTO 9                                                            DRKG2070
C                                                                       DRKG2080
C     COMPUTATION OF TEST VALUE DELT                                    DRKG2090
   23 DELT=0.D0                                                         DRKG2100
      DO 24 I=1,NDIM                                                    DRKG2110
   24 DELT=DELT+AUX(8,I)*DABS(AUX(4,I)-Y(I))                            DRKG2120
      IF(DELT-PRMT(4))28,28,25                                          DRKG2130
C                                                                       DRKG2140
C     ERROR IS TOO GREAT                                                DRKG2150
   25 IF(IHLF-10)26,36,36                                               DRKG2160
   26 DO 27 I=1,NDIM                                                    DRKG2170
   27 AUX(4,I)=AUX(5,I)                                                 DRKG2180
      ISTEP=ISTEP+ISTEP-4                                               DRKG2190
      X=X-H                                                             DRKG2200
      IEND=0                                                            DRKG2210
      GOTO 18                                                           DRKG2220
C                                                                       DRKG2230
C     RESULT VALUES ARE GOOD                                            DRKG2240
   28 CALL FCT(X,Y,DERY)                                                DRKG2250
      DO 29 I=1,NDIM                                                    DRKG2260
      AUX(1,I)=Y(I)                                                     DRKG2270
      AUX(2,I)=DERY(I)                                                  DRKG2280
      AUX(3,I)=AUX(6,I)                                                 DRKG2290
      Y(I)=AUX(5,I)                                                     DRKG2300
   29 DERY(I)=AUX(7,I)                                                  DRKG2310
      CALL OUTP(X-H,Y,DERY,IHLF,NDIM,PRMT)                              DRKG2320
      IF(PRMT(5))40,30,40                                               DRKG2330
   30 DO 31 I=1,NDIM                                                    DRKG2340
      Y(I)=AUX(1,I)                                                     DRKG2350
   31 DERY(I)=AUX(2,I)                                                  DRKG2360
      IREC=IHLF                                                         DRKG2370
      IF(IEND)32,32,39                                                  DRKG2380
C                                                                       DRKG2390
C     INCREMENT GETS DOUBLED                                            DRKG2400
   32 IHLF=IHLF-1                                                       DRKG2410
      ISTEP=ISTEP/2                                                     DRKG2420
      H=H+H                                                             DRKG2430
      IF(IHLF)4,33,33                                                   DRKG2440
   33 IMOD=ISTEP/2                                                      DRKG2450
      IF(ISTEP-IMOD-IMOD)4,34,4                                         DRKG2460
   34 IF(DELT-.02D0*PRMT(4))35,35,4                                     DRKG2470
   35 IHLF=IHLF-1                                                       DRKG2480
      ISTEP=ISTEP/2                                                     DRKG2490
      H=H+H                                                             DRKG2500
      GOTO 4                                                            DRKG2510
C                                                                       DRKG2520
C                                                                       DRKG2530
C     RETURNS TO CALLING PROGRAM                                        DRKG2540
   36 IHLF=11                                                           DRKG2550
      CALL FCT(X,Y,DERY)                                                DRKG2560
      GOTO 39                                                           DRKG2570
   37 IHLF=12                                                           DRKG2580
      GOTO 39                                                           DRKG2590
   38 IHLF=13                                                           DRKG2600
   39 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                DRKG2610
   40 RETURN                                                            DRKG2620
      END                                                               DRKG2630
```

Subroutines HPCG and DHPCG

These subroutines use Hamming's modified predictor-corrector method for the solution of general initial-value problems.

The purpose of Hamming's modified predictor-corrector method is to obtain an approximate solution of a general system of first-order ordinary differential equations with given initial values. It is a stable fourth-order integration procedure that requires the evaluation of the right-hand side of the system only two times per step. This is a great advantage compared with other methods of the same order of accuracy, especially the Runge-Kutta method, which requires the evaluation of the right-hand side four times per step. Another advantage is that at each step the calculation procedure gives an estimate for the local truncation error; thus the procedure is able, without a significant amount of calculation time, to choose and change the step size h. On the other hand, Hamming's predictor-corrector method itself is not self-starting; that is, the functional values at a single previous point are not enough to get the functional values ahead. Therefore, to obtain the starting values, a special Runge-Kutta procedure followed by one iteration step is added to the predictor-corrector method.

Given the general system of first-order ordinary differential equations:

$$y_1' = \frac{dy_1}{dx} = f_1(x, y_1, y_2, \ldots, y_n)$$

$$y_2' = \frac{dy_2}{dx} = f_2(x, y_1, y_2, \ldots, y_n)$$

$$\cdots\cdots\cdots\cdots$$

$$y_n' = \frac{dy_n}{dx} = f_n(x, y_1, y_2, \ldots, y_n)$$

and the initial values:

$$y_1(x_0) = y_{1,0},\ y_2(x_0) = y_{2,0},\ \ldots,\ y_n(x_0) = y_{n,0}$$

and using the following vector notations:

$$Y(x) = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_n(x) \end{pmatrix},\quad F(x,Y) = \begin{pmatrix} f_1(x,Y) \\ f_2(x,Y) \\ \vdots \\ f_n(x,Y) \end{pmatrix},\quad Y_0 = \begin{pmatrix} y_{1,0} \\ y_{2,0} \\ \vdots \\ y_{n,0} \end{pmatrix}$$

where Y, F, and Y_0 are column vectors, the given problem appears as follows:

$$Y' = \frac{dY}{dx} = F(x, Y) \text{ with } Y(x_0) = Y_0$$

For stability purposes, the modification by Hamming of Milne's classical modified predictor-corrector method is preferred. Thus, knowing the results at the equidistant points x_{j-3}, x_{j-2}, x_{j-1} and x_j, the results at point $x_{j+1} = x_j + h$ are computed by the formulas below.

Predictor: $$P_{j+1} = Y_{j-3} + \frac{4h}{3}(2Y_j' - Y_{j-1}' + 2Y_{j-2}') \quad (1)$$

Modifier: $$M_{j+1} = P_{j+1} - \frac{112}{121}(P_j - C_j);$$

$$M_{j+1}' = F(x_{j+1}, M_{j+1}) \quad (2), (3)$$

Corrector: $$C_{j+1} = \frac{1}{8}\left[9Y_j - Y_{j-2} + 3h(M_{j+1}' + 2Y_j' - Y_{j-1}')\right] \quad (4)$$

Final value: $$Y_{j+1} = C_{j+1} + \frac{9}{121}(P_{j+1} - C_{j+1}) \quad (5)$$

where Y, Y', P, M, M', F and C are column vectors with n components. Formulas (1) and (4) have local truncation errors:

$$T_1 = \frac{14}{45} h^5 Y^{(5)}(\xi_1) \text{ with } \xi_1 \in (x_{j-3}, x_{j+1})$$

and

$$T_2 = -\frac{1}{40} h^5 Y^{(5)}(\xi_2) \text{ with } \xi_2 \in (x_{j-2}, x_{j+1})$$

respectively, such that

$$C_{j+1} - P_{j+1} = \frac{121}{360} h^5 Y^{(5)}(\xi) \quad \left[\xi \in (x_{j-3}, x_{j+1})\right]$$

Assuming that $Y^{(5)}(x)$ does not vary to any great extent in the interval (x_{j-3}, x_{j+1}), it follows that:

$$T_2 \approx \frac{9}{121} (P_{j+1} - C_{j+1})$$

This formula shows that the components of column vector $P_{j+1} - C_{j+1}$ are measures for the local truncation errors in the components of column vector Y_{j+1}, and therefore control of accuracy and adjustment of step size h can be done by generating the following test value:

$$\delta = \sum_{i=1}^{n} a_i \cdot |p_{j+1,i} - c_{j+1,i}| \qquad (6)$$

where the coefficients a_i $(i = 1, 2, \ldots, n)$ are error-weights specified in the input of the procedure.

If δ is greater than a given tolerance ϵ†, increment h is halved and the procedure computes $Y_{j+1/2}$ -- that is, $Y(x_j + \frac{h}{2})$ -- after having interpolated $Y_{j-1/2} = Y(x_j - \frac{h}{2})$ and $Y_{j-3/2} = Y(x_j - \frac{3}{2}h)$, with previous increment h, using the following sixth-order interpolation formulas):

$$Y_{j-1/2} = \frac{1}{256} (80Y_j + 135Y_{j-1} + 40Y_{j-2} + Y_{j-3}) + \frac{h}{2} \cdot \frac{15}{128} (-Y'_j + 6Y'_{j-1} + Y'_{j-2}) \qquad (7)$$

$$Y_{j-3/2} = \frac{1}{256} (12Y_j + 135Y_{j-1} + 108Y_{j-2} + Y_{j-3}) + \frac{h}{2} \cdot \frac{3}{128} (-Y'_j - 18Y'_{j-1} + 9Y'_{j-2}) \qquad (8)$$

If δ is less than ϵ, the result Y_{j+1} is assumed to be correct and is handed, together with x_{j+1} and the vector of derivatives $Y'_{j+1} = F(x_{j+1}, Y_{j+1})$, to a user-supplied output subroutine.

†Numerical experience seems to show that the procedure does not exceed a global relative error approximately equal to ϵ.

If δ is less than $\epsilon/50$, the next step is carried out with the doubled increment; however, care is taken in the procedure that the increment never gets greater than the increment h specified as an input parameter, and further, that among the output are all points $x_0 + j \cdot h$ (where $j = 0, 1, 2, \ldots$, and h is the input step size) which are situated between the lower and the upper bound of the integration interval.

Changing the step size by halving or doubling requires changing of $P_j - C_j$ or $P_{j+1} - C_{j+1}$, respectively. Using the following interpolation formula:

$$Y_j = Y_{j-3} + \frac{3}{8} h(Y'_{j-3} + 3Y'_{j-2} + 3Y'_{j-1} + Y'_j) - \frac{3}{80} h^5 Y^{(5)}(\xi_3) \left[\xi_3 \epsilon (x_{j-3}, x_j)\right]$$

and assuming that $Y^{(5)}(x)$ does not vary to any great extent in this interval, $P_j - C_j$ can be written as:

$$P_j - C_j \approx \frac{242}{27} (Y_j - Y_{j-3}) - \frac{121}{36} h(Y'_j + 3Y'_{j-1} + 3Y'_{j-2} + Y'_{j-3})$$

When halving increment h, this formula becomes:

$$P_j - C_j \approx \frac{242}{27} (Y_j - Y_{j-3/2}) - \frac{121}{36} \cdot \frac{h}{2} (Y'_j + 3Y'_{j-1/2} + 3Y'_{j-1} + Y'_{j-3/2}) \qquad (9)$$

and when doubling:

$$P_{j+1} - C_{j+1} \approx \frac{242}{27} (Y_{j+1} - Y_{j-5}) - \frac{121}{36} \cdot 2h(Y'_{j+1} + 3Y'_{j-1} + 3Y'_{j-3} + Y'_{j-5}) \qquad (10)$$

Starting Hamming's modified predictor-corrector method requires the functional and derivative values at four preceding equidistant points; that is, x_0, x_1, x_2 and x_3. The values Y_0 and $Y'_0 = F(x_0, Y_0)$ are specified by input. For computation of Y_1, Y'_1, Y_2, Y'_2, Y_3 and Y'_3 and for adjustment of the step size h to accuracy requirements, a special Runge-Kutta procedure suggested by Ralston is used. Starting at x_j, result values at point $x_{j+1} = x_j + h$ are computed using the following formulas:

$$K_1 = h \cdot Y'_j \tag{11}$$

$$k_2 = h \cdot F(x_j + 0.4h,\ Y_j + 0.4K_1) \tag{12}$$

$$K_3 = h \cdot F(x_j + 0.45573725421878943h,\ Y_j + 0.29697760924775360\ K_1 + 0.15875964497103583\ K_2) \tag{13}$$

$$K_4 = h \cdot F(x_j + h,\ Y_j + 0.21810038822592047\ K_1 - 3.0509651486929308\ K_2 + 3.8328647604670103\ K_3) \tag{14}$$

$$Y_{j+1} = Y_j + 0.17476028226269037\ K_1 - 0.55148066287873294\ K_2 + 1.2055355993965235\ K_3 + 0.17118478121951903\ K_4 \tag{15}$$

where Y_j, Y_{j+1}, K_1, K_2, K_3 and K_4 are all column vectors with n components.

These formulas are not very stable, but this does not matter because they are used only in three successive steps (j = 0, 1, 2). On the other hand, they have the smallest bound of truncation error of all fourth-order Runge-Kutta procedures. Therefore they are best suited to start other integration methods which are not self-starting.

For initial control of accuracy and adjustment of step size h, in starting the procedure an approximation for $Y_2 = Y(x_0 + h)$, named $Y_2^{(1)}$, is computed using the step size h, and then an approximation named $Y_2^{(2)}$, using the step size h/2 twice. From these two approximations a test value for accuracy is generated in the following way:

$$\delta = \frac{1}{15} \sum_{i=1}^{n} a_i \cdot \left| y_{2,i}^{(1)} - y_{2,i}^{(2)} \right| \tag{16}$$

If δ is greater than ϵ, increment h is halved, and the procedure starts again at point x_0.

If δ is less than ϵ, the results $Y_1^{(2)} = Y(x_0 + \frac{h}{2})$ and $Y_2^{(2)} = Y(x_0 + h)$ are assumed to be correct, and a third step follows, which computes the results at point $x_0 + \frac{3}{2}h$ -- that is, $Y_3 = Y(x_0 + \frac{3}{2}h)$. The step size h/2 of these three steps is handed as initial step size h to the predictor-corrector method.

It is very important that the starting values be as accurate as possible, because errors in these starting values may increase during the following predictor-corrector procedure. Therefore the starting values computed by the Runge-Kutta method are refined by one iteration step using the following fourth-order interpolation formulas:

$$Y_1 = Y_0 + \frac{h}{24}(9\ Y'_0 + 19\ Y'_1 - 5\ Y'_2 + Y'_3) \tag{17}$$

$$Y_2 = Y_0 + \frac{h}{3}(Y'_0 + 4\ Y'_1 + Y'_2) \tag{18}$$

$$Y_3 = Y_0 + \frac{3h}{8}(Y'_0 + 3\ Y'_1 + 3\ Y'_2 + Y'_3) \tag{19}$$

Use of these must be considered an iteration procedure; that is, first the result values of the Runge-Kutta method are used in the right-hand side of formula (17) to compute a refined Y_1. After computing the refined $Y'_1 = F(x_1, Y_1)$, the refined vector Y_2 is generated using formula (18). Finally, the refined $Y'_2 = F(x_2, Y_2)$ is used, together with the other values, in the right-hand side of formula (19) to compute the refined vector Y_3. During this iteration, the refined data sets x_j, Y_j, Y'_j are handed, together with the number of bisections of the initial step size (specified by input), to the output subroutine.

It can be shown that, using this iterative procedure, the initial column vector $P_3 - C_3$ used in formula (2) for j=3 is equal to zero.

The entire input of the procedure is:

1. Lower and upper bound of the integration interval, initial step size h of the independent variable, and upper bound ϵ of the local truncation error
2. Initial values Y_0 of the dependent variables and weights a_i (i = 1, 2, ..., n) for the local truncation errors in each component of the dependent variables
3. The number n of differential equations in the system
4. As external subroutine subprograms, the computation of the right-hand side of the system of differential equations; for flexibility in output, an output subroutine
5. An auxiliary storage array named AUX with 16 rows and n columns

Output is done in the following way. If a set of approximations to the dependent variables Y(x) is found to be of sufficient accuracy, it is handed -- together with x, the derivative Y' (x), the number of bisections of the initial increment, the number of differential equations, the lower and upper bound of the interval, the initial step size, error bound ε, and a parameter for terminating subroutine HPCG or DHPCG -- to the output subroutine. By means of this output subroutine, the user has the opportunity to choose his own output format, to handle the output values as he wants, to change the upper error bound, and to terminate subroutine HPCG or DHPCG at any output point. In particular, the user is able to drop the output of some intermediate points, printing only the result values at the special points $x_0 + j \cdot h$(where j = 0, 1, 2, . . . , and h is the initial step size specified by input). The user may also perform intermediate computations using the integration results before continuing the process.

For better understanding of the flowchart and of the FORTRAN program, the allocation of special intermediate-result vectors within the storage array AUX is given in Figure 33.

function vector	$Y_{j-6} = Y(x_{j-6})$	1. row (AUX (1) in flowchart)
function vector	$Y_{j-5} = Y(x_{j-5})$	2. row (AUX (2) in flowchart)
function vector	$Y_{j-4} = Y(x_{j-4})$	3. row (AUX (3) in flowchart)
function vector	$Y_{j-3} = Y(x_{j-3})$	4. row (AUX (4) in flowchart)
function vector	$Y_{j-2} = Y(x_{j-2})$	5. row (AUX (5) in flowchart)
function vector	$Y_{j-1} = Y(x_{j-1})$	6. row (AUX (6) in flowchart)
function vector	$Y_j = Y(x_j)$	7. row (AUX (7) in flowchart)
derivative vector	$Y'_{j-6} = Y'(x_{j-6})$	8. row (AUX (8) in flowchart)
derivative vector	$Y'_{j-5} = Y'(x_{j-5})$	9. row (AUX (9) in flowchart)
derivative vector	$Y'_{j-4} = Y'(x_{j-4})$	10. row (AUX (10) in flowchart)
derivative vector	$Y'_{j-3} = Y'(x_{j-3})$	11. row (AUX (11) in flowchart)
derivative vector	$Y'_{j-2} = Y'(x_{j-2})$	12. row (AUX (12) in flowchart)
derivative vector	$Y'_{j-1} = Y'(x_{j-1})$	13. row (AUX (13) in flowchart)
derivative vector	$Y'_j = Y'(x_j)$	14. row (AUX (14) in flowchart)
vector of error weights		15. row (AUX (15) in flowchart)
vector $P_j - C_j$		16. row (AUX (16) in flowchart)

Figure 33. Allocation of special intermediate-result vectors within the special array AUX

For reference see:

(1) A. Ralston/ H.S. Wilf, Mathematical Methods for Digital Computers, Wiley, New York/London, 1960, pp. 95 - 109.

(2) A. Ralston, "Runge-Kutta Methods with Minimum Error Bounds", MTAC, vol. 16, no. 80 (1962), pp. 431 - 437.


```
C                                                                       HPCG  10
C                                                                       HPCG  20
C     ..................................................................HPCG  30
C
C        SUBROUTINE HPCG                                                HPCG  40
C                                                                       HPCG  50
C        PURPOSE                                                        HPCG  60
C           TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY GENERAL           HPCG  70
C           DIFFERENTIAL EQUATIONS WITH GIVEN INITIAL VALUES.           HPCG  80
C                                                                       HPCG  90
C        USAGE                                                          HPCG 100
C           CALL HPCG (PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)              HPCG 110
C           PARAMETERS FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.      HPCG 120
C                                                                       HPCG 130
C        DESCRIPTION OF PARAMETERS                                      HPCG 140
C                                                                       HPCG 150
C           PRMT    - AN INPUT AND OUTPUT VECTOR WITH DIMENSION GREATER HPCG 160
C                     OR EQUAL TO 5, WHICH SPECIFIES THE PARAMETERS OF  HPCG 170
C                     THE INTERVAL AND OF ACCURACY AND WHICH SERVES FOR HPCG 180
C                     COMMUNICATION BETWEEN OUTPUT SUBROUTINE (FURNISHED HPCG 190
C                     BY THE USER) AND SUBROUTINE HPCG. EXCEPT PRMT(5)  HPCG 200
C                     THE COMPONENTS ARE NOT DESTROYED BY SUBROUTINE    HPCG 210
C                     HPCG AND THEY ARE                                 HPCG 220
C           PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),               HPCG 230
C           PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),               HPCG 240
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE      HPCG 250
C                     (INPUT),                                          HPCG 260
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS    HPCG 270
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.      HPCG 280
C                     IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE    HPCG 290
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.HPCG 300
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS       HPCG 310
C                     OUTPUT SUBROUTINE.                                HPCG 320
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE HPCG INITIALIZES    HPCG 330
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE         HPCG 340
C                     SUBROUTINE HPCG AT ANY OUTPUT POINT, HE HAS TO    HPCG 350
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE HPCG 360
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE       HPCG 370
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER      HPCG 380
C                     THAN 5. HOWEVER SUBROUTINE HPCG DOES NOT REQUIRE  HPCG 390
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL  HPCG 400
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM     HPCG 410
C                     (CALLING HPCG) WHICH ARE OBTAINED BY SPECIAL      HPCG 420
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.HPCG 430
C           Y       - INPUT VECTOR OF INITIAL VALUES.  (DESTROYED)      HPCG 440
C                     LATERON Y IS THE RESULTING VECTOR OF DEPENDENT    HPCG 450
C                     VARIABLES COMPUTED AT INTERMEDIATE POINTS X.      HPCG 460
C           DERY    - INPUT VECTOR OF ERROR WEIGHTS.  (DESTROYED)       HPCG 470
C                     THE SUM OF ITS COMPONENTS MUST BE EQUAL TO 1.     HPCG 480
C                     LATERON DERY IS THE VECTOR OF DERIVATIVES, WHICH  HPCG 490
C                     BELONG TO FUNCTION VALUES Y AT A POINT X.         HPCG 500
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF     HPCG 510
C                     EQUATIONS IN THE SYSTEM.                          HPCG 520
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF    HPCG 530
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS HPCG 540
C                     GREATER THAN 10, SUBROUTINE HPCG RETURNS WITH     HPCG 550
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.          HPCG 560
C                     ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE  HPCG 570
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-HPCG 580
C                     PRMT(1)) RESPECTIVELY.                            HPCG 590
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT       HPCG 600
C                     COMPUTES THE RIGHT HAND SIDES DERY OF THE SYSTEM  HPCG 610
C                     TO GIVEN VALUES OF X AND Y. ITS PARAMETER LIST    HPCG 620
C                     MUST BE X,Y,DERY. THE SUBROUTINE SHOULD NOT       HPCG 630
C                     DESTROY X AND Y.                                  HPCG 640
C           OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.   HPCG 650
C                     ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.HPCG 660
C                     NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,   HPCG 670
C                     PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY         HPCG 680
C                     SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO.HPCG 690
C                     SUBROUTINE HPCG IS TERMINATED.                    HPCG 700
C           AUX     - AN AUXILIARY STORAGE ARRAY WITH 16 ROWS AND NDIM  HPCG 710
C                     COLUMNS.                                          HPCG 720
C                                                                       HPCG 730
C        REMARKS                                                        HPCG 740
C           THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF HPCG 750
C           (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE    HPCG 760
C               NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE   HPCG 770
C               IHLF=11),                                               HPCG 780
C           (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN       HPCG 790
C               (ERROR MESSAGES IHLF=12 OR IHLF=13),                    HPCG 800
C           (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,       HPCG 810
C           (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.        HPCG 820
C                                                                       HPCG 830
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  HPCG 840
C           THE EXTERNAL SUBROUTINES FCT(X,Y,DERY) AND                  HPCG 850
C           OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.HPCG 860
C                                                                       HPCG 870
C        METHOD                                                         HPCG 880
C           EVALUATION IS DONE BY MEANS OF HAMMINGS MODIFIED PREDICTOR- HPCG 890
C           CORRECTOR METHOD. IT IS A FOURTH ORDER METHOD, USING 4      HPCG 900
C           PRECEDING POINTS FOR COMPUTATION OF A NEW VECTOR Y OF THE   HPCG 910
C           DEPENDENT VARIABLES.                                        HPCG 920
C           FOURTH ORDER RUNGE-KUTTA METHOD SUGGESTED BY RALSTON IS     HPCG 930
C           USED FOR ADJUSTMENT OF THE INITIAL INCREMENT AND FOR        HPCG 940
C           COMPUTATION OF STARTING VALUES.                             HPCG 950
C           SUBROUTINE HPCG AUTOMATICALLY ADJUSTS THE INCREMENT DURING  HPCG 960
C           THE WHOLE COMPUTATION BY HALVING OR DOUBLING.               HPCG 970
C           TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE     HPCG 980
C           MUST BE CODED BY THE USER.                                  HPCG 990
C           FOR REFERENCE, SEE                                          HPCG1000
C           (1) RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL          HPCG1010
C               COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.     HPCG1020
C           (2) RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS,HPCG1030
C               MTAC, VOL.16, ISS.80 (1962), PP.431-437.                HPCG1040
C                                                                       HPCG1050
C     ..................................................................HPCG1060
C                                                                       HPCG1070
      SUBROUTINE HPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)               HPCG1080
```

```
C                                                                       HPCG1090
C                                                                       HPCG1100
      DIMENSION PRMT(1),Y(1),DERY(1),AUX(16,1)                          HPCG1110
      N=1                                                               HPCG1120
      IHLF=0                                                            HPCG1130
      X=PRMT(1)                                                         HPCG1140
      H=PRMT(3)                                                         HPCG1150
      PRMT(5)=0.                                                        HPCG1160
      DO 1 I=1,NDIM                                                     HPCG1170
      AUX(16,I)=0.                                                      HPCG1180
      AUX(15,I)=DERY(I)                                                 HPCG1190
    1 AUX(1,I)=Y(I)                                                     HPCG1200
      IF(H*(PRMT(2)-X))3,2,4                                            HPCG1210
C                                                                       HPCG1220
C     ERROR RETURNS                                                     HPCG1230
    2 IHLF=12                                                           HPCG1240
      GOTO 4                                                            HPCG1250
    3 IHLF=13                                                           HPCG1260
C                                                                       HPCG1270
C     COMPUTATION OF DERY FOR STARTING VALUES                           HPCG1280
    4 CALL FCT(X,Y,DERY)                                                HPCG1290
C                                                                       HPCG1300
C     RECORDING OF STARTING VALUES                                      HPCG1310
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                HPCG1320
      IF(PRMT(5))6,5,6                                                  HPCG1330
    5 IF(IHLF)7,7,6                                                     HPCG1340
    6 RETURN                                                            HPCG1350
    7 DO 8 I=1,NDIM                                                     HPCG1360
    8 AUX(8,I)=DERY(I)                                                  HPCG1370
C                                                                       HPCG1380
C     COMPUTATION OF AUX(2,I)                                           HPCG1390
      ISW=1                                                             HPCG1400
      GOTO 100                                                          HPCG1410
C                                                                       HPCG1420
    9 X=X+H                                                             HPCG1430
      DO 10 I=1,NDIM                                                    HPCG1440
   10 AUX(2,I)=Y(I)                                                     HPCG1450
C                                                                       HPCG1460
C     INCREMENT H IS TESTED BY MEANS OF BISECTION                       HPCG1470
   11 IHLF=IHLF+1                                                       HPCG1480
      X=X-H                                                             HPCG1490
      DO 12 I=1,NDIM                                                    HPCG1500
   12 AUX(4,I)=AUX(2,I)                                                 HPCG1510
      H=.5*H                                                            HPCG1520
      N=1                                                               HPCG1530
      ISW=2                                                             HPCG1540
      GOTO 100                                                          HPCG1550
C                                                                       HPCG1560
   13 X=X+H                                                             HPCG1570
      CALL FCT(X,Y,DERY)                                                HPCG1580
      N=2                                                               HPCG1590
      DO 14 I=1,NDIM                                                    HPCG1600
      AUX(2,I)=Y(I)                                                     HPCG1610
   14 AUX(9,I)=DERY(I)                                                  HPCG1620
      ISW=3                                                             HPCG1630
      GOTO 100                                                          HPCG1640
C                                                                       HPCG1650
C     COMPUTATION OF TEST VALUE DELT                                    HPCG1660
   15 DELT=0.                                                           HPCG1670
      DO 16 I=1,NDIM                                                    HPCG1680
   16 DELT=DELT+AUX(15,I)*ABS(Y(I)-AUX(4,I))                            HPCG1690
      DELT=.0666667*DELT                                                HPCG1700
      IF(DELT-PRMT(4))19,19,17                                          HPCG1710
   17 IF(IHLF-10)11,18,18                                               HPCG1720
C                                                                       HPCG1730
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.      HPCG1740
   18 IHLF=11                                                           HPCG1750
      X=X+H                                                             HPCG1760
      GOTO 4                                                            HPCG1770
C                                                                       HPCG1780
C     THERE IS SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS.     HPCG1790
   19 X=X+H                                                             HPCG1800
      CALL FCT(X,Y,DERY)                                                HPCG1810
      DO 20 I=1,NDIM                                                    HPCG1820
      AUX(3,I)=Y(I)                                                     HPCG1830
   20 AUX(10,I)=DERY(I)                                                 HPCG1840
      N=3                                                               HPCG1850
      ISW=4                                                             HPCG1860
      GOTO 100                                                          HPCG1870
C                                                                       HPCG1880
   21 N=1                                                               HPCG1890
      X=X+H                                                             HPCG1900
      CALL FCT(X,Y,DERY)                                                HPCG1910
      X=PRMT(1)                                                         HPCG1920
      DO 22 I=1,NDIM                                                    HPCG1930
      AUX(11,I)=DERY(I)                                                 HPCG1940
   22CY(I)=AUX(1,I)+H*(.375*AUX(8,I)+.7916667*AUX(9,I)                  HPCG1950
     1-.2083333*AUX(10,I)+.04166667*DERY(I))                            HPCG1960
   23 X=X+H                                                             HPCG1970
      N=N+1                                                             HPCG1980
      CALL FCT(X,Y,DERY)                                                HPCG1990
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                HPCG2000
      IF(PRMT(5))6,24,6                                                 HPCG2010
   24 IF(N-4)25,200,200                                                 HPCG2020
   25 DO 26 I=1,NDIM                                                    HPCG2030
      AUX(N,I)=Y(I)                                                     HPCG2040
   26 AUX(N+7,I)=DERY(I)                                                HPCG2050
      IF(N-3)27,29,200                                                  HPCG2060
C                                                                       HPCG2070
   27 DO 28 I=1,NDIM                                                    HPCG2080
      DELT=AUX(9,I)+AUX(9,I)                                            HPCG2090
      DELT=DELT+DELT                                                    HPCG2100
   28 Y(I)=AUX(1,I)+.3333333*H*(AUX(8,I)+DELT+AUX(10,I))                HPCG2110
      GOTO 23                                                           HPCG2120
C                                                                       HPCG2130
   29 DO 30 I=1,NDIM                                                    HPCG2140
      DELT=AUX(9,I)+AUX(10,I)                                           HPCG2150
      DELT=DELT+DELT+DELT                                               HPCG2160
   30 Y(I)=AUX(1,I)+.375*H*(AUX(8,I)+DELT+AUX(11,I))                    HPCG2170
      GOTO 23                                                           HPCG2180
C                                                                       HPCG2190
C     THE FOLLOWING PART OF SUBROUTINE HPCG COMPUTES BY MEANS OF        HPCG2200
C     RUNGE-KUTTA METHOD STARTING VALUES FOR THE NOT SELF-STARTING      HPCG2210
C     PREDICTOR-CORRECTOR METHOD.                                       HPCG2220
  100 DO 101 I=1,NDIM                                                   HPCG2230
      Z=H*AUX(N+7,I)                                                    HPCG2240
      AUX(5,I)=Z                                                        HPCG2250
  101 Y(I)=AUX(N,I)+.4*Z                                                HPCG2260
C     Z IS AN AUXILIARY STORAGE LOCATION                                HPCG2270
C                                                                       HPCG2280
      Z=X+.4*H                                                          HPCG2290
      CALL FCT(Z,Y,DERY)                                                HPCG2300
      DO 102 I=1,NDIM                                                   HPCG2310
      Z=H*DERY(I)                                                       HPCG2320
      AUX(6,I)=Z                                                        HPCG2330
  102 Y(I)=AUX(N,I)+.2969776*AUX(5,I)+.1587596*Z                        HPCG2340
C                                                                       HPCG2350
      Z=X+.4557372*H                                                    HPCG2360
      CALL FCT(Z,Y,DERY)                                                HPCG2370
      DO 103 I=1,NDIM                                                   HPCG2380
      Z=H*DERY(I)                                                       HPCG2390
      AUX(7,I)=Z                                                        HPCG2400
  103 Y(I)=AUX(N,I)+.2181004*AUX(5,I)-3.050965*AUX(6,I)+3.832865*Z      HPCG2410
C                                                                       HPCG2420
      Z=X+H                                                             HPCG2430
      CALL FCT(Z,Y,DERY)                                                HPCG2440
      DO 104 I=1,NDIM                                                   HPCG2450
  104OY(I)=AUX(N,I)+.1747603*AUX(5,I)-.5514807*AUX(6,I)                 HPCG2460
     1+1.205536*AUX(7,I)+.1711848*H*DERY(I)                             HPCG2470
      GOTO(9,13,15,21),ISW                                              HPCG2480
C                                                                       HPCG2490
C     POSSIBLE BREAK-POINT FOR LINKAGE                                  HPCG2500
C                                                                       HPCG2510
C     STARTING VALUES ARE COMPUTED.                                     HPCG2520
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.           HPCG2530
  200 ISTEP=3                                                           HPCG2540
  201 IF(N-8)204,202,204                                                HPCG2550
C                                                                       HPCG2560
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS      HPCG2570
  202 DO 203 N=2,7                                                      HPCG2580
      DO 203 I=1,NDIM                                                   HPCG2590
      AUX(N-1,I)=AUX(N,I)                                               HPCG2600
  203 AUX(N+6,I)=AUX(N+7,I)                                             HPCG2610
      N=7                                                               HPCG2620
C                                                                       HPCG2630
C     N LESS THAN 8 CAUSES N+1 TO GET N                                 HPCG2640
  204 N=N+1                                                             HPCG2650
C                                                                       HPCG2660
C     COMPUTATION OF NEXT VECTOR Y                                      HPCG2670
      DO 205 I=1,NDIM                                                   HPCG2680
      AUX(N-1,I)=Y(I)                                                   HPCG2690
  205 AUX(N+6,I)=DERY(I)                                                HPCG2700
      X=X+H                                                             HPCG2710
  206 ISTEP=ISTEP+1                                                     HPCG2720
      DO 207 I=1,NDIM                                                   HPCG2730
     0DELT=AUX(N-4,I)+1.333333*H*(AUX(N+6,I)+AUX(N+6,I)-AUX(N+5,I)+     HPCG2740
     1AUX(N+4,I)+AUX(N+4,I))                                            HPCG2750
      Y(I)=DELT-.9256198*AUX(16,I)                                      HPCG2760
  207 AUX(16,I)=DELT                                                    HPCG2770
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX, MODIFIED PREDICTOR   HPCG2780
C     IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.               HPCG2790
C                                                                       HPCG2800
      CALL FCT(X,Y,DERY)                                                HPCG2810
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY             HPCG2820
C                                                                       HPCG2830
      DO 208 I=1,NDIM                                                   HPCG2840
     0DELT=.125*(9.*AUX(N-1,I)-AUX(N-3,I)+3.*H*(DERY(I)+AUX(N+6,I)+     HPCG2850
     1AUX(N+6,I)-AUX(N+5,I)))                                           HPCG2860
      AUX(16,I)=AUX(16,I)-DELT                                          HPCG2870
  208 Y(I)=DELT+.07438017*AUX(16,I)                                     HPCG2880
C                                                                       HPCG2890
C     TEST WHETHER H MUST BE HALVED OR DOUBLED                          HPCG2900
      DELT=0.                                                           HPCG2910
      DO 209 I=1,NDIM                                                   HPCG2920
  209 DELT=DELT+AUX(15,I)*ABS(AUX(15,I))                                HPCG2930
      IF(DELT-PRMT(4))210,222,222                                       HPCG2940
C                                                                       HPCG2950
C     H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.                   HPCG2960
  210 CALL FCT(X,Y,DERY)                                                HPCG2970
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                HPCG2980
      IF(PRMT(5))212,211,212                                            HPCG2990
  211 IF(IHLF-11)213,212,212                                            HPCG3000
  212 RETURN                                                            HPCG3010
  213 IF(H*(X-PRMT(2)))214,212,212                                      HPCG3020
  214 IF(ABS(X-PRMT(2))-.1*ABS(H))212,215,215                           HPCG3030
  215 IF(DELT-.02*PRMT(4))216,216,201                                   HPCG3040
C                                                                       HPCG3050
C                                                                       HPCG3060
C     H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE         HPCG3070
C     AVAILABLE                                                         HPCG3080
  216 IF(IHLF)201,201,217                                               HPCG3090
  217 IF(N-7)201,218,218                                                HPCG3100
  218 IF(ISTEP-4)201,219,219                                            HPCG3110
  219 IMOD=ISTEP/2                                                      HPCG3120
      IF(ISTEP-IMOD-IMOD)201,220,201                                    HPCG3130
  220 H=H+H                                                             HPCG3140
      IHLF=IHLF-1                                                       HPCG3150
      ISTEP=0                                                           HPCG3160
      DO 221 I=1,NDIM                                                   HPCG3170
      AUX(N-1,I)=AUX(N-2,I)                                             HPCG3180
      AUX(N-2,I)=AUX(N-4,I)                                             HPCG3190
      AUX(N-3,I)=AUX(N-6,I)                                             HPCG3200
      AUX(N+6,I)=AUX(N+5,I)                                             HPCG3210
      AUX(N+5,I)=AUX(N+3,I)                                             HPCG3220
      AUX(N+4,I)=AUX(N+1,I)                                             HPCG3230
      DELT=AUX(N+6,I)+AUX(N+5,I)                                        HPCG3240
      DELT=DELT+DELT+DELT                                               HPCG3250
  221OAUX(16,I)=8.962963*(Y(I)-AUX(N-3,I))-3.361111*H*(DERY(I)+DELT     HPCG3260
     1+AUX(N+4,I))                                                      HPCG3270
      GOTO 201                                                          HPCG3280
C                                                                       HPCG3290
C                                                                       HPCG3300
C     H MUST BE HALVED                                                  HPCG3310
  222 IHLF=IHLF+1                                                       HPCG3320
      IF(IHLF-10)223,223,210                                            HPCG3330
  223 H=.5*H                                                            HPCG3340
      ISTEP=0                                                           HPCG3350
      DO 224 I=1,NDIM                                                   HPCG3360
     0Y(I)=.00390625*(80.*AUX(N-1,I)+135.*AUX(N-2,I)+40.*AUX(N-3,I)+    HPCG3370
     1AUX(N-4,I))-.1171875*(AUX(N+6,I)-6.*AUX(N+5,I)-AUX(N+4,I))*H      HPCG3380
     0AUX(N-4,I)=.00390625*(12.*AUX(N-1,I)+135.*AUX(N-2,I)+             HPCG3390
     1108.*AUX(N-3,I)+AUX(N-4,I))-.0234375*(AUX(N+6,I)*18.*AUX(N+5,I)-  HPCG3400
     29.*AUX(N+4,I))*H                                                  HPCG3410
      AUX(N-3,I)=AUX(N-2,I)                                             HPCG3420
  224 AUX(N+4,I)=AUX(N+5,I)                                             HPCG3430
      X=X-H                                                             HPCG3440
      DELT=X-(H+H)                                                      HPCG3450
      CALL FCT(DELT,Y,DERY)                                             HPCG3460
      DO 225 I=1,NDIM                                                   HPCG3470
      AUX(N-2,I)=Y(I)                                                   HPCG3480
      AUX(N+5,I)=DERY(I)                                                HPCG3490
  225 Y(I)=AUX(N-4,I)                                                   HPCG3500
      DELT=DELT-(H+H)                                                   HPCG3510
      CALL FCT(DELT,Y,DERY)                                             HPCG3520
      DO 226 I=1,NDIM                                                   HPCG3530
      DELT=AUX(N+5,I)+AUX(N+4,I)                                        HPCG3540
      DELT=DELT+DELT+DELT                                               HPCG3550
     0AUX(16,I)=8.962963*(AUX(N-1,I)-Y(I))-3.361111*H*(AUX(N+6,I)+DELT  HPCG3560
     1+DERY(I))                                                         HPCG3570
  226 AUX(N+3,I)=DERY(I)                                                HPCG3580
      GOTO 206                                                          HPCG3590
      END                                                               HPCG3600
```

```
C                                                                       DHCG  10
C     ******************************************************************DHCG  20
C                                                                       DHCG  30
C        SUBROUTINE DHPCG                                               DHCG  40
C                                                                       DHCG  50
C        PURPOSE                                                        DHCG  60
C           TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY GENERAL           DHCG  70
C           DIFFERENTIAL EQUATIONS WITH GIVEN INITIAL VALUES.           DHCG  80
C                                                                       DHCG  90
C        USAGE                                                          DHCG 100
C           CALL DHPCG (PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)             DHCG 110
C           PARAMETERS FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.      DHCG 120
C                                                                       DHCG 130
C        DESCRIPTION OF PARAMETERS                                      DHCG 140
C           PRMT    - DOUBLE PRECISION INPUT AND OUTPUT VECTOR WITH     DHCG 150
C                     DIMENSION GREATER THAN OR EQUAL TO 5, WHICH       DHCG 160
C                     SPECIFIES THE PARAMETERS OF THE INTERVAL AND OF   DHCG 170
C                     ACCURACY AND WHICH SERVES FOR COMMUNICATION BETWEENDHCG 180
C                     OUTPUT SUBROUTINE (FURNISHED BY THE USER) AND     DHCG 190
C                     SUBROUTINE DHPCG. EXCEPT PRMT(5) THE COMPONENTS   DHCG 200
C                     ARE NOT DESTROYED BY SUBROUTINE DHPCG AND THEY ARE DHCG 210
C           PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),               DHCG 220
C           PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),               DHCG 230
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE      DHCG 240
C                     (INPUT),                                          DHCG 250
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS    DHCG 260
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.      DHCG 270
C                     IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE    DHCG 280
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.DHCG 290
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS       DHCG 300
C                     OUTPUT SUBROUTINE.                                DHCG 310
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE DHPCG INITIALIZES   DHCG 320
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE         DHCG 330
C                     SUBROUTINE DHPCG AT ANY OUTPUT POINT, HE HAS TO   DHCG 340
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE DHCG 350
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE       DHCG 360
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER      DHCG 370
C                     THAN 5. HOWEVER SUBROUTINE DHPCG DOES NOT REQUIRE DHCG 380
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL  DHCG 390
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM     DHCG 400
C                     (CALLING DHPCG) WHICH ARE OBTAINED BY SPECIAL     DHCG 410
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.DHCG 420
C           Y       - DOUBLE PRECISION INPUT VECTOR OF INITIAL VALUES   DHCG 430
C                     (DESTROYED). LATERON Y IS THE RESULTING VECTOR OF DHCG 440
C                     DEPENDENT VARIABLES COMPUTED AT INTERMEDIATE      DHCG 450
C                     POINTS X.                                         DHCG 460
C           DERY    - DOUBLE PRECISION INPUT VECTOR OF ERROR WEIGHTS    DHCG 470
C                     (DESTROYED). THE SUM OF ITS COMPONENTS MUST BE    DHCG 480
C                     EQUAL TO 1. LATERON DERY IS THE VECTOR OF         DHCG 490
C                     DERIVATIVES, WHICH BELONG TO FUNCTION VALUES Y AT DHCG 500
C                     INTERMEDIATE POINTS X.                            DHCG 510
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF     DHCG 520
C                     EQUATIONS IN THE SYSTEM.                          DHCG 530
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF    DHCG 540
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS DHCG 550
C                     GREATER THAN 10, SUBROUTINE DHPCG RETURNS WITH    DHCG 560
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.          DHCG 570
C                     ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE  DHCG 580
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-DHCG 590
C                     PRMT(1)) RESPECTIVELY.                            DHCG 600
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT       DHCG 610
C                     COMPUTES THE RIGHT HAND SIDES DERY OF THE SYSTEM  DHCG 620
C                     TO GIVEN VALUES OF X AND Y. ITS PARAMETER LIST    DHCG 630
C                     MUST BE X,Y,DERY. THE SUBROUTINE SHOULD NOT       DHCG 640
C                     DESTROY X AND Y.                                  DHCG 650
C           OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.   DHCG 660
C                     ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.DHCG 670
C                     NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,   DHCG 680
C                     PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY         DHCG 690
C                     SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,DHCG 700
C                     SUBROUTINE DHPCG IS TERMINATED.                   DHCG 710
C           AUX     - DOUBLE PRECISION AUXILIARY STORAGE ARRAY WITH 16  DHCG 720
C                     ROWS AND NDIM COLUMNS.                            DHCG 730
C                                                                       DHCG 740
C        REMARKS                                                        DHCG 750
C           THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF DHCG 760
C           (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE    DHCG 770
C               NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE   DHCG 780
C               IHLF=11),                                               DHCG 790
C           (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN       DHCG 800
C               (ERROR MESSAGES IHLF=12 OR IHLF=13),                    DHCG 810
C           (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,       DHCG 820
C           (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.        DHCG 830
C                                                                       DHCG 840
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DHCG 850
C           THE EXTERNAL SUBROUTINES FCT(X,Y,DERY) AND                  DHCG 860
C           OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.DHCG 870
C                                                                       DHCG 880
C        METHOD                                                         DHCG 890
C           EVALUATION IS DONE BY MEANS OF HAMMINGS MODIFIED PREDICTOR- DHCG 900
C           CORRECTOR METHOD. IT IS A FOURTH ORDER METHOD, USING 4      DHCG 910
C           PRECEEDING POINTS FOR COMPUTATION OF A NEW VECTOR Y OF THE  DHCG 920
C           DEPENDENT VARIABLES.                                        DHCG 930
C           FOURTH ORDER RUNGE-KUTTA METHOD SUGGESTED BY RALSTON IS     DHCG 940
C           USED FOR ADJUSTMENT OF THE INITIAL INCREMENT AND FOR        DHCG 950
C           COMPUTATION OF STARTING VALUES.                             DHCG 960
C           SUBROUTINE DHPCG AUTOMATICALLY ADJUSTS THE INCREMENT DURING DHCG 970
C           THE WHOLE COMPUTATION BY HALVING OR DOUBLING.               DHCG 980
C           TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE     DHCG 990
C           MUST BE CODED BY THE USER.                                  DHCG1000
C           FOR REFERENCE, SEE                                          DHCG1010
C           (1)  RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL         DHCG1020
C                COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.    DHCG1030
C           (2)  RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS,DHCG1040
C                MTAC, VOL.16, ISS.80 (1962), PP.431-437.               DHCG1050
C                                                                       DHCG1060
C     ******************************************************************DHCG1070
C                                                                       DHCG1080
      SUBROUTINE DHPCG(PRMT,Y,DERY,NDIM,IHLF,FCT,OUTP,AUX)              DHCG1090
C                                                                       DHCG1100
C                                                                       DHCG1110
      DIMENSION PRMT(1),Y(1),DERY(1),AUX(16,1)                          DHCG1120
      DOUBLE PRECISION Y,DERY,AUX,PRMT,X,H,Z,DELT                       DHCG1130
      N=1                                                               DHCG1140
      IHLF=0                                                            DHCG1150
      X=PRMT(1)                                                         DHCG1160
      H=PRMT(3)                                                         DHCG1170
      PRMT(5)=0.D0                                                      DHCG1180
      DO 1 I=1,NDIM                                                     DHCG1190
      AUX(16,I)=0.D0                                                    DHCG1200
      AUX(15,I)=DERY(I)                                                 DHCG1210
    1 AUX(1,I)=Y(I)                                                     DHCG1220
      IF(H*(PRMT(2)-X))3,2,4                                            DHCG1230
C                                                                       DHCG1240
C     ERROR RETURNS                                                     DHCG1250
    2 IHLF=12                                                           DHCG1260
      GOTO 4                                                            DHCG1270
    3 IHLF=13                                                           DHCG1280
C                                                                       DHCG1290
C     COMPUTATION OF DERY FOR STARTING VALUES                           DHCG1300
    4 CALL FCT(X,Y,DERY)                                                DHCG1310
C                                                                       DHCG1320
C     RECORDING OF STARTING VALUES                                      DHCG1330
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                DHCG1340
      IF(PRMT(5))6,5,6                                                  DHCG1350
    5 IF(IHLF)7,7,6                                                     DHCG1360
    6 RETURN                                                            DHCG1370
    7 DO 8 I=1,NDIM                                                     DHCG1380
    8 AUX(8,I)=DERY(I)                                                  DHCG1390
C                                                                       DHCG1400
C     COMPUTATION OF AUX(2,I)                                           DHCG1410
      ISW=1                                                             DHCG1420
      GOTO 100                                                          DHCG1430
C                                                                       DHCG1440
    9 X=X+H                                                             DHCG1450
      DO 10 I=1,NDIM                                                    DHCG1460
   10 AUX(2,I)=Y(I)                                                     DHCG1470
C                                                                       DHCG1480
C     INCREMENT H IS TESTED BY MEANS OF BISECTION                       DHCG1490
   11 IHLF=IHLF+1                                                       DHCG1500
      X=X-H                                                             DHCG1510
      DO 12 I=1,NDIM                                                    DHCG1520
   12 AUX(4,I)=AUX(2,I)                                                 DHCG1530
      H=.5D0*H                                                          DHCG1540
      N=1                                                               DHCG1550
      ISW=2                                                             DHCG1560
      GOTO 100                                                          DHCG1570
C                                                                       DHCG1580
   13 X=X+H                                                             DHCG1590
      CALL FCT(X,Y,DERY)                                                DHCG1600
      N=2                                                               DHCG1610
      DO 14 I=1,NDIM                                                    DHCG1620
      AUX(2,I)=Y(I)                                                     DHCG1630
   14 AUX(9,I)=DERY(I)                                                  DHCG1640
      ISW=3                                                             DHCG1650
      GOTO 100                                                          DHCG1660
C                                                                       DHCG1670
C     COMPUTATION OF TEST VALUE DELT                                    DHCG1680
   15 DELT=0.D0                                                         DHCG1690
      DO 16 I=1,NDIM                                                    DHCG1700
   16 DELT=DELT+AUX(15,I)*DABS(Y(I)-AUX(4,I))                           DHCG1710
      DELT=.06666666666666666667D0*DELT                                 DHCG1720
      IF(DELT-PRMT(4))19,19,17                                          DHCG1730
   17 IF(IHLF-10)11,18,18                                               DHCG1740
C                                                                       DHCG1750
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.      DHCG1760
   18 IHLF=11                                                           DHCG1770
      X=X+H                                                             DHCG1780
      GOTO 4                                                            DHCG1790
C                                                                       DHCG1800
C     THERE IS SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS.     DHCG1810
   19 X=X+H                                                             DHCG1820
      CALL FCT(X,Y,DERY)                                                DHCG1830
      DO 20 I=1,NDIM                                                    DHCG1840
      AUX(3,I)=Y(I)                                                     DHCG1850
   20 AUX(10,I)=DERY(I)                                                 DHCG1860
      N=3                                                               DHCG1870
      ISW=4                                                             DHCG1880
      GOTO 100                                                          DHCG1890
C                                                                       DHCG1900
   21 N=1                                                               DHCG1910
      X=X+H                                                             DHCG1920
      CALL FCT(X,Y,DERY)                                                DHCG1930
      X=PRMT(1)                                                         DHCG1940
      DO 22 I=1,NDIM                                                    DHCG1950
      AUX(11,I)=DERY(I)                                                 DHCG1960
   22Y(I)=AUX(1,I)+H*(.375D0*AUX(8,I)+.79166666666666667D0*AUX(9,I)     DHCG1970
     1-.20833333333333333D0*AUX(10,I)+.04166666666666666667D0*DERY(I))  DHCG1980
   23 X=X+H                                                             DHCG1990
      N=N+1                                                             DHCG2000
      CALL FCT(X,Y,DERY)                                                DHCG2010
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                DHCG2020
      IF(PRMT(5))6,24,6                                                 DHCG2030
   24 IF(N-4)25,200,200                                                 DHCG2040
   25 DO 26 I=1,NDIM                                                    DHCG2050
      AUX(N,I)=Y(I)                                                     DHCG2060
   26 AUX(N+7,I)=DERY(I)                                                DHCG2070
      IF(N-3)27,29,200                                                  DHCG2080
C                                                                       DHCG2090
   27 DO 28 I=1,NDIM                                                    DHCG2100
      DELT=AUX(9,I)+AUX(9,I)                                            DHCG2110
      DELT=DELT+DELT                                                    DHCG2120
   28 Y(I)=AUX(1,I)+.33333333333333333D0*H*(AUX(8,I)+DELT+AUX(10,I))    DHCG2130
      GOTO 23                                                           DHCG2140
C                                                                       DHCG2150
   29 DO 30 I=1,NDIM                                                    DHCG2160
      DELT=AUX(9,I)+AUX(10,I)                                           DHCG2170
      DELT=DELT+DELT+DELT                                               DHCG2180
   30 Y(I)=AUX(1,I)+.375D0*H*(AUX(8,I)+DELT+AUX(11,I))                  DHCG2190
      GOTO 23                                                           DHCG2200
C                                                                       DHCG2210
C     THE FOLLOWING PART OF SUBROUTINE DHPCG COMPUTES BY MEANS OF       DHCG2220
C     RUNGE-KUTTA METHOD STARTING VALUES FOR THE NOT SELF-STARTING      DHCG2230
C     PREDICTOR-CORRECTOR METHOD.                                       DHCG2240
  100 DO 101 I=1,NDIM                                                   DHCG2250
      Z=H*AUX(N+7,I)                                                    DHCG2260
      AUX(5,I)=Z                                                        DHCG2270
  101 Y(I)=AUX(N,I)+.4D0*Z                                              DHCG2280
C     Z IS AN AUXILIARY STORAGE LOCATION                                DHCG2290
C                                                                       DHCG2300
      Z=X+.4D0*H                                                        DHCG2310
      CALL FCT(Z,Y,DERY)                                                DHCG2320
      DO 102 I=1,NDIM                                                   DHCG2330
      Z=H*DERY(I)                                                       DHCG2340
      AUX(6,I)=Z                                                        DHCG2350
  102 Y(I)=AUX(N,I)+.29697760924775360D0*AUX(5,I)+.15875964497103583D0*ZDHCG2360
C                                                                       DHCG2370
      Z=X+.45573725421878943D0*H                                        DHCG2380
      CALL FCT(Z,Y,DERY)                                                DHCG2390
      DO 103 I=1,NDIM                                                   DHCG2400
      Z=H*DERY(I)                                                       DHCG2410
      AUX(7,I)=Z                                                        DHCG2420
  103 Y(I)=AUX(N,I)+.21810038822592047D0*AUX(5,I)-3.05096514869293080D0* DHCG2430
     1AUX(6,I)+3.83286476046701030D0*Z                                  DHCG2440
C                                                                       DHCG2450
      Z=X+H                                                             DHCG2460
      CALL FCT(Z,Y,DERY)                                                DHCG2470
      DO 104 I=1,NDIM                                                   DHCG2480
 1040Y(I)=AUX(N,I)+.17476028226269037D0*AUX(5,I)-.55148066287873294D0* DHCG2490
     1AUX(6,I)+1.20553559939652350D0*AUX(7,I)+.17118478121951903D0*     DHCG2500
     2H*DERY(I)                                                         DHCG2510
      GOTO(9,13,15,21),ISW                                              DHCG2520
C                                                                       DHCG2530
C     POSSIBLE BREAK-POINT FOR LINKAGE                                  DHCG2540
C                                                                       DHCG2550
C     STARTING VALUES ARE COMPUTED.                                     DHCG2560
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.           DHCG2570
  200 ISTEP=3                                                           DHCG2580
```

```
  201 IF(N-8)204,202,204                                                DHCG2590
C                                                                       DHCG2600
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS      DHCG2610
  202 DO 203 N=2,7                                                      DHCG2620
      DO 203 I=1,NDIM                                                   DHCG2630
      AUX(N-1,I)=AUX(N,I)                                               DHCG2640
  203 AUX(N+6,I)=AUX(N+7,I)                                             DHCG2650
      N=7                                                               DHCG2660
C                                                                       DHCG2670
C     N LESS THAN 8 CAUSES N+1 TO GET N                                 DHCG2680
  204 N=N+1                                                             DHCG2690
C                                                                       DHCG2700
C     COMPUTATION OF NEXT VECTOR Y                                      DHCG2710
      DO 205 I=1,NDIM                                                   DHCG2720
      AUX(N-1,I)=Y(I)                                                   DHCG2730
  205 AUX(N+6,I)=DERY(I)                                                DHCG2740
      X=X+H                                                             DHCG2750
  206 ISTEP=ISTEP+1                                                     DHCG2760
      DO 207 I=1,NDIM                                                   DHCG2770
     CDELT=AUX(N-4,I)+(.3333333333333333D0*H*(AUX(N+6,I)+AUX(N+6,I)-    DHCG2780
     1AUX(N+5,I)+AUX(N+4,I)+AUX(N+4,I))                                 DHCG2790
      Y(I)=DELT-.9256198347107438D0*AUX(16,I)                           DHCG2800
  207 AUX(16,I)=DELT                                                    DHCG2810
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX, MODIFIED PREDICTOR   DHCG2820
C     IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.               DHCG2830
C                                                                       DHCG2840
      CALL FCT(X,Y,DERY)                                                DHCG2850
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY             DHCG2860
C                                                                       DHCG2870
      DO 208 I=1,NDIM                                                   DHCG2880
     0DELT=.125D0*(9.D0*AUX(N-1,I)-AUX(N-3,I)+3.D0*H*(DERY(I)+AUX(N+6,I)DHCG2890
     1+AUX(N+6,I)-AUX(N+5,I)))                                          DHCG2900
      AUX(16,I)=AUX(16,I)-DELT                                          DHCG2910
  208 Y(I)=DELT+.0743801652892562D0*AUX(16,I)                           DHCG2920
C                                                                       DHCG2930
C     TEST WHETHER H MUST BE HALVED OR DOUBLED                          DHCG2940
      DELT=0.D0                                                         DHCG2950
      DO 209 I=1,NDIM                                                   DHCG2960
  209 DELT=DELT+AUX(15,I)*DABS(AUX(16,I))                               DHCG2970
      IF(DELT-PRMT(4))210,222,222                                       DHCG2980
C                                                                       DHCG2990
C     H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.                   DHCG3000
  210 CALL FCT(X,Y,DERY)                                                DHCG3010
      CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                DHCG3020
      IF(PRMT(5))212,211,212                                            DHCG3030
  211 IF(IHLF-11)213,212,212                                            DHCG3040
  212 RETURN                                                            DHCG3050
  213 IF(H*(X-PRMT(2)))214,212,212                                      DHCG3060
  214 IF(DABS(X-PRMT(2))-.1D0*DABS(H))212,215,215                       DHCG3070
  215 IF(DELT-.02D0*PRMT(4))216,216,201                                 DHCG3080
C                                                                       DHCG3090
C                                                                       DHCG3100
C     H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE         DHCG3110
C     AVAILABLE                                                         DHCG3120
  216 IF(IHLF)201,201,217                                               DHCG3130
  217 IF(N-7)201,218,218                                                DHCG3140
  218 IF(ISTEP-4)201,219,219                                            DHCG3150
  219 IMOD=ISTEP/2                                                      DHCG3160
      IF(ISTEP-(MOD-(MOD))201,220,201                                   DHCG3170
  220 H=H+H                                                             DHCG3180
      IHLF=IHLF-1                                                       DHCG3190
      ISTEP=0                                                           DHCG3200
      DO 221 I=1,NDIM                                                   DHCG3210
      AUX(N-1,I)=AUX(N-2,I)                                             DHCG3220
      AUX(N-2,I)=AUX(N-4,I)                                             DHCG3230
      AUX(N-3,I)=AUX(N-6,I)                                             DHCG3240
      AUX(N+6,I)=AUX(N+5,I)                                             DHCG3250
      AUX(N+5,I)=AUX(N+3,I)                                             DHCG3260
      AUX(N+4,I)=AUX(N+1,I)                                             DHCG3270
      DELT=AUX(N+6,I)+AUX(N+5,I)                                        DHCG3280
      DELT=DELT+DELT+DELT                                               DHCG3290
 2210AUX(16,I)=8.9629629629629630D0*(Y(I)-AUX(N-3,I))                   DHCG3300
     1-3.3611111111111111D0*H*(DERY(I)+DELT+AUX(N+4,I))                 DHCG3310
      GOTO 201                                                          DHCG3320
C                                                                       DHCG3330
C                                                                       DHCG3340
C     H MUST BE HALVED                                                  DHCG3350
  222 IHLF=IHLF+1                                                       DHCG3360
      IF(IHLF-10)223,223,210                                            DHCG3370
  223 H=.5D0*H                                                          DHCG3380
      ISTEP=0                                                           DHCG3390
      DO 224 I=1,NDIM                                                   DHCG3400
     0Y(I)=.390625D-2*(8.D1*AUX(N-1,I)+135.D0*AUX(N-2,I)+4.D1*AUX(N-3,I)DHCG3410
     1+AUX(N-4,I))-.1171875D0*(AUX(N+6,I)-6.D0*AUX(N+5,I)-AUX(N+4,I))*H DHCG3420
     0AUX(N-4,I)=.390625D-2*(12.D0*AUX(N-1,I)+135.D0*AUX(N-2,I)+        DHCG3430
     1108.D0*AUX(N-3,I)+AUX(N-4,I))-.0234375D0*(AUX(N+6,I)+             DHCG3440
     218.D0*AUX(N+5,I)-9.D0*AUX(N+4,I))*H                               DHCG3450
      AUX(N-3,I)=AUX(N-2,I)                                             DHCG3460
  224 AUX(N+4,I)=AUX(N+5,I)                                             DHCG3470
      X=X-H                                                             DHCG3480
      DELT=X-(H+H)                                                      DHCG3490
      CALL FCT(DELT,Y,DERY)                                             DHCG3500
      DO 225 I=1,NDIM                                                   DHCG3510
      AUX(N-2,I)=Y(I)                                                   DHCG3520
      AUX(N+5,I)=DERY(I)                                                DHCG3530
  225 Y(I)=AUX(N-4,I)                                                   DHCG3540
      DELT=DELT-(H+H)                                                   DHCG3550
      CALL FCT(DELT,Y,DERY)                                             DHCG3560
      DO 226 I=1,NDIM                                                   DHCG3570
      DELT=AUX(N+5,I)+AUX(N+4,I)                                        DHCG3580
      DELT=DELT+DELT+DELT                                               DHCG3590
     0AUX(16,I)=8.9629629629629630D0*(AUX(N-1,I)-Y(I))                  DHCG3600
     1-3.3611111111111111D0*H*(AUX(N+6,I)+DELT+DERY(I))                 DHCG3610
  226 AUX(N+3,I)=DERY(I)                                                DHCG3620
      GOTO 206                                                          DHCG3630
      END                                                               DHCG3640
```

Subroutines HPCL and DHPCL

These subroutines use Hamming's modified predictor-corrector method for the solution of linear initial - value problems. The purpose of this version of Hamming's modified predictor-corrector method is to obtain an approximate solution of a linear system of first-order ordinary differential equations with given initial values.† It is a stable fourth-order integration procedure, requiring the evaluation of the right-hand side of the system only two times per step. This is a great advantage compared with other methods of the same order of accuracy, especially the Runge-Kutta method, which requires the evaluation of the right-hand side four times per step. Another advantage is that at each step the calculation procedure gives an estimate for the local truncation error; thus the procedure is able -- without a significant amount of calculation time -- to choose and change the step size h. On the other hand, Hamming's predictor-corrector method is not self-starting; that is, the functional values at a single previous point are not enough to obtain the functional values ahead. Therefore, to get the starting values, a special Runge-Kutta procedure followed by one iteration step is added to the predictor-corrector method.

Given the linear system of first-order ordinary differential equations:

$$y'_1 = \frac{dy_1}{dx} = a_{11}(x)\, y_1 + a_{12}(x)\, y_2 + \ldots + a_{1n}(x)\, y_n + f_1(x)$$

$$y'_2 = \frac{dy_2}{dx} = a_{21}(x)\, y_1 + a_{22}(x)\, y_2 + \ldots + a_{2n}(x)\, y_n + f_2(x)$$

$$\cdots\cdots\cdots\cdots\cdots\cdots$$

$$y'_n = \frac{dy_n}{dx} = a_{n1}(x)\, y_1 + a_{n2}(x)\, y_2 + \ldots + a_{nn}(x)\, y_n + f_n(x)$$

† Solution can be done with subroutines HPCG and DHPCG too. The main reason for this special version, suitable to systems of linear differential equations, is that external subroutines furnished by the user for the solution of linear boundary-value problems can be used without changes by subroutine HPCL or DHPCL for the solution of corresponding linear initial-value problems.

and the initial values:

$$y_1(x_0) = y_{1,0},\ y_2(x_0) = y_{2,0},\ \ldots,\ y_n(x_0) = y_{n,0}$$

and using the following vector and matrix notations:

$$Y(x) = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_n(x) \end{pmatrix}, \qquad f(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{pmatrix},$$

$$A(x) = \begin{pmatrix} a_{11}(x) & a_{12}(x) & \ldots & a_{1n}(x) \\ a_{21}(x) & a_{22}(x) & \ldots & a_{2n}(x) \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1}(x) & a_{n2}(x) & \ldots & a_{nn}(x) \end{pmatrix}$$

$$\text{and } Y_0 = \begin{pmatrix} y_{1,0} \\ y_{2,0} \\ \vdots \\ y_{n,0} \end{pmatrix}$$

where Y, F and Y_0 are column vectors and A is an n by n square matrix, the given problem appears as follows:

$$Y' = \frac{dY}{dx} = A(x) * Y + F(x) \text{ with } Y(x_0) = Y_0$$

For stability purposes, the modification by Hamming of Milne's classical modified predictor-corrector method is preferred. Thus, knowing the results at the equidistant points x_{j-3}, x_{j-2}, x_{j-1} and x_j, the results at point $x_{j+1} = x_j + h$ are computed by the formulas given below.

$$\text{Predictor: } P_{j+1} = Y_{j-3} + \frac{4h}{3}(2Y'_j - Y'_{j-1} + 2Y'_{j-2}) \tag{1}$$

$$\text{Modifier: } M_{j+1} = P_{j+1} - \frac{112}{121}(P_j - C_j) \tag{2}$$

$$M'_{j+1} = A(x_{j+1}) * M_{j+1} + F(x_{j+1}) \tag{3}$$

$$\text{Corrector: } C_{j+1} = \frac{1}{8}\left[9Y_j - Y_{j-2} + 3h(M'_{j+1} + 2Y'_j - Y'_{j-1})\right] \tag{4}$$

$$\text{Final Value: } Y_{j+1} = C_{j+1} + \frac{9}{121}(P_{j+1} - C_{j+1}) \tag{5}$$

where Y, Y', P, M, M' and C are all column vectors with n components, and A is an n by n square matrix. Formulas (1) and (4) have local truncation errors:

$$T_1 = \frac{14}{45}h^5 Y^{(5)}(\xi_1) \text{ with } \xi_1 \in (x_{j-3}, x_{j+1})$$

and

$$T_2 = -\frac{1}{40}h^5 Y^{(5)}(\xi_2) \text{ with } \xi_2 \in (x_{j-2}, x_{j+1})$$

respectively such that

$$C_{j+1} - P_{j+1} = \frac{121}{360}h^5 Y^{(5)}(\xi) \text{ with } \xi \in (x_{j-3}, x_{j+1})$$

Assuming that $Y^{(5)}(x)$ does not vary to any great extent in the interval (x_{j-3}, x_{j+1}), it follows that:

$$T_2 \approx \frac{9}{121}(P_{j+1} - C_{j+1})$$

This formula shows that the components of column vector $P_{j+1} - C_{j+1}$ are measures for the local truncation errors in the components of column vector Y_{j+1}. Therefore, control of accuracy and adjustment of step size h can be done by generating the following test value:

$$\delta = \sum_{i=1}^{n} a_i \cdot \left| P_{j+1,i} - c_{j+1,i} \right| \tag{6}$$

where the coefficients a_i $(i = 1, 2, \ldots, n)$ are error-weights specified in the input of the procedure.

If δ is greater than a given tolerance ε†, increment h is halved and the procedure computes $Y_{j-\frac{1}{2}}$ -- that is, $Y(x_j + \frac{h}{2})$ -- after having interpolated $Y_{j-\frac{1}{2}} = Y(x_j - \frac{h}{2})$ and $Y_{j-\frac{3}{2}} = Y(x_j - \frac{3h}{2})$, with previous increment h, using the following sixth-order interpolation formulas:

$$Y_{j-\frac{1}{2}} = \frac{1}{256}(80Y_j + 135Y_{j-1} + 40Y_{j-2} + Y_{j-3}) + \frac{h}{2} \cdot \frac{15}{128}(-Y'_j + 6Y'_{j-1} + Y'_{j-2}) \tag{7}$$

† Numerical experience seems to show that the procedure does not exceed a global relative error approximately equal to ε.

$$Y_{j-\frac{3}{2}} = \frac{1}{256}(12\,Y_j + 135\,Y_{j-1} + 108\,Y_{j-2} + Y_{j-3})$$

$$+ \frac{h}{2} \cdot \frac{3}{128}(-Y'_j - 18\,Y'_{j-1} + 9\,Y'_{j-2}) \qquad (8)$$

If δ is less than ε, the result Y_{j+1} is assumed to be correct and is handed, together with x_{j+1} and the vector of derivatives Y'_{j+1}, to a user-supplied output subroutine.

If δ is less than $\varepsilon/50$, the next step is carried out with the doubled increment. However, care is taken in the procedure, so that the increment never becomes greater than the increment h specified as an input parameter, and further that among the output are all points $x_0 + j \cdot h$ (where $j = 0, 1, 2, \ldots$, and h is the input step size) which are situated between the lower and the upper bound of the integration interval.

Changing the step size by halving or doubling requires changing of $P_j - C_j$ or $P_{j+1} - C_{j+1}$ respectively. Using the following interpolation formula:

$$Y_j = Y_{j-3} + \frac{3}{8}\,h\,(Y'_{j-3} + 3\,Y'_{j-2} + 3\,Y'_{j-1} + Y_j)$$

$$- \frac{3}{80}\,h^5\,Y^{(5)}(\xi_3)$$

with

$$\xi_3\ \varepsilon(x_{j-3},\ x_j)$$

and assuming that $Y^{(5)}(x)$ does not vary to any great extent in this interval, $P_j - C_j$ can be written in the following form:

$$P_j - C_j \approx \frac{242}{27}(Y_j - Y_{j-3}) - \frac{121}{36}h\,(Y'_j + 3\,Y'_{j-1}$$

$$+ 3\,Y'_{j-2} + Y'_{j-3})$$

When halving increment h, this formula becomes:

$$P_j - C_j \approx \frac{242}{27}(Y_j - Y_{j-\frac{3}{2}}) - \frac{121}{36} \cdot \frac{h}{2}(Y'_j + 3\,Y'_{j-\frac{3}{2}})$$

$$+ 3\,Y'_{j-1} + Y'_{j-\frac{3}{2}}) \qquad (9)$$

and when doubling:

$$P_{j+1} - C_{j+1} \approx \frac{242}{27}(Y_{j+1} - Y_{j-5}) - \frac{121}{36} \cdot 2\,h$$

$$(Y'_{j+1} + 3\,Y'_{j-1} + 3\,Y'_{j-3} + Y'_{j-5}) \qquad (10)$$

Starting Hamming's modified predictor-corrector method requires the functional and derivative values at four preceding equidistant points: x_0, x_1, x_2 and x_3. The values Y_0 and $Y'_0 = A(x_0) * Y_0 + F(x_0)$ are specified by input. For computation of Y_1, Y'_1, Y_2, Y'_2, Y_3 and Y'_3 and for adjustment of step size h to accuracy requirements, a special Runge-Kutta procedure suggested by Ralston is used. Starting at x_j, result values at point $x_{j+1} = x_j + h$ are computed using the following formulas:

$$K_1 = h \cdot Y'_j \qquad (11)$$

$$K_2 = h \cdot \Big[A\,(x_j + 0.4\,h) * (Y_j + 0.4\,K_1)$$

$$+ F(x_j + 0.4\,h)\Big] \qquad (12)$$

$$K_3 = h \cdot \Big[A\,(x_j + 0.45573725421878943\,h) *$$

$$(Y_j + 0.29697760924775360\,K_1$$

$$+ 0.15875964497103583\,K_2)$$

$$+ F(x_j + 0.45573725421878943\,h)\Big] \qquad (13)$$

$$K_4 = h \cdot \Big[A\,(x_j + h) * (Y_j + 0.21810038822592047\,K_1$$

$$- 3.0509651486929308\,K_2$$

$$+ 3.8328647604670103\,K_3) + F\,(x_j + h)\Big] \qquad (14)$$

$$Y_{j+1} = Y_j + 0.17476028226269037\,K_1$$

$$- 0.55148066287873294\,K_2$$

$$+ 1.2055355993965235\,K_3$$

$$+ 0.17118478121951903\,K_4 \qquad (15)$$

where Y_j, Y_{j+1}, Y'_j, K_1, K_2, K_3 and K_4 are all column vectors with n components, and A is an n by n square matrix. These formulas are not very stable, but this does not matter because they are used only in three successive steps ($j = 0, 1, 2$). On the other hand, they have the smallest bound of truncation errors of all fourth-order Runge-Kutta procedures. Therefore they are best suited to start other integration methods which are not self-starting.

For initial control of accuracy and adjustment of step size h, in starting the procedure an approximation for $Y_2 = Y\,(x_0 + h)$ called $Y_2^{(1)}$ is computed using the step size h, and then an approximation called $Y_2^{(2)}$, using the step size h/2 twice. From these two approximations, a test value δ for accuracy is generated in the following way:

$$\delta = \frac{1}{15} \sum_{i=1}^{n} a_i \cdot \left| y_{2,i}^{(1)} - y_{2,i}^{(2)} \right| \qquad (16)$$

If δ is greater than ε, increment h is halved and the procedure starts anew at point x_0.

If δ is less than ε, the results $Y_1^{(2)} = Y(x_0 + \frac{h}{2})$ and $Y_2^{(2)} = Y(x_0 + h)$ are assumed to be correct and a third step follows which computes the results at point $x_0 + \frac{3h}{2}$ -- that is, $Y_3 = Y(x_0 + \frac{3}{2}h)$.

The step size h/2 of these three steps is used as initial step size h in the predictor-corrector method.

It is very important that the starting values be as accurate as possible, because errors in these starting values may increase during the following predictor-corrector procedure. Therefore the starting values computed by the Runge-Kutta method are refined by one iteration step using the following fourth-order interpolation formulas:

$$Y_1 = Y_0 + \frac{h}{24}(9\,Y'_0 + 19\,Y'_1 - 5\,Y'_2 + Y'_3) \qquad (17)$$

$$Y_2 = Y_0 + \frac{h}{3}(Y'_0 + 4\,Y'_1 + Y'_2) \qquad (18)$$

$$Y_3 = Y_0 + \frac{3h}{8}(Y'_0 + 3\,Y'_1 + 3\,Y'_2 + Y'_3) \qquad (19)$$

which must be considered as an iteration procedure. That is, first the result values of the Runge-Kutta method are handed to the right-hand side of formula (17) to compute a refined Y_1. After having computed the refined Y'_1, the refined vector Y_2 is generated using formula (18). Finally, refined Y'_2 is used, together with the other values, in the right-hand side of formula (19) to compute the refined vector Y_3. During this iteration, the refined data sets x_j, Y_j, Y'_j are handed, together with the number of bisections of the initial step size (specified by input), to the output subroutine.

It can be shown that, using this iterative procedure, the initial column vector $P_3 - C_3$ used in formula (2) for j = 3 is equal to zero.

The entire input of the procedure is:

1. Lower and upper bound of the integration interval, initial step size h of the independent variable, and upper bound ε of the local truncation error

2. Initial values Y_0 of the dependent variables and weights a_i (i = 1, 2, . . . , n) for the local truncation errors in each component of the dependent variables

3. The number n of differential equations in the system

4. As external subroutine subprograms, the computation of the right-hand side matrix A of the system of linear differential equations and of the inhomogeneous vector F; for flexibility in output, an output subroutine

5. An auxiliary storage array with 16 rows and n columns

6. An n by n matrix as auxiliary storage array

Output is done in the following way. If a set of approximations to the dependent variables Y (x) is found to be of sufficient accuracy, it is handed -- together with x, the derivative Y' (x), the number of bisections of the initial increment, the number of differential equations, the lower and upper bound of the interval, the initial step size, error bound ε, and a parameter for terminating subroutine HPCL or DHPCL -- to the output subroutine. By means of this output subroutine, the user has the opportunity to choose his own output format, to handle the output values as he wants, to change the upper error bound and to terminate subroutine HPCL or DHPCL at any output point. In particular, the user is able to drop the output of some intermediate points, printing only the result values at the special points $x_0 + j \cdot h$, where j = 0, 1, 2, ... , and he is the initial step size specified by input. The user may also perform intermediate computation using the integration results before continuing the process. For better understanding of the flowchart and of the FORTRAN program, see the allocation of special intermediate result vectors within the storage array AUX, shown previously in Figure 12.

For reference see:

(1) A. Ralston/ H.S. Wilf, Mathematical Methods for Digital Computers, Wiley, New York/London, 1960, pp. 95 - 109.

(2) A. Ralston, "Runge-Kutta Methods with Minimum Error Bounds", MTAC, vol. 16, no. 80 (1962), pp. 431 - 437.

```
C
C     ..................................................................
C
C        SUBROUTINE HPCL
C
C        PURPOSE
C           TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY LINEAR
C           DIFFERENTIAL EQUATIONS WITH GIVEN INITIAL VALUES.
C
C        USAGE
C           CALL HPCL (PRMT,Y,DERY,NDIM,IHLF,AFCT,FCT,OUTP,AUX,A)
C           PARAMETERS AFCT,FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.
C
C        DESCRIPTION OF PARAMETERS
C           PRMT    - AN INPUT AND OUTPUT VECTOR WITH DIMENSION GREATER
C                     OR EQUAL TO 5, WHICH SPECIFIES THE PARAMETERS OF
C                     THE INTERVAL AND OF ACCURACY AND WHICH SERVES FOR
C                     COMMUNICATION BETWEEN OUTPUT SUBROUTINE (FURNISHED
C                     BY THE USER) AND SUBROUTINE HPCL. EXCEPT PRMT(5)
C                     THE COMPONENTS ARE NOT DESTROYED BY SUBROUTINE
C                     HPCL AND THEY ARE
C           PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),
C           PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE
C                     (INPUT),
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.
C                     IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS
C                     OUTPUT SUBROUTINE.
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE HPCL INITIALIZES
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE
C                     SUBROUTINE HPCL AT ANY OUTPUT POINT, HE HAS TO
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER
C                     THAN 5. HOWEVER SUBROUTINE HPCL DOES NOT REQUIRE
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM
C                     (CALLING HPCL) WHICH ARE OBTAINED BY SPECIAL
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.
C           Y       - INPUT VECTOR OF INITIAL VALUES.  (DESTROYED)
C                     LATERON Y IS THE RESULTING VECTOR OF DEPENDENT
C                     VARIABLES COMPUTED AT INTERMEDIATE POINTS X.
C           DERY    - INPUT VECTOR OF ERROR WEIGHTS.  (DESTROYED)
C                     THE SUM OF ITS COMPONENTS MUST BE EQUAL TO 1.
C                     LATERON DERY IS THE VECTOR OF DERIVATIVES, WHICH
C                     BELONG TO FUNCTION VALUES Y AT A POINT X.
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF
C                     EQUATIONS IN THE SYSTEM.
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS
C                     GREATER THAN 10, SUBROUTINE HPCL RETURNS WITH
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.
C                     ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-
C                     PRMT(1)) RESPECTIVELY.
C           AFCT    - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT
C                     COMPUTES MATRIX A (FACTOR OF VECTOR Y ON THE
C                     RIGHT HAND SIDE OF THE SYSTEM) FOR A GIVEN X-VALUE.
C                     ITS PARAMETER LIST MUST BE X,A. THE SUBROUTINE
C                     SHOULD NOT DESTROY X.
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT
C                     COMPUTES VECTOR F (INHOMOGENEOUS PART OF THE
C                     RIGHT HAND SIDE OF THE SYSTEM) FOR A GIVEN X-VALUE.
C                     ITS PARAMETER LIST MUST BE X,F. THE SUBROUTINE
C                     SHOULD NOT DESTROY X.
C           OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.
C                     ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.
C                     NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,
C                     PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY
C                     SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,
C                     SUBROUTINE HPCL IS TERMINATED.
C           AUX     - AN AUXILIARY STORAGE ARRAY WITH 16 ROWS AND NDIM
C                     COLUMNS.
C           A       - AN NDIM BY NDIM MATRIX, WHICH IS USED AS AUXILIARY
C                     STORAGE ARRAY.
C
C        REMARKS
C           THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF
C           (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE
C               NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE
C               IHLF=11),
C           (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN
C               (ERROR MESSAGES IHLF=12 OR IHLF=13),
C           (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,
C           (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL SUBROUTINES AFCT(X,A), FCT(X,F) AND
C           OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF HAMMINGS MODIFIED PREDICTOR-
C           CORRECTOR METHOD. IT IS A FOURTH ORDER METHOD, USING 4
C           PRECEEDING POINTS FOR COMPUTATION OF A NEW VECTOR Y OF THE
C           DEPENDENT VARIABLES.
C           FOURTH ORDER RUNGE-KUTTA METHOD SUGGESTED BY RALSTON IS
C           USED FOR ADJUSTMENT OF THE INITIAL INCREMENT AND FOR
C           COMPUTATION OF STARTING VALUES.
C           SUBROUTINE HPCL AUTOMATICALLY ADJUSTS THE INCREMENT DURING
C           THE WHOLE COMPUTATION BY HALVING OR DOUBLING.
C           TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE
C           MUST BE CODED BY THE USER.
C           FOR REFERENCE, SEE
C           (1) RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL
C               COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.
C           (2) RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS,
C               MTAC, VOL.16, ISS.80 (1962), PP.431-437.
C
C     ..................................................................
C
      SUBROUTINE HPCL(PRMT,Y,DERY,NDIM,IHLF,AFCT,FCT,OUTP,AUX,A)
C
C
C     THE FOLLOWING FIRST PART OF SUBROUTINE HPCL (UNTIL FIRST BREAK-
C     POINT FOR LINKAGE) HAS TO STAY IN CORE DURING THE WHOLE
C     COMPUTATION
C
      DIMENSION PRMT(1),Y(1),DERY(1),AUX(16,1),A(1)
      GOTO 100
C
C     THIS PART OF SUBROUTINE HPCL COMPUTES THE RIGHT HAND SIDE DERY OF
C     THE GIVEN SYSTEM OF LINEAR DIFFERENTIAL EQUATIONS.
    1 CALL AFCT(X,A)
      CALL FCT(X,DERY)
      DO 3 M=1,NDIM
      LL=M-NDIM
      HS=0.
      DO 2 L=1,NDIM
      LL=LL+NDIM
    2 HS=HS+A(LL)*Y(L)
    3 DERY(M)=HS+DERY(M)
      GOTO(105,202,204,206,115,122,125,308,312,327,329,128),ISW2
C
C     POSSIBLE BREAK-POINT FOR LINKAGE
C
  100 N=1
      IHLF=0
      X=PRMT(1)
      H=PRMT(3)
      PRMT(5)=0.
      DO 101 I=1,NDIM
      AUX(16,I)=0.
      AUX(15,I)=DERY(I)
  101 AUX(1,I)=Y(I)
      IF(H*(PRMT(2)-X))103,102,104
C
C     ERROR RETURNS
  102 IHLF=12
      GOTO 104
  103 IHLF=13
C
C     COMPUTATION OF DERY FOR STARTING VALUES
  104 ISW2=1
      GOTO 1
C
C     RECORDING OF STARTING VALUES
  105 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))107,106,107
  106 IF(IHLF)108,108,107
  107 RETURN
  108 DO 109 I=1,NDIM
  109 AUX(8,I)=DERY(I)
C
C     COMPUTATION OF AUX(2,I)
      ISW1=1
      GOTO 200
C
  110 X=X+H
      DO 111 I=1,NDIM
  111 AUX(2,I)=Y(I)
C
C     INCREMENT H IS TESTED BY MEANS OF BISECTION
  112 IHLF=IHLF+1
      X=X-H
      DO 113 I=1,NDIM
  113 AUX(4,I)=AUX(2,I)
      H=.5*H
      N=1
      ISW1=2
      GOTO 200
C
  114 X=X+H
      ISW2=5
      GOTO 1
  115 N=2
      DO 116 I=1,NDIM
      AUX(2,I)=Y(I)
  116 AUX(9,I)=DERY(I)
      ISW1=3
      GOTO 200
C
C     COMPUTATION OF TEST VALUE DELT
  117 DELT=0.
      DO 118 I=1,NDIM
  118 DELT=DELT+AUX(15,I)*ABS(Y(I)-AUX(4,I))
      DELT=.06666667*DELT
      IF(DELT-PRMT(4))121,121,119
  119 IF(IHLF-10)112,120,120
C
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.
  120 IHLF=11
      X=X+H
      GOTO 104
C
C     SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS
  121 X=X+H
      ISW2=6
      GOTO 1
  122 DO 123 I=1,NDIM
      AUX(3,I)=Y(I)
  123 AUX(10,I)=DERY(I)
      N=3
      ISW1=4
      GOTO 200
C
  124 N=1
      X=X+H
      ISW2=7
      GOTO 1
  125 X=PRMT(1)
      DO 126 I=1,NDIM
      AUX(11,I)=DERY(I)
 1260Y(I,=AUX(1,I)+H*(.375*AUX(8,I)+.7916667*AUX(9,I)
     1-.2083333*AUX(10,I)+.04166667*DERY(I))
  127 X=X+H
      N=N+1
      ISW2=12
      GOTO 1
  128 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))107,129,107
  129 IF(N-4)130,300,300
  130 DO 131 I=1,NDIM
      AUX(N,I)=Y(I)
  131 AUX(N+7,I)=DERY(I)
      IF(N-3)132,134,300
C
  132 DO 133 I=1,NDIM
      DELT=AUX(9,I)+AUX(9,I)
      DELT=DELT+DELT
  133 Y(I)=AUX(1,I)+.3333333*H*(AUX(8,I)+DELT+AUX(10,I))
      GOTO 127
C
  134 DO 135 I=1,NDIM
      DELT=AUX(9,I)+AUX(10,I)
      DELT=DELT+DELT+DELT
  135 Y(I)=AUX(1,I)+.375*H*(AUX(8,I)+DELT+AUX(11,I))
      GOTO 127
C
C     THE FOLLOWING PART OF SUBROUTINE HPCL COMPUTES BY MEANS OF
C     RUNGE-KUTTA METHOD STARTING VALUES FOR THE NOT SELF-STARTING
C     PREDICTOR-CORRECTOR METHOD.
  200 Z=X
      DO 201 I=1,NDIM
      X=H*AUX(N+7,I)
      AUX(5,I)=X
```

```
  201 Y(I)=AUX(N,I)+.4*X
C     X IS AN AUXILIARY STORAGE LOCATION
C
      X=Z+.4*H
      ISW2=2
      GOTO 1
  202 DO 203 I=1,NDIM
      X=H*DERY(I)
      AUX(6,I)=X
  203 Y(I)=AUX(N,I)+.2969776*AUX(5,I)+.1587596*X
C
      X=Z+.4557372*H
      ISW2=3
      GOTO 1
  204 DO 205 I=1,NDIM
      X=H*DERY(I)
      AUX(7,I)=X
  205 Y(I)=AUX(N,I)+.2181004*AUX(5,I)-3.050965*AUX(6,I)+3.832865*X
C
      X=Z*H
      ISW2=4
      GOTO 1
  206 DO 207 I=1,NDIM
  2070Y(I)=AUX(N,I)+.1747603*AUX(5,I)-.5514807*AUX(6,I)
     1+1.205536*AUX(7,I)+.1711848*H*DERY(I)
      X=Z
      GOTO(110,114,117,124),ISW1
C
C     POSSIBLE BREAK-POINT FOR LINKAGE
C
C     STARTING VALUES ARE COMPUTED.
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.
  309 ISTEP=1
  301 IF(N-8)304,302,304
C
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS
  302 DO 303 N=2,7
      DO 303 I=1,NDIM
      AUX(N-1,I)=AUX(N,I)
  303 AUX(N+6,I)=AUX(N+7,I)
      N=7
C
C     N LESS THAN 8 CAUSES N+1 TO GET N
  304 N=N+1
C
C     COMPUTATION OF NEXT VECTOR Y
      DO 305 I=1,NDIM
      AUX(N-1,I)=Y(I)
  305 AUX(N+6,I)=DERY(I)
      X=X+H
  306 ISTEP=ISTEP+1
      DO 307 I=1,NDIM
     ODELT=AUX(N-4,I)+1.333333*H*(AUX(N+6,I)+AUX(N+6,I)-AUX(N+5,I)+
     1AUX(N+4,I)+AUX(N+4,I))
      Y(I)=DELT-.9256198*AUX(16,I)
  307 AUX(16,I)=DELT
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX, MODIFIED PREDICTOR
C     IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.
      ISW2=8
      GOTO 1
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY
C
  308 DO 309 I=1,NDIM
     ODELT=.125*(9.*AUX(N-1,I)-AUX(N-3,I)+3.*H*(DERY(I)+AUX(N+6,I)+
     1AUX(N+6,I)-AUX(N+5,I)))
      AUX(16,I)=AUX(16,I)-DELT
  309 Y(I)=DELT+.07438017*AUX(16,I)
C
C     TEST WHETHER H MUST BE HALVED OR DOUBLED
      DELT=0.
      DO 310 I=1,NDIM
  310 DELT=DELT+AUX(15,I)*ABS(AUX(16,I))
      IF(DELT-PRMT(4))311,324,324
C
C     H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.
  311 ISW2=9
      GOTO 1
  312 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))314,313,314
  313 IF(IHLF-11)315,314,314
  314 RETURN
  315 IF(H*(X-PRMT(2)))316,314,314
  316 IF(ABS(X-PRMT(2))-.1*ABS(H))314,317,317
  317 IF(DELT-.02*PRMT(4))318,318,301
C
C
C     H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE
C     AVAILABLE
  318 IF(IHLF)301,301,319
  319 IF(N-7)301,320,320
  320 IF(ISTEP-4)301,321,321
  321 IMOD=ISTEP/2
      IF(ISTEP-IMOD-IMOD)301,322,301
  322 H=H+H
      IHLF=IHLF-1
      ISTEP=0
      DO 323 I=1,NDIM
      AUX(N-1,I)=AUX(N-2,I)
      AUX(N-2,I)=AUX(N-4,I)
      AUX(N-3,I)=AUX(N-6,I)
      AUX(N+6,I)=AUX(N+5,I)
      AUX(N+5,I)=AUX(N+3,I)
      AUX(N+4,I)=AUX(N+1,I)
      DELT=AUX(N+6,I)+AUX(N+5,I)
      DELT=DELT+DELT+DELT
 3230AUX(16,I)=8.962963*(Y(I)-AUX(N-3,I))-3.361111*H*(DERY(I)+DELT
     1+AUX(N+4,I))
      GOTO 301
C
C
C     H MUST BE HALVED
  324 IHLF=IHLF+1
      IF(IHLF-10)325,325,311
  325 H=.5*H
      ISTEP=0
      DO 326 I=1,NDIM
     0Y(I)=.00390625*(80.*AUX(N-1,I)+135.*AUX(N-2,I)+40.*AUX(N-3,I)+
     1AUX(N-4,I))-.1171875*(AUX(N+6,I)-6.*AUX(N+5,I)-AUX(N+4,I))*H
     0AUX(N-4,I)=.00390625*(12.*AUX(N-1,I)+135.*AUX(N-2,I)+
     1108.*AUX(N-3,I)+AUX(N-4,I))-.0234375*(AUX(N+6,I)+18.*AUX(N+5,I)-
     29.*AUX(N+4,I))*H
      AUX(N-3,I)=AUX(N-2,I)
  326 AUX(N+4,I)=AUX(N+5,I)
      DELT=X-H
      X=DELT-(H+H)
      ISW2=10
      GOTO 1
  327 DO 328 I=1,NDIM
      AUX(N-2,I)=Y(I)
      AUX(N+5,I)=DERY(I)
  328 Y(I)=AUX(N-4,I)
      X=X-(H+H)
      ISW2=11
      GOTO 1
  329 X=DELT
      DO 330 I=1,NDIM
      DELT=AUX(N+5,I)+AUX(N+4,I)
      DELT=DELT+DELT+DELT
     0AUX(16,I)=8.962963*(AUX(N-1,I)-Y(I))-3.361111*H*(AUX(N+6,I)+DELT
     1+DERY(I))
  330 AUX(N+3,I)=DERY(I)
      GOTO 306
      END
```

```
C
C     ..................................................................
C
C        SUBROUTINE DHPCL
C
C        PURPOSE
C           TO SOLVE A SYSTEM OF FIRST ORDER ORDINARY LINEAR
C           DIFFERENTIAL EQUATIONS WITH GIVEN INITIAL VALUES.
C
C        USAGE
C           CALL DHPCL (PRMT,Y,DERY,NDIM,IHLF,AFCT,FCT,OUTP,AJX,A)
C           PARAMETERS AFCT,FCT AND OUTP REQUIRE AN EXTERNAL STATEMENT.
C
C        DESCRIPTION OF PARAMETERS
C           PRMT    - DOUBLE PRECISION INPUT AND OUTPUT VECTOR WITH
C                     DIMENSION GREATER THAN OR EQUAL TO 5, WHICH
C                     SPECIFIES THE PARAMETERS OF THE INTERVAL AND OF
C                     ACCURACY AND WHICH SERVES FOR COMMUNICATION BETWEEN
C                     OUTPUT SUBROUTINE (FURNISHED BY THE USER) AND
C                     SUBROUTINE DHPCL. EXCEPT PRMT(5) THE COMPONENTS
C                     ARE NOT DESTROYED BY SUBROUTINE DHPCL AND THEY ARE
C           PRMT(1)- LOWER BOUND OF THE INTERVAL (INPUT),
C           PRMT(2)- UPPER BOUND OF THE INTERVAL (INPUT),
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE
C                     (INPUT),
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF ABSOLUTE ERROR IS
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.
C                     IF INCREMENT IS LESS THAN PRMT(3) AND ABSOLUTE
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS
C                     OUTPUT SUBROUTINE.
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE DHPCL INITIALIZES
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE
C                     SUBROUTINE DHPCL AT ANY OUTPUT POINT, HE HAS TO
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER
C                     THAN 5. HOWEVER SUBROUTINE DHPCL DOES NOT REQUIRE
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM
C                     (CALLING DHPCL) WHICH ARE OBTAINED BY SPECIAL
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.
C           Y       - DOUBLE PRECISION INPUT VECTOR OF INITIAL VALUES
C                     (DESTROYED). LATERON Y IS THE RESULTING VECTOR OF
C                     DEPENDENT VARIABLES COMPUTED AT INTERMEDIATE
C                     POINTS X.
C           DERY    - DOUBLE PRECISION INPUT VECTOR OF ERROR WEIGHTS
C                     (DESTROYED). THE SUM OF ITS COMPONENTS MUST BE
C                     EQUAL TO 1. LATERON DERY IS THE VECTOR OF
C                     DERIVATIVES, WHICH BELONG TO FUNCTION VALUES Y AT
C                     INTERMEDIATE POINTS X.
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF
C                     EQUATIONS IN THE SYSTEM.
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS
C                     GREATER THAN 10, SUBROUTINE DHPCL RETURNS WITH
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.
C                     ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-
C                     PRMT(1)) RESPECTIVELY.
C           AFCT    - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT
C                     COMPUTES MATRIX A (FACTOR OF VECTOR Y ON THE
C                     RIGHT HAND SIDE OF THE SYSTEM) FOR A GIVEN X-VALUE.
C                     ITS PARAMETER LIST MUST BE X,A. THE SUBROUTINE
C                     SHOULD NOT DESTROY X.
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT
C                     COMPUTES VECTOR F (INHOMOGENEOUS PART OF THE
C                     RIGHT HAND SIDE OF THE SYSTEM) FOR A GIVEN X-VALUE.
C                     ITS PARAMETER LIST MUST BE X,F. THE SUBROUTINE
C                     SHOULD NOT DESTROY X.
C           OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.
C                     ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.
C                     NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,
C                     PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY
C                     SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,
C                     SUBROUTINE DHPCL IS TERMINATED.
C           AUX     - DOUBLE PRECISION AUXILIARY STORAGE ARRAY WITH 16
C                     ROWS AND NDIM COLUMNS.
C           A       - DOUBLE PRECISION NDIM BY NDIM MATRIX, WHICH IS USED
C                     AS AUXILIARY STORAGE ARRAY.
C
C        REMARKS
C           THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF
C           (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE
C               NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE
C               IHLF=11),
C           (2) INITIAL INCREMENT IS EQUAL TO 0 OR HAS WRONG SIGN
C               (ERROR MESSAGES IHLF=12 OR IHLF=13),
C           (3) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,
C           (4) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           THE EXTERNAL SUBROUTINES AFCT(X,A), FCT(X,F) AND
C           OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED BY THE USER.
C
C        METHOD
C           EVALUATION IS DONE BY MEANS OF HAMMINGS MODIFIED PREDICTOR-
C           CORRECTOR METHOD. IT IS A FOURTH ORDER METHOD, USING 4
C           PRECEEDING POINTS FOR COMPUTATION OF A NEW VECTOR Y OF THE
C           DEPENDENT VARIABLES.
C           FOURTH ORDER RUNGE-KUTTA METHOD SUGGESTED BY RALSTON IS
C           USED FOR ADJUSTMENT OF THE INITIAL INCREMENT AND FOR
C           COMPUTATION OF STARTING VALUES.
C           SUBROUTINE DHPCL AUTOMATICALLY ADJUSTS THE INCREMENT DURING
C           THE WHOLE COMPUTATION BY HALVING OR DOUBLING.
```

```
C           TO GET FULL FLEXIBILITY IN OUTPUT, AN OUTPUT SUBROUTINE
C           MUST BE CODED BY THE USER.
C           FOR REFERENCE, SEE
C           (1)  RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL
C                COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.
C           (2)  RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS,
C                MTAC, VOL.16, ISS.80 (1962), PP.431-437.
C
C     ..................................................................
C
      SUBROUTINE DHPCL(PRMT,Y,DERY,NDIM,IHLF,AFCT,FCT,OUTP,AUX,A)
C
C
C     THE FOLLOWING FIRST PART OF SUBROUTINE DHPCL (UNTIL FIRST BREAK-
C     POINT FOR LINKAGE) HAS TO STAY IN CORE DURING THE WHOLE
C     COMPUTATION
C
      DIMENSION PRMT(1),Y(1),DERY(1),AUX(16,1),A(1)
      DOUBLE PRECISION PRMT,Y,DERY,AUX,X,H,Z,DELT,A,HS
      GOTO 100
C
C     THIS PART OF SUBROUTINE DHPCL COMPUTES THE RIGHT HAND SIDE DERY OF
C     THE GIVEN SYSTEM OF LINEAR DIFFERENTIAL EQUATIONS.
    1 CALL AFCT(X,A)
      CALL FCT(X,DERY)
      DO 3 M=1,NDIM
      LL=M-NDIM
      HS=0.D0
      DO 2 L=1,NDIM
      LL=LL+NDIM
    2 HS=HS+A(LL)*Y(L)
    3 DERY(M)=HS+DERY(M)
      GOTO(105,202,204,206,115,122,125,308,312,327,329,128),ISW2
C
C     POSSIBLE BREAK-POINT FOR LINKAGE
C
  100 N=1
      IHLF=0
      X=PRMT(1)
      H=PRMT(3)
      PRMT(5)=0.D0
      DO 101 I=1,NDIM
      AUX(16,I)=0.D0
      AUX(15,I)=DERY(I)
  101 AUX(1,I)=Y(I)
      IF(H*(PRMT(2)-X))103,102,104
C
C     ERROR RETURNS
  102 IHLF=12
      GOTO 104
  103 IHLF=13
C
C     COMPUTATION OF DERY FOR STARTING VALUES
  104 ISW2=1
      GOTO 1
C
C     RECORDING OF STARTING VALUES
  105 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))107,106,107
  106 IF(IHLF)108,108,107
  107 RETURN
  108 DO 109 I=1,NDIM
C
  109 AUX(8,I)=DERY(I)
C     COMPUTATION OF AUX(2,I)
      ISW1=1
      GOTO 200
  110 X=X+H
      DO 111 I=1,NDIM
  111 AUX(2,I)=Y(I)
C
C     INCREMENT H IS TESTED BY MEANS OF BISECTION
  112 IHLF=IHLF+1
      X=X-H
      DO 113 I=1,NDIM
  113 AUX(4,I)=AUX(2,I)
      H=.5D0*H
      N=1
      ISW1=2
      GOTO 200
C
  114 X=X+H
      ISW2=5
      GOTO 1
  115 N=2
      DO 116 I=1,NDIM
      AUX(2,I)=Y(I)
  116 AUX(9,I)=DERY(I)
      ISW1=3
      GOTO 200
C
C     COMPUTATION OF TEST VALUE DELT
  117 DELT=0.D0
      DO 118 I=1,NDIM
  118 DELT=DELT+AUX(15,I)*DABS(Y(I)-AUX(4,I))
      DELT=.06666666666666667D0*DELT
      IF(DELT-PRMT(4))121,121,119
  119 IF(IHLF-10)112,120,120
C
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.
  120 IHLF=11
      X=X+H
      GOTO 104
C
C     SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS
  121 X=X+H
      ISW2=6
      GOTO 1
  122 DO 123 I=1,NDIM
      AUX(3,I)=Y(I)
  123 AUX(10,I)=DERY(I)
      N=3
      ISW1=4
      GOTO 200
C
  124 N=1
      X=X+H
      ISW2=7
      GOTO 1
  125 X=PRMT(1)
      DO 126 I=1,NDIM
      AUX(11,I)=DERY(I)
 1260Y(I)=AUX(1,I)+H*(.375D0*AUX(8,I)+.7916666666666667D0*AUX(9,I)
     1-.2083333333333333333D0*AUX(10,I)+.04166666666666667D0*DERY(I))
  127 X=X+H
      N=N+1
      ISW2=12
      GOTO 1
  128 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))107,129,107
  129 IF(N-4)130,300,300
  130 DO 131 I=1,NDIM
      AUX(N,I)=Y(I)
  131 AUX(N+7,I)=DERY(I)
      IF(N-3)132,134,300
C
  132 DO 133 I=1,NDIM
      DELT=AUX(9,I)+AUX(9,I)
      DELT=DELT+DELT
  133 Y(I)=AUX(1,I)+.3333333333333333333D0*H*(AUX(8,I)+DELT+AUX(10,I))
      GOTO 127
C
  134 DO 135 I=1,NDIM
      DELT=AUX(9,I)+AUX(10,I)
      DELT=DELT+DELT+DELT
  135 Y(I)=AUX(1,I)+.375D0*H*(AUX(8,I)+DELT+AUX(11,I))
      GOTO 127
C
C     THE FOLLOWING PART OF SUBROUTINE DHPCL COMPUTES BY MEANS OF
C     RUNGE-KUTTA METHOD STARTING VALUES FOR THE NOT SELF-STARTING
C     PREDICTOR-CORRECTOR METHOD.
  200 Z=X
      DO 201 I=1,NDIM
      X=H*AUX(N+7,I)
      AUX(5,I)=X
  201 Y(I)=AUX(N,I)+.4D0*X
C     X IS AN AUXILIARY STORAGE LOCATION
C
      X=Z+.4D0*H
      ISW2=2
      GOTO 1
  202 DO 203 I=1,NDIM
      X=H*DERY(I)
      AUX(6,I)=X
  203 Y(I)=AUX(N,I)+.2969776092477536 0D0*AUX(5,I)+.1587596449710358 3D0*X
C
      X=Z+.4557372542187894 3D0*H
      ISW2=3
      GOTO 1
  204 DO 205 I=1,NDIM
      X=H*DERY(I)
      AUX(7,I)=X
  205 Y(I)=AUX(N,I)+.2181003882259204 7D0*AUX(5,I)-3.0509651486929308D0*
     1AUX(6,I)+3.8328647604670103D0*X
C
      X=Z+H
      ISW2=4
      GOTO 1
  206 DO 207 I=1,NDIM
  207CY(I)=AUX(N,I)+.1747602822626903 7D0*AUX(5,I)-.5514806628787329 4D0*
     1AUX(6,I)+1.2055355993965235D0*AUX(7,I)+.1711847812195190 3D0*
     2H*DERY(I)
      X=Z
      GOTO(110,114,117,124),ISW1
C
C     POSSIBLE BREAK-POINT FOR LINKAGE
C
C     STARTING VALUES ARE COMPUTED.
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.
  300 ISTEP=3
  301 IF(N-8)304,302,304
C
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS
  302 DO 303 N=2,7
      DO 303 I=1,NDIM
      AUX(N-1,I)=AUX(N,I)
  303 AUX(N+6,I)=AUX(N+7,I)
      N=7
C
C     N LESS THAN 8 CAUSES N+1 TO GET N
  304 N=N+1
C
C     COMPUTATION OF NEXT VECTOR Y
      DO 305 I=1,NDIM
      AUX(N-1,I)=Y(I)
  305 AUX(N+6,I)=DERY(I)
      X=X+H
  306 ISTEP=ISTEP+1
      DO 307 I=1,NDIM
     0DELT=AUX(N-4,I)+1.3333333333333333D0*H*(AUX(N+6,I)+AUX(N+6,I)-
     1AUX(N+5,I)+AUX(N+4,I)+AUX(N+4,I))
      Y(I)=DELT-.9256198347107438 00*AUX(16,I)
  307 AUX(16,I)=DELT
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX, MODIFIED PREDICTOR
C     IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.
      ISW2=8
      GOTO 1
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY
C
  308 DO 309 I=1,NDIM
     0DELT=.125D0*(9.D0*AUX(N-1,I)-AUX(N-3,I)+3.D0*H*(DERY(I)+AUX(N+6,I)
     1+AUX(N+6,I)-AUX(N+5,I)))
      AUX(16,I)=AUX(16,I)-DELT
  309 Y(I)=DELT+.0743801652892562 0D0*AUX(16,I)
C
C     TEST WHETHER H MUST BE HALVED OR DOUBLED
      DELT=0.D0
      DO 310 I=1,NDIM
  310 DELT=DELT+AUX(15,I)*DABS(AUX(16,I))
      IF(DELT-PRMT(4))311,324,324
C
C     H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.
  311 ISW2=9
      GOTO 1
  312 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)
      IF(PRMT(5))314,313,314
  313 IF(IHLF-11)315,314,314
  314 RETURN
  315 IF(H*(X-PRMT(2)))316,314,314
  316 IF(DABS(X-PRMT(2))-.1D0*DABS(H))314,317,317
  317 IF(DELT-.02D0*PRMT(4))318,318,301
C
C     H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE
C     AVAILABLE
  318 IF(IHLF)301,301,319
  319 IF(N-7)301,320,320
  320 IF(ISTEP-4)301,321,321
  321 IMOD=ISTEP/2
      IF(ISTEP-IMOD-IMOD)301,322,301
  322 H=H+H
      IHLF=IHLF-1
      ISTEP=0
      DO 323 I=1,NDIM
      AUX(N-1,I)=AUX(N-2,I)
      AUX(N-2,I)=AUX(N-4,I)
      AUX(N-3,I)=AUX(N-6,I)
      AUX(N+6,I)=AUX(N+5,I)
      AUX(N+5,I)=AUX(N+3,I)
```

```
      AUX(N+4,I)=AUX(N+1,I)                                             DHCL3640
      DELT=AUX(N+6,I)+AUX(N+5,I)                                        DHCL3650
      DELT=DELT+DELT+DELT                                               DHCL3660
 3230AUX(16,I)=8.962962962962963D0*(Y(I)-AUX(N-3,I))                    DHCL3670
     1-3.3611111111111111D0*H*(DERY(I)+DELT+AUX(N+4,I))                 DHCL3680
      GOTO 301                                                          DHCL3690
C                                                                       DHCL3700
C     H MUST BE HALVED                                                  DHCL3710
  324 IHLF=IHLF+1                                                       DHCL3720
      IF(IHLF-10)325,325,311                                            DHCL3730
  325 H=.5D0*H                                                          DHCL3740
      ISTEP=0                                                           DHCL3750
      DO 326 I=1,NDIM                                                   DHCL3760
      OY(I)=.3906625D-2*(8.D1*AUX(N-1,I)+135.D0*AUX(N-2,I)+4.D1*AUX(N-3,I)DHCL3770
     1+AUX(N-4,I))-.1171875D0*(AUX(N+6,I)-6.D0*AUX(N+5,I)-AUX(N+4,I))*H DHCL3780
      OAUX(N-4,I)=.3906625D-2*(12.D0*AUX(N-1,I)+135.D0*AUX(N-2,I)+      DHCL3790
     1108.D0*AUX(N-3,I)+AUX(N-4,I))-.0234375D0*(AUX(N+6,I)+             DHCL3800
     218.D0*AUX(N+5,I)-9.D0*AUX(N+4,I))*H                               DHCL3810
      AUX(N-3,I)=AUX(N-2,I)                                             DHCL3820
  326 AUX(N+4,I)=AUX(N+5,I)                                             DHCL3830
      DELT=X-H                                                          DHCL3840
      X=DELT-(H+H)                                                      DHCL3850
      ISW2=10                                                           DHCL3860
      GOTO 1                                                            DHCL3870
  327 DO 328 I=1,NDIM                                                   DHCL3880
      AUX(N-2,I)=Y(I)                                                   DHCL3890
      AUX(N+5,I)=DERY(I)                                                DHCL3900
  328 Y(I)=AUX(N-4,I)                                                   DHCL3910
      X=X-(H+H)                                                         DHCL3920
      ISW2=11                                                           DHCL3930
      GOTO 1                                                            DHCL3940
  329 X=DELT                                                            DHCL3950
      DO 330 I=1,NDIM                                                   DHCL3960
      DELT=AUX(N+5,I)+AUX(N+4,I)                                        DHCL3970
      DELT=DELT+DELT+DELT                                               DHCL3980
      OAUX(16,I)=8.962962962962963D0*(AUX(N-1,I)-Y(I))                  DHCL3990
     1-3.3611111111111111D0*H*(AUX(N+6,I)+DELT+DERY(I))                 DHCL4000
  330 AUX(N+3,I)=DERY(I)                                                DHCL4010
      GOTO 306                                                          DHCL4020
      END                                                               DHCL4030
```

Subroutines LBVP and DLBVP

These subroutines generate an approximate solution of a given linear boundary-value problem consisting of a system of linear first-order ordinary differential equations with linear boundary conditions. Solution is done by the method of adjoint equations, which generates successively adjoint initial-value problems. By means of the solutions of these adjoint initial-value problems (done by Hamming's modified predictor-corrector method) it is possible to generate missing boundary conditions, which are necessary to compute initial values suitable to the given boundary-value problem (done by subroutine GELG or DGELG: solution of simultaneous linear equations by means of Gauss elimination). Thus the boundary-value problem is reduced to a linear initial-value problem, which is finally solved by Hamming's modified predictor-corrector method also.

Given the system of n linear first-order ordinary differential equations:

$$\left.\begin{aligned}
y'_1 \equiv \frac{dy_1}{dx} &= a_{11}(x)y_1 + a_{12}(x)y_2 + \dots \\
&\quad + a_{1n}(x)y_n + f_1(x) \\
y'_2 \equiv \frac{dy_2}{dx} &= a_{21}(x)y_1 + a_{22}(x)y_2 + \dots \\
&\quad + a_{2n}(x)y_n + f_2(x) \\
&\dots\dots \\
y'_n \equiv \frac{dy_n}{dx} &= a_{n1}(x)y_1 + a_{n2}(x)y_2 + \dots \\
&\quad + a_{nn}(x)y_n + f_n(x)
\end{aligned}\right\}$$

together with n boundary conditions:

$$\left.\begin{aligned}
&b_{11}y_1(x_l)+b_{12}y_2(x_l)+\dots+b_{1n}y_n(x_l)+c_{11}y_1(x_u) \\
&\quad +c_{12}y_2(x_u)+\dots+c_{1n}y_n(x_u) = r_1 \\
&b_{21}y_1(x_l)+b_{22}y_2(x_l)+\dots+b_{2n}y_n(x_l)+c_{21}y_1(x_u) \\
&\quad +c_{22}y_2(x_u)+\dots+c_{2n}y_n(x_u) = r_2 \\
&\dots\dots \\
&b_{n1}y_1(x_l)+b_{n2}y_2(x_l)+\dots+b_{nn}y_n(x_l)+c_{n1}y_1(x_u) \\
&\quad +c_{n2}y_2(x_u)+\dots+c_{nn}y_n(x_u) = r_n
\end{aligned}\right\}$$

where x_l and x_u are the lower and upper bound respectively of the integration interval, and using the following vector and matrix notations:

$$Y(x) = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_n(x) \end{pmatrix}, \quad F(x) = \begin{pmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_n(x) \end{pmatrix},$$

$$Y_l = \begin{pmatrix} y_1(x_l) \\ y_2(x_l) \\ \vdots \\ y_n(x_l) \end{pmatrix}, \quad Y_u = \begin{pmatrix} y_1(x_u) \\ y_2(x_u) \\ \vdots \\ y_n(x_u) \end{pmatrix},$$

$$R = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{pmatrix},$$

$$A(x) = \begin{pmatrix} a_{11}(x) a_{12}(x) \ldots a_{1n}(x) \\ a_{21}(x) a_{22}(x) \ldots a_{2n}(x) \\ \ldots \\ a_{n1}(x) a_{n2}(x) \ldots a_{nn}(x) \end{pmatrix}$$

$$B = \begin{pmatrix} b_{11} b_{12} \ldots b_{1n} \\ b_{21} b_{22} \ldots b_{2n} \\ \ldots \\ b_{n1} b_{n2} \ldots b_{nn} \end{pmatrix} \quad \text{and}$$

$$C = \begin{pmatrix} c_{11} c_{12} \ldots c_{1n} \\ c_{21} c_{22} \ldots c_{2n} \\ \ldots \\ c_{n1} c_{n2} \ldots c_{nn} \end{pmatrix}$$

where $Y(x)$, $F(x)$, Y_l, Y_u and R are column vectors with n components, and $A(x)$, and B and C are n by n square matrices, the given problem appears as follows:

$$Y' \equiv \frac{dY}{dx} = A(x) * Y + F(x) \tag{1}$$

with

$$B * Y_l + C * Y_u = R \tag{2}$$

The first problem is to generate new boundary conditions. Assuming that B has fewer zero-columns than C†, the procedure eliminates Y_u from the boundary conditions and generates a system of simultaneous linear equations for the unknown initial values $y_1(x_l)$, $y_2(x_l)$, ..., $y_n(x_l)$. This is done in the following way.

Assuming that the kth column of matrix C (k = 1, 2, ..., n) contains at least one nonzero element (that is, $y_k(x_u)$ appears in the boundary conditions), the following adjoint linear initial-value problem is generated and solved by Hamming's modified predictor-corrector method:

$$\left.\begin{aligned}
z'_1(x) &\equiv \frac{dz_1}{dx} = -a_{11}(x) z_1 - a_{21}(x) z_2 - \ldots \\
&\quad - a_{n1}(x) z_n \\
z'_2(x) &\equiv \frac{dz_2}{dx} = -a_{12}(x) z_1 - a_{22}(x) z_2 - \ldots \\
&\quad - a_{n2}(x) z_n \\
&\ldots \\
z'_n(x) &\equiv \frac{dz_n}{dx} = -a_{1n}(x) z_1 - a_{2n}(x) z_2 - \ldots \\
&\quad - a_{nn}(x) z_n
\end{aligned}\right\}$$

with initial values:

$$z_1(x_u) = 0,\ z_2(x_u) = 0,\ \ldots,\ z_{k-1}(x_u) = 0,\ z_k(x_u) = 1,\ z_{k+1}(x_u) = 0,\ \ldots,\ z_n(x_u)$$

or in vector and matrix notations††:

$$Z' \equiv \frac{dZ}{dx} = -A^T(x) * Z \tag{3}$$

with initial values $Z(x_u) = E_k$ (4)

† If this is not the case, the procedure interchanges x_l with x_u and B with C, in order to get a minimum number of adjoint initial-value problems.

†† As vectors Z and Z' are actually stored in the same storage arrays as Y and Y', in the flowchart and the FORTRAN program they are also named Y and DERY.

where $A^T(x)$ is the transpose of $A(x)$, and E_k is the column vector with only one nonzero element; that is, 1 as k^{th} element.

Assuming that $Z_k(x)$ is the solution of this k^{th} adjoint initial-value problem, and using equations (1) and (3), then:

$$\frac{d}{dx}\left[\sum_{i=1}^{n} z_i^{(k)}(x) \cdot y_i(x)\right] = \frac{d}{dx}\left[Z_k^T(x) * Y(x)\right]$$

$$= Z_k^T(x) * Y'(x) + Z'^T(x) * Y(x)$$

$$= Z_k^T * (A * Y + F) + (-A^T * Z_k)^T * Y$$

$$= Z_k^T * A * Y + Z_k^T * F - Z_k^T * A * Y$$

$$= Z_k^T * F$$

from which it follows that:

$$\frac{d}{dx}\left[\sum_{i=1}^{n} z_i^{(k)}(x) \cdot y_i(x)\right] = \left[\sum_{i=1}^{n} z_i^{(k)}(x) \cdot f_i(x)\right]$$

Integrating this equation from x_1 up to x_u the result is:

$$\sum_{i=1}^{n} z_i^{(k)}(x_u) \cdot y_i(x_u) - \sum_{i=1}^{n} z_i^{(k)}(x_1)\, y_i(x_1)$$

$$= \int_{x_1}^{x_u} \left[\sum_{i=1}^{n} z_i^{(k)}(x) \cdot f_i(x)\right] dx$$

or:

$$y_k(x_u) = \sum_{i=1}^{n} z_i^{(k)}(x_1) \cdot y_i(x_1) - \mathrm{sum}_k(x_1) \qquad (5)$$

with:

$$\mathrm{sum}_k(x_1) = \int_{x_u}^{x_1} \left[\sum_{i=1}^{n} z_i^{(k)}(x) \cdot f_i(x)\right] dx \qquad (6)$$

Elimination of $y_k(x_u)$ in the boundary conditions can be done by the following transformation formulas, which transform matrix B_{k-1} and column vector R_{k-1} into matrix B_k and column vector R_k (with $B_0 = B$ and $R_0 = R$) on the storage locations of B and R:

$$\left.\begin{aligned} b_{ij}^{(k)} &= b_{ij}^{(k-1)} + c_{ik} \cdot z_j^{(k)}(x_1) \quad (j = 1, 2, \ldots, n) && (7) \\ r_i^{(k)} &= r_i^{(k-1)} + c_{ik} \cdot \mathrm{sum}_k(x_1) && (8) \end{aligned}\right\} (i-1, 2, \ldots, n)$$

If the kth column of matrix C contains only zero-elements (that is, $y_k(x_u)$ does not appear in the boundary conditions), elimination of $y_k(x_u)$ is not necessary, and k can at once be increased by one.

If at least all $y_k(x_u)$ ($k = 1, 2, \ldots, n$) in the boundary conditions are eliminated, the remaining system of simultaneous linear equations

$$B_n * Y_1 = R_n \qquad (9)$$

is solved by Gauss elimination (subroutine GELG or DGELG). However, this is only possible if matrix B_n is not singular. In case of a singular matrix B_n (that is, the pivot element at any elimination step is exactly equal to zero), the procedure is bypassed, giving an error message which shows that the given boundary-value problem has no or more than one solution, or that the method of adjoint equations was not able to find a solution because one or more of the generated new boundary conditions were linear combinations of the given boundary conditions.

There is another critical case in the solution of equation (9), indicated by a warning given by subroutine GELG or DGELG which means possible loss of significance in the Gauss elimination of at least as many significant digits as specified by the upper bound ε of the local truncation error in the integration procedures. If this warning appears, it is highly probable that matrix B_n is singular too. But as this may not necessarily be the case, the procedure goes on after having given a warning to the user's output subroutine in the form of a negative number of bisections of the initial step size together with the resulting initial values of the finally generated initial-value problem (first set of output values). The absolute value of this "warning parameter" indicates the step of the Gauss algorithm immediately preceding the detection of the loss of significance.

After the vector of initial values Y_1 is computed (that is after the boundary-value problem is reduced to an initial-value problem), the final step to solve the generated initial-value problem is done by

Hamming's modified predictor-corrector method, incorporated in subroutines LBVP and DLBVP.

Both the solution of adjoint linear initial-value problems described above and the solution of the finally generated linear initial-value problem equivalent to the given boundary-value problem are done by means of Hamming's modified predictor-corrector method for systems of linear differential equations. An extensive description of this method can be found in the mathematical description of subroutines HPCL and DHPCL (except that control of accuracy and adjustment of step size is done differently). Therefore it will be sufficient to give a short description of this method in the form of a notation of the formulas used.

1. Problem to be solved --
System of linear differential equations $Y' = L(x, Y)$ with initial values $Y(x_0) = Y_0$, $Y'(x_0) = L(x_0, Y_0) = Y'_0$
For adjoint initial-value problems, this is:

$$Y' \equiv L(x, Y) = -A^T(x) * Y$$

$$x_0 = x_u,\ Y(x_0) = E_k,\ Y'(x_0) = -A^T(x_0) * E_k,\ h = \frac{1}{16}(x_1 - x_u)$$

and for the finally generated initial-value problem:

$$Y' = L(x, Y) = A(x) * Y + F(x)$$

$$x_0 = x_1,\ Y_0 = Y_1,\ Y'_0 = A(x_1) * Y_1 + F(x_1)$$

2. Hamming's modified predictor-corrector formulas -- (h represents the step size adjusted to accuracy requirements) for $j = 3, 4, 5, \ldots$ (that means at points $x_j = x_3, x_4, x_5, \ldots$)

Predictor:
$$P_{j+1} = Y_{j-3} + \frac{4}{3}h\,(2\,Y'_j - Y'_{j-1} + 2\,Y'_{j-2}) \tag{10}$$

Modifier:
$$M_{j+1} = P_{j+1} - \frac{112}{121}(P_j - C_j) \tag{11}$$

$$M'_{j+1} = L(x_{j+1}, M_{j+1}) \tag{12}$$

Corrector:
$$C_{j+1} = \frac{1}{8}\left[9\,Y_j - Y_{j-2} + 3\,h \cdot (M'_{j+1} + 2Y'_j - Y'_{j-1})\right] \tag{13}$$

Final value:
$$Y_{j+1} = C_{j+1} + \frac{9}{121}(P_{j+1} - C_{j+1}) \tag{14}$$

3. Test value for control of accuracy and adjustment of step size h in predictor-corrector part ("local truncation error")†:

$$\delta = \max_i \frac{a_i \cdot |p_{j+1,\,i} - c_{j+1,\,i}|}{\max(1,\ |y_{j+1,\,i}|)} \tag{15}$$

with error weights $a_i = 1$ $(i=1,2,\ldots,n)$ for the adjoint initial-value problems, and given by input for the finally generated initial-value problem.
Adjustment of step size h: halving, if $\delta > \varepsilon$
doubling, if $\delta \leqq \varepsilon/50$
with the upper bound ε for the local truncation error given by input.

4. Formulas for interpolating intermediate points in case of halving the step size h:

$$Y_{j-1/2} \equiv Y(x_j - \frac{h}{2}) = \frac{1}{256}(80Y_j + 135Y_{j-1} + 40Y_{j-2} + Y_{j-3}) - \frac{h}{2} \cdot \frac{15}{128} \cdot (Y'_j - 6Y'_{j-1} - Y'_{j-2}) \tag{16}$$

$$Y_{j-3/2} \equiv Y(x_j - \frac{3h}{2}) = \frac{1}{256}(12Y_j + 135Y_{j-1} + 108Y_{j-2} + Y_{j-3}) - \frac{h}{2} \cdot \frac{3}{128} \cdot (Y'_j + 18Y'_{j-1} - 9Y'_{j-2}) \tag{17}$$

5. Changing vectors $P_j - C_j$ in case of halving, or $P_{j+1} - C_{j+1}$ in case of doubling the step size h:

$$P_j - C_j = \frac{242}{27}(Y_j - Y_{j-3/2}) - \frac{121}{36} \cdot \frac{h}{2}(Y'_j + 3Y'_{j-1/2} + 3Y'_{j-1} + Y'_{j-3/2}) \tag{18}$$

$$P_{j+1} - C_{j+1} = \frac{242}{27}(Y_{j+1} - Y_{j-5}) - \frac{121}{36} \cdot 2h\,(Y'_{j+1} + 3Y'_{j-1} + 3Y'_{j-3} + Y'_{j-5}) \tag{19}$$

---

† The terms from which δ is generated by searching for the maximum are weighted relative errors if the absolute values of $y_{j+1,i}(x)$ are greater than 1, and weighted absolute errors if the absolute values are not greater than 1.

6. Starting Hamming's modified predictor-corrector method by means of three Runge-Kutta steps:

For $j = 0, 1, 2$

$$K_1 = h \cdot Y'_j \tag{20}$$

$$K_2 = h \cdot L(x_j + 0.4h, Y_j + 0.4K_1) \tag{21}$$

$$K_3 = h \cdot L(x_j + 0.45573725421878943\, h, Y_j + 0.29697760924775360\, K_1 + 0.15875964497103583\, K_2) \tag{22}$$

$$K_4 = h \cdot L(x_j + h, Y_j + 0.21810038822592047\, K_1 - 3.0509651486929308\, K_2 + 3.8328647604670103\, K_3) \tag{23}$$

$$Y_{j+1} = Y_j + 0.17476028226269037\, K_1 - 0.55148066287873294\, K_2 + 1.2055355993965235\, K_3 + 0.17118478121951903\, K_4 \tag{24}$$

7. Test value for control of accuracy and adjustment of step size h in the Runge-Kutta part ("local truncation error")†:

$$\delta = \frac{1}{15} \cdot \max_i \frac{a_i \left| y^{(1)}_{2,i} - y^{(2)}_{2,i} \right|}{\max\left(1, \left| y^{(2)}_{2,i} \right|\right)} \tag{25}$$

with error weights $a_i = 1$ ($i = 1, 2, \ldots, n$) for adjoint initial-value problems and given by input for the finally generated initial-value problem. Step size h is adjusted by halving if $\delta > \varepsilon$, with the upper bound ε of the local truncation error given by input.

† The terms from which δ is generated by searching for the maximum are weighted relative errors if the absolute values of $y^{(2)}_{2,i}$ are greater than 1, and weighted absolute errors if the absolute values are not greater than 1.

8. Iteration formulas for refinement of Runge-Kutta results:

$$Y_1 = Y_0 + \frac{h}{24}(9\,Y'_0 + 19\,Y'_1 - 5\,Y'_2 + Y'_3) \tag{26}$$

$$Y_2 = Y_0 + \frac{h}{3}(Y'_0 + 4\,Y'_1 + Y'_2) \tag{27}$$

$$Y_3 = Y_0 + \frac{3h}{8}(Y'_0 + 3Y'_1 + 3Y'_2 + Y'_3) \tag{28}$$

As transformation formulas (7) and (8) show, the output of Hamming's modified predictor-corrector method for the solution of any adjoint initial-value problem must contain only the vector $Z_k(x_1)$ of function values at the lower bound x_1 (actually stored in vector Y) and the integral value $sum_k(x_1)$. With respect to the predictor-corrector method, which is a fourth-order integration procedure generating function vectors $Z_k(x)$ (in storage array Y) and derivative vectors $Z'_k(x)$ (in storage array DERY) at intermediate (not necessarily equidistant) points x, the following fourth-order Hermitian integration formula -- using the function and the first derivative values of the function to be integrated -- is suitable to accumulate integral values $sum_k(x)$ and finally $sum_k(x_1)$:

$$sum_k(x+h) = sum_k(x) + \frac{h}{2}\left\{ g_k(x) + g_k(x+h) + \frac{h}{6}\left[g'_k(x) - g'_k(x+h) \right] \right\} \tag{29}$$

using the instantaneous step size h of the predictor-corrector method, and with:

$$g_k(x) = \sum_{i=1}^{n} z_i^{(k)}(x)\, f_i(x) = Z_k^T(x) + F(x) \tag{30}$$

and:

$$g'_k(x) = \sum_{i=1}^{n} \left[\frac{dz_i^{(k)}}{dx} \cdot f_i(x) + z_i^{(k)}(x) \cdot \frac{df_i}{dx} \right]$$

$$= \frac{dZ_k^T}{dx} * F(x) + Z_k^T(x) * \frac{dF}{dx} \tag{31}$$

Therefore, generation of sum_k (x+h) is done by means of the following formulas:

$$g_u^{(k)} = g_k(x+h) = \sum_{i=1}^{n} z_i^{(k)}(x+h)\, f_i(x+h)$$

$$= Z_k^T(x+h) * F(x+h) \qquad (32)$$

$$dg_u^{(k)} = g'_k(x+h) = \sum_{i=1}^{n} \left[z'^{(k)}_i(x+h)\, f_i(x+h) + z_i^{(k)}(x+h)\, f'_i(x+h) \right] = Z'^T_k(x+h) * F(x+h) + Z_k^T(x+h) * F'(x+h) \qquad (33)$$

$$sum_k(x+h) = sum_k(x) + \frac{h}{2}\left[g_1^{(k)} + g_u^{(k)} + \frac{h}{6}\left(dg_1^{(k)} - dg_u^{(k)}\right)\right] \qquad (34)$$

After these three formulas have been evaluated, $g_u^{(k)}$ becomes $g_1^{(k)}$, and $dg_u^{(k)}$ becomes $dg_1^{(k)}$. Initial values of $g_1^{(k)}$ and $dg_1^{(k)}$ are generated, using formulas (30) and (31) at point $x = x_u$, leading to:

$$g_1^{(k)} = g_k(x_u) = f_k(x_u) \qquad (35)$$

and:

$$dg_1^{(k)} = g'_k(x_u) = f'_k(x_u) - \sum_{i=1}^{n} a_{ki}(x_u) \cdot f_i(x_u) \qquad (36)$$

The entire input of subroutines LBVP and DLBVP is:

1. Lower and upper bound x_l and x_u of the integration interval, initial step size h of the independent variable and upper bound ε (≤ 1) of the local "relative" truncation error in integration procedures
2. Coefficient matrices B and C of vectors Y_l and Y_u in boundary conditions and right-hand side vector R
3. Vector of error weights a_i (i = 1, 2, ..., n) for the local "relative" truncation error in each component of the dependent variables ($a_i \leq 1$ for all i)
4. The number n of differential equations and boundary conditions
5. As external subroutine subprograms, the computation of the right-hand side matrix A(x) of the system of linear differential equations, and the computation of the inhomogeneous vector F(x) and of its derivative F'(x); for flexibility in output, an output subroutine
6. A one-dimensional auxiliary storage array with n storage locations, an n by n auxiliary storage array, and a two-dimensional auxiliary storage array (named AUX in flowchart, FORTRAN subroutine, and mathematical description) with 20 rows and n columns

Output is done in the following way. If a set of approximations to the dependent variables Y(x) is found to be of sufficient accuracy (that is $\delta \leq \varepsilon$), it is handed -- together with x, the vector of derivatives Y' (x), the number of bisections of the initial increment, the number of differential equations, the lower and upper bound of the integration interval, the initial step size, error bound ε, and a parameter for terminating subroutine LBVP or DLBVP -- to the output subroutine. By means of this output subroutine, the user has the opportunity to choose his own output format, to handle the output values as he wants, to change the upper error bound and to terminate subroutine LBVP or DLBVP at any output point (for example, in case of possible loss of accuracy in Gauss elimination). In particular, the user is able to drop the output of some intermediate points, printing only the result values at the special points $x_0 + j \cdot h$, where j = 0, 1, 2, ..., and h is the initial step size specified by input. The user may also perform intermediate computations using the integration results before continuing the process.

For better understanding of the flowchart and of the FORTRAN program, Figure 34 shows the allocation of special intermediate result vectors within the storage array AUX.

For reference see:

(1) G.N. Lance, Numerical Methods for High Speed Computers, Iliffe, London, 1960, pp. 64 - 67.

(2) A. Ralston/H.S. Wilf, Mathematical Methods for Digital Computers, Wiley, New York/London, 1960, pp. 95 - 109.

(3) A. Ralston, "Runge-Kutta Methods with Minimum Error Bounds", MTAC, vol. 16, no. 80 (1962), pp. 431 - 437.

(4) R. Zurmühl, Praktische Mathematik für Ingenieure und Physiker, Springer, Berlin/Göttingen/Heidelberg, 1963, pp. 227 - 232.

function vector $Y_{j-6} = Y(x_{j-6})$	1. row (AUX (1) in flowchart)
function vector $Y_{j-5} = Y(x_{j-5})$	2. row (AUX (2) in flowchart)
function vector $Y_{j-4} = Y(x_{j-4})$	3. row (AUX (3) in flowchart)
function vector $Y_{j-3} = Y(x_{j-3})$	4. row (AUX (4) in flowchart)
function vector $Y_{j-2} = Y(x_{j-2})$	5. row (AUX (5) in flowchart)
function vector $Y_{j-1} = Y(x_{j-1})$	6. row (AUX (6) in flowchart)
function vector $Y_j = Y(x_j)$	7. row (AUX (7) in flowchart)
derivative vector $Y'_{j-6} = Y'(x_{j-6})$	8. row (AUX (8) in flowchart)
derivative vector $Y'_{j-5} = Y'(x_{j-5})$	9. row (AUX (9) in flowchart)
derivative vector $Y'_{j-4} = Y'(x_{j-4})$	10. row (AUX (10) in flowchart)
derivative vector $Y'_{j-3} = Y'(x_{j-3})$	11. row (AUX (11) in flowchart)
derivative vector $Y'_{j-2} = Y'(x_{j-2})$	12. row (AUX (12) in flowchart)
derivative vector $Y'_{j-1} = Y'(x_{j-1})$	13. row (AUX (13) in flowchart)
derivative vector $Y'_j = Y'(x_j)$	14. row (AUX (14) in flowchart)
vector of error weights	15. row (AUX (15) in flowchart)
vector $P_j - C_j$	16. row (AUX (16) in flowchart)
decision vector for zero-columns in Matrix C	17. row (AUX (17) in flowchart)
vector $F(x_u)$	18. row (AUX (18) in flowchart)
vector $F'(x_u)$	19. row (AUX (19) in flowchart)
right-hand side vector R	20. row (AUX (20) in flowchart)

Figure 34. Storage allocation in auxiliary storage array AUX (LBVP-DLBVP)

```
C                                                                       LBVP  10
C     ..................................................................LBVP  20
C                                                                       LBVP  30
C        SUBROUTINE LBVP                                                LBVP  40
C                                                                       LBVP  50
C        PURPOSE                                                        LBVP  60
C           TO SOLVE A LINEAR BOUNDARY VALUE PROBLEM, WHICH CONSISTS OF LBVP  70
C           A SYSTEM OF NDIM LINEAR FIRST ORDER DIFFERENTIAL EQUATIONS  LBVP  80
C                 DY/DX=A(X)*Y(X)+F(X)                                  LBVP  90
C           AND NDIM LINEAR BOUNDARY CONDITIONS                         LBVP 100
C                 B*Y(XL)+C*Y(XU)=R.                                    LBVP 110
C                                                                       LBVP 120
C        USAGE                                                          LBVP 130
C           CALL LBVP (PRMT,B,C,R,Y,DERY,NDIM,IHLF,AFCT,FCT,DFCT,OUTP,  LBVP 140
C                 AUX,A)                                                LBVP 150
C           PARAMETERS AFCT,FCT,DFCT,OUTP REQUIRE AN EXTERNAL STATEMENT.LBVP 160
C                                                                       LBVP 170
C        DESCRIPTION OF PARAMETERS                                      LBVP 180
C           PRMT    - AN INPUT AND OUTPUT VECTOR WITH DIMENSION GREATER LBVP 190
C                     OR EQUAL TO 5, WHICH SPECIFIES THE PARAMETERS OF  LBVP 200
C                     THE INTERVAL AND OF ACCURACY AND WHICH SERVES FOR LBVP 210
C                     COMMUNICATION BETWEEN OUTPUT SUBROUTINE (FURNISHED LBVP 220
C                     BY THE USER) AND SUBROUTINE LBVP.                 LBVP 230
C                     THE COMPONENTS ARE                                LBVP 240
C           PRMT(1)- LOWER BOUND XL OF THE INTERVAL (INPUT),            LBVP 250
C           PRMT(1)- UPPER BOUND XU OF THE INTERVAL (INPUT),            LBVP 260
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE      LBVP 270
C                     (INPUT),                                          LBVP 280
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF RELATIVE ERROR IS    LBVP 290
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.      LBVP 300
C                     IF INCREMENT IS LESS THAN PRMT(3) AND RELATIVE    LBVP 310
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.LBVP 320
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS       LBVP 330
C                     OUTPUT SUBROUTINE.                                LBVP 340
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE LBVP INITIALIZES    LBVP 350
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE         LBVP 360
C                     SUBROUTINE LBVP AT ANY OUTPUT POINT, HE HAS TO    LBVP 370
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE LBVP 380
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE       LBVP 390
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER      LBVP 400
C                     THAN 5. HOWEVER SUBROUTINE LBVP DOES NOT REQUIRE  LBVP 410
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL  LBVP 420
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM     LBVP 430
C                     (CALLING LBVP) WHICH ARE OBTAINED BY SPECIAL      LBVP 440
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP.LBVP 450
C           B       - AN NDIM BY NDIM INPUT MATRIX.  (DESTROYED)        LBVP 460
C                     IT IS THE COEFFICIENT MATRIX OF Y(XL) IN          LBVP 470
C                     THE BOUNDARY CONDITIONS.                          LBVP 480
C           C       - AN NDIM BY NDIM INPUT MATRIX (POSSIBLY DESTROYED). LBVP 490
C                     IT IS THE COEFFICIENT MATRIX OF Y(XU) IN          LBVP 500
C                     THE BOUNDARY CONDITIONS.                          LBVP 510
C
C           R       - AN INPUT VECTOR WITH DIMENSION NDIM.  (DESTROYED) LBVP 520
C                     IT SPECIFIES THE RIGHT HAND SIDE OF THE           LBVP 530
C                     BOUNDARY CONDITIONS.                              LBVP 540
C           Y       - AN AUXILIARY VECTOR WITH DIMENSION NDIM.          LBVP 550
C                     IT IS USED AS STORAGE LOCATION FOR THE RESULTING  LBVP 560
C                     VALUES OF DEPENDENT VARIABLES COMPUTED AT         LBVP 570
C                     INTERMEDIATE POINTS.                              LBVP 580
C           DERY    - INPUT VECTOR OF ERROR WEIGHTS.  (DESTROYED)       LBVP 590
C                     ITS MAXIMAL COMPONENT SHOULD BE EQUAL TO 1.       LBVP 600
C                     LATERON DERY IS THE VECTOR OF DERIVATIVES, WHICH  LBVP 610
C                     BELONG TO FUNCTION VALUES Y AT INTERMEDIATE POINTS.LBVP 620
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF     LBVP 630
C                     DIFFERENTIAL EQUATIONS IN THE SYSTEM.             LBVP 640
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF    LBVP 650
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS LBVP 660
C                     GREATER THAN 10, SUBROUTINE LBVP RETURNS WITH     LBVP 670
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.          LBVP 680
C                     ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE  LBVP 690
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-LBVP 700
C                     PRMT(1)) RESPECTIVELY. FINALLY ERROR MESSAGE      LBVP 710
C                     IHLF=14 INDICATES, THAT THERE IS NO SOLUTION OR   LBVP 720
C                     THAT THERE ARE MORE THAN ONE SOLUTION OF THE      LBVP 730
C                     PROBLEM.                                          LBVP 740
C                     A NEGATIVE VALUE OF IHLF HANDED TO SUBROUTINE OUTP LBVP 750
C                     TOGETHER WITH INITIAL VALUES OF FINALLY GENERATED LBVP 760
C                     INITIAL VALUE PROBLEM INDICATES, THAT THERE WAS   LBVP 770
C                     POSSIBLE LOSS OF SIGNIFICANCE IN THE SOLUTION OF  LBVP 780
C                     THE SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS FOR   LBVP 790
C                     THESE INITIAL VALUES. THE ABSOLUTE VALUE OF IHLF  LBVP 800
C                     SHOWS, AFTER WHICH ELIMINATION STEP OF GAUSS      LBVP 810
C                     ALGORITHM POSSIBLE LOSS OF SIGNIFICANCE WAS       LBVP 820
C                     DETECTED.                                         LBVP 830
C           AFCT    - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT       LBVP 840
C                     COMPUTES THE COEFFICIENT MATRIX A OF VECTOR Y ON  LBVP 850
C                     THE RIGHT HAND SIDE OF THE SYSTEM OF DIFFERENTIAL LBVP 860
C                     EQUATIONS FOR A GIVEN X-VALUE. ITS PARAMETER LIST LBVP 870
C                     MUST BE X,A. SUBROUTINE AFCT SHOULD NOT DESTROY X. LBVP 880
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT       LBVP 890
C                     COMPUTES VECTOR F (INHOMOGENEOUS PART OF THE      LBVP 900
C                     RIGHT HAND SIDE OF THE SYSTEM OF DIFFERENTIAL     LBVP 910
C                     EQUATIONS) FOR A GIVEN X-VALUE. ITS PARAMETER LIST LBVP 920
C                     MUST BE X,F. SUBROUTINE FCT SHOULD NOT DESTROY X. LBVP 930
C           DFCT    - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT       LBVP 940
C                     COMPUTES VECTOR DF (DERIVATIVE OF THE INHOMOGENEOUSLBVP 950
C                     PART ON THE RIGHT HAND SIDE OF THE SYSTEM OF      LBVP 960
C                     DIFFERENTIAL EQUATIONS) FOR A GIVEN X-VALUE. ITS  LBVP 970
C                     PARAMETER LIST MUST BE X,DF. SUBROUTINE DFCT      LBVP 980
C                     SHOULD NOT DESTROY X.                             LBVP 990
C           OUTP    - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.   LBVP1000
C                     ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT.LBVP1010
C                     NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,   LBVP1020
C                     PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY         LBVP1030
C                     SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO,LBVP1040
C                     SUBROUTINE LBVP IS TERMINATED.                    LBVP1050
C           AUX     - AN AUXILIARY STORAGE ARRAY WIRH 20 ROWS AND       LBVP1060
C                     NDIM COLUMNS.                                     LBVP1070
C           A       - AN NDIM BY NDIM MATRIX, WHICH IS USED AS AUXILIARY LBVP1080
C                     STORAGE ARRAY.                                    LBVP1090
C                                                                       LBVP1100
C        REMARKS                                                        LBVP1110
C           THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF LBVP1120
C           (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE    LBVP1130
C               NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE   LBVP1140
C               IHLF=11),                                               LBVP1150
C           (2) INITIAL INCREMENT IS EQUAL TO 0 OR IF IT HAS WRONG SIGN LBVP1160
C               (ERROR MESSAGES IHLF=12 OR IHLF=13),                    LBVP1170
C           (3) THERE IS NO OR MORE THAN ONE SOLUTION OF THE PROBLEM    LBVP1180
C               (ERROR MESSAGE IHLF=14),                                LBVP1190
C           (4) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,       LBVP1200
C           (5) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.        LBVP1210
C                                                                       LBVP1220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  LBVP1230
C           SUBROUTINE GELG      SYSTEM OF LINEAR EQUATIONS.            LBVP1240
C           THE EXTERNAL SUBROUTINES AFCT(X,A), FCT(X,F), DFCT(X,DF),   LBVP1250
C           AND OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED         LBVP1260
C           BY THE USER.                                                LBVP1270
C                                                                       LBVP1280
C        METHOD                                                         LBVP1290
C           EVALUATION IS DONE USING THE METHOD OF ADJOINT EQUATIONS.   LBVP1300
C           HAMMINGS FOURTH ORDER MODIFIED PREDICTOR-CORRECTOR METHOD   LBVP1310
C           IS USED TO SOLVE THE ADJOINT INITIAL VALUE PROBLEMS AND FI- LBVP1320
C           NALLY TO SOLVE THE GENERATED INITIAL VALUE PROBLEM FOR Y(X).LBVP1330
C           THE INITIAL INCREMENT PRMT(3) IS AUTOMATICALLY ADJUSTED.    LBVP1340
C           FOR COMPUTATION OF INTEGRAL SUM, A FOURTH ORDER HERMITEAN   LBVP1350
C           INTEGRATION FORMULA IS USED.                                LBVP1360
C           FOR REFERENCE, SEE                                          LBVP1370
C           (1) LANCE, NUMERICAL METHODS FOR HIGH SPEED COMPUTERS,      LBVP1380
C               ILIFFE, LONDON, 1960, PP.64-67.                         LBVP1390
C           (2) RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL          LBVP1400
C               COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.     LBVP1410
C           (3) RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS, LBVP1420
C               MTAC, VOL.16, ISS.80 (1962), PP.431-437.                LBVP1430
C           (4) ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND     LBVP1440
C               PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963, LBVP1450
C               PP.227-232.                                             LBVP1460
C                                                                       LBVP1470
C     ..................................................................LBVP1480
C                                                                       LBVP1490
      OSUBROUTINE LBVP(PRMT,B,C,R,Y,DERY,NDIM,IHLF,AFCT,FCT,DFCT,OUTP,  LBVP1500
     1AUX,A)                                                            LBVP1510
C                                                                       LBVP1520
      DIMENSION PRMT(1),B(1),C(1),R(1),Y(1),DERY(1),AUX(20,1),A(1)      LBVP1530
C                                                                       LBVP1540
C     ERROR TEST                                                        LBVP1550
      IF(PRMT(3)*(PRMT(2)-PRMT(1)))2,1,3                                LBVP1560
    1 IHLF=12                                                           LBVP1570
      RETURN                                                            LBVP1580
    2 IHLF=13                                                           LBVP1590
      RETURN                                                            LBVP1600
C                                                                       LBVP1610
C     SEARCH FOR ZERO-COLUMNS IN MATRICES B AND C                       LBVP1620
    3 KK=-NDIM                                                          LBVP1630
      IB=0                                                              LBVP1640
      IC=0                                                              LBVP1650
      DO 7 K=1,NDIM                                                     LBVP1660
      AUX(15,K)=DERY(K)                                                 LBVP1670
      AUX(1,K)=1.                                                       LBVP1680
      AUX(17,K)=1.                                                      LBVP1690
      KK=KK+NDIM                                                        LBVP1700
      DO 4 I=1,NDIM                                                     LBVP1710
      II=KK+I                                                           LBVP1720
      IF(B(II))5,4,5                                                    LBVP1730
    4 CONTINUE                                                          LBVP1740
      IB=IB+1                                                           LBVP1750
      AUX(1,K)=0.                                                       LBVP1760
    5 DO 6 I=1,NDIM                                                     LBVP1770
      II=KK+I                                                           LBVP1780
      IF(C(II))7,6,7                                                    LBVP1790
    6 CONTINUE                                                          LBVP1800
```

```
      IC=IC+1                                                           LBVP1810
      AUX(17,K)=0.                                                      LBVP1820
    7 CONTINUE                                                          LBVP1830
C                                                                       LBVP1840
C     DETERMINATION OF LOWER AND UPPER BOUND                            LBVP1850
      IF(IC-IB)8,11,11                                                  LBVP1860
    8 H=PRMT(2)                                                         LBVP1870
      PRMT(2)=PRMT(1)                                                   LBVP1880
      PRMT(1)=H                                                         LBVP1890
      PRMT(3)=-PRMT(3)                                                  LBVP1900
      DO 9 I=1,NDIM                                                     LBVP1910
    9 AUX(17,I)=AUX(1,I)                                                LBVP1920
      II=NDIM*NDIM                                                      LBVP1930
      DO 10 I=1,II                                                      LBVP1940
      H=B(I)                                                            LBVP1950
      B(I)=C(I)                                                         LBVP1960
   10 C(I)=H                                                            LBVP1970
C                                                                       LBVP1980
C     PREPARATIONS FOR CONSTRUCTION OF ADJOINT INITIAL VALUE PROBLEMS   LBVP1990
   11 X=PRMT(2)                                                         LBVP2000
      CALL FCT(X,Y)                                                     LBVP2010
      CALL DFCT(X,DERY)                                                 LBVP2020
      DO 12 I=1,NDIM                                                    LBVP2030
      AUX(18,I)=Y(I)                                                    LBVP2040
   12 AUX(19,I)=DERY(I)                                                 LBVP2050
C                                                                       LBVP2060
C     POSSIBLE BREAK-POINT FOR LINKAGE                                  LBVP2070
C                                                                       LBVP2080
C     THE FOLLOWING PART OF SUBROUTINE LBVP UNTIL NEXT BREAK-POINT FOR  LBVP2090
C     LINKAGE HAS TO REMAIN IN CORE DURING THE WHOLE REST OF THE        LBVP2100
C     COMPUTATIONS                                                      LBVP2110
C                                                                       LBVP2120
C     START LOOP FOR GENERATING ADJOINT INITIAL VALUE PROBLEMS          LBVP2130
      K=0                                                               LBVP2140
      KK=0                                                              LBVP2150
  100 K=K+1                                                             LBVP2160
      IF(AUX(17,K))108,108,101                                          LBVP2170
C                                                                       LBVP2180
C     INITIALIZATION OF ADJOINT INITIAL VALUE PROBLEM                   LBVP2190
  101 X=PRMT(2)                                                         LBVP2200
      CALL AFCT(X,A)                                                    LBVP2210
      SUM=0.                                                            LBVP2220
      GL=AUX(18,K)                                                      LBVP2230
      DGL=AUX(19,K)                                                     LBVP2240
      II=K                                                              LBVP2250
      DO 104 I=1,NDIM                                                   LBVP2260
      H=-A(II)                                                          LBVP2270
      DERY(I)=H                                                         LBVP2280
      AUX(20,I)=R(I)                                                    LBVP2290
      Y(I)=0.                                                           LBVP2300
      IF(I-K)103,102,103                                                LBVP2310
  102 Y(I)=1.                                                           LBVP2320
  103 DGL=DGL+H*AUX(18,I)                                               LBVP2330
  104 II=II+NDIM                                                        LBVP2340
      XEND=PRMT(1)                                                      LBVP2350
      H=.0625*(XEND-X)                                                  LBVP2360
      ISW=0                                                             LBVP2370
      GOTO 400                                                          LBVP2380
C     THIS IS BRANCH TO ADJOINT LINEAR INITIAL VALUE PROBLEM            LBVP2390
C                                                                       LBVP2400
C     THIS IS RETURN FROM ADJOINT LINEAR INITIAL VALUE PROBLEM          LBVP2410
  105 IF(IHLF-10)106,106,117                                            LBVP2420
C                                                                       LBVP2430
C     UPDATING OF COEFFICIENT MATRIX B AND VECTOR R                     LBVP2440
  106 DO 107 I=1,NDIM                                                   LBVP2450
      KK=KK+1                                                           LBVP2460
      H=C(KK)                                                           LBVP2470
      R(I)=AUX(20,I)+H*SUM                                              LBVP2480
      II=I                                                              LBVP2490
      DO 107 J=1,NDIM                                                   LBVP2500
      B(II)=B(II)+H*Y(J)                                                LBVP2510
  107 II=II+NDIM                                                        LBVP2520
      GOTO 109                                                          LBVP2530
  108 KK=KK+NDIM                                                        LBVP2540
  109 IF(K-NDIM)100,110,110                                             LBVP2550
C                                                                       LBVP2560
C     GENERATION OF LAST INITIAL VALUE PROBLEM                          LBVP2570
  110 X=PRMT(4)                                                         LBVP2580
      CALL GELG(R,B,NDIM,1,X,I)                                         LBVP2590
      IF(I)111,112,112                                                  LBVP2600
  111 IHLF=14                                                           LBVP2610
      RETURN                                                            LBVP2620
C                                                                       LBVP2630
  112 PRMT(5)=0.                                                        LBVP2640
      IHLF=-I                                                           LBVP2650
      X=PRMT(1)                                                         LBVP2660
      XEND=PRMT(2)                                                      LBVP2670
      H=PRMT(3)                                                         LBVP2680
      DO 113 I=1,NDIM                                                   LBVP2690
  113 Y(I)=R(I)                                                         LBVP2700
      ISW=1                                                             LBVP2710
  114 ISW2=12                                                           LBVP2720
      GOTO 200                                                          LBVP2730
  115 ISW3=-1                                                           LBVP2740
      GOTO 300                                                          LBVP2750
  116 IF(IHLF)400,400,117                                               LBVP2760
C     THIS WAS BRANCH INTO INITIAL VALUE PROBLEM                        LBVP2770
C                                                                       LBVP2780
C     THIS IS RETURN FROM INITIAL VALUE PROBLEM                         LBVP2790
  117 RETURN                                                            LBVP2800
C                                                                       LBVP2810
C     THIS PART OF LINEAR BOUNDARY VALUE PROBLEM COMPUTES THE RIGHT     LBVP2820
C     HAND SIDE DERY OF THE SYSTEM OF ADJOINT LINEAR DIFFERENTIAL       LBVP2830
C     EQUATIONS (IN CASE ISW=0) OR OF THE GIVEN SYSTEM (IN CASE ISW=1). LBVP2840
  200 CALL AFCT(X,A)                                                    LBVP2850
      IF(ISW)201,201,205                                                LBVP2860
C                                                                       LBVP2870
C     ADJOINT SYSTEM                                                    LBVP2880
  201 LL=0                                                              LBVP2890
      DO 203 M=1,NDIM                                                   LBVP2900
      HS=0.                                                             LBVP2910
      DO 202 L=1,NDIM                                                   LBVP2920
      LL=LL+1                                                           LBVP2930
  202 HS=HS-A(LL)*Y(L)                                                  LBVP2940
  203 DERY(M)=HS                                                        LBVP2950
  204 GOTO(502,504,506,407,415,418,608,617,632,634,421,115),ISW2        LBVP2960
C                                                                       LBVP2970
C     GIVEN SYSTEM                                                      LBVP2980
  205 CALL FCT(X,DERY)                                                  LBVP2990
      DO 207 M=1,NDIM                                                   LBVP3000
      LL=M-NDIM                                                         LBVP3010
      HS=0.                                                             LBVP3020
      DO 206 L=1,NDIM                                                   LBVP3030
      LL=LL+NDIM                                                        LBVP3040
  206 HS=HS+A(LL)*Y(L)                                                  LBVP3050
  207 DERY(M)=HS*DERY(M)                                                LBVP3060
      GOTO 204                                                          LBVP3070
C                                                                       LBVP3080
C     THIS PART OF LINEAR BOUNDARY VALUE PROBLEM COMPUTES THE VALUE OF  LBVP3090
C     INTEGRAL SUM, WHICH IS A PART OF THE OUTPUT OF ADJOINT INITIAL    LBVP3100
C     VALUE PROBLEM (IN CASE ISW=0) OR RECORDS RESULT VALUES OF THE     LBVP3110
C     FINAL INITIAL VALUE PROBLEM (IN CASE ISW=1).                      LBVP3120
  300 IF(ISW)301,301,305                                                LBVP3130
C                                                                       LBVP3140
C     ADJOINT PROBLEM                                                   LBVP3150
  301 CALL FCT(X,R)                                                     LBVP3160
      GU=0.                                                             LBVP3170
      DGU=0.                                                            LBVP3180
      DO 302 L=1,NDIM                                                   LBVP3190
      GU=GU+Y(L)*R(L)                                                   LBVP3200
  302 DGU=DGU+DERY(L)*R(L)                                              LBVP3210
      CALL DFCT(X,R)                                                    LBVP3220
      DO 303 L=1,NDIM                                                   LBVP3230
  303 DGU=DGU+Y(L)*R(L)                                                 LBVP3240
      SUM=SUM+.5*H*(IGL+GU)+.1666667*H*(DGL-DGU))                       LBVP3250
      GL=GU                                                             LBVP3260
      DGL=DGU                                                           LBVP3270
  304 IF(ISW)1116,422,618                                               LBVP3280
C                                                                       LBVP3290
C     GIVEN PROBLEM                                                     LBVP3300
  305 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                                LBVP3310
      IF(PRMT(5))117,304,117                                            LBVP3320
C                                                                       LBVP3330
C     POSSIBLE BREAK-POINT FOR LINKAGE                                  LBVP3340
C                                                                       LBVP3350
C     THE FOLLOWING PART OF SUBROUTINE LBVP SOLVES IN CASE ISW=0 THE    LBVP3360
C     ADJOINT INITIAL VALUE PROBLEM. IT COMPUTES INTEGRAL SUM AND       LBVP3370
C     THE VECTOR Y OF DEPENDENT VARIABLES AT THE LOWER BOUND PRMT(1).   LBVP3380
C     IN CASE ISW=1 IT SOLVES FINALLY GENERATED INITIAL VALUE PROBLEM.  LBVP3390
  400 N=1                                                               LBVP3400
      XST=X                                                             LBVP3410
      IHLF=0                                                            LBVP3420
      DO 401 I=1,NDIM                                                   LBVP3430
      AUX(16,I)=0.                                                      LBVP3440
      AUX(1,I)=Y(I)                                                     LBVP3450
  401 AUX(8,I)=DERY(I)                                                  LBVP3460
      ISW1=1                                                            LBVP3470
      GOTO 500                                                          LBVP3480
C                                                                       LBVP3490
  402 X=X+H                                                             LBVP3500
      DO 403 I=1,NDIM                                                   LBVP3510
  403 AUX(2,I)=Y(I)                                                     LBVP3520
C                                                                       LBVP3530
C     INCREMENT H IS TESTED BY MEANS OF BISECTION                       LBVP3540
  404 IHLF=IHLF+1                                                       LBVP3550
      X=X-H                                                             LBVP3560
      DO 405 I=1,NDIM                                                   LBVP3570
  405 AUX(4,I)=AUX(2,I)                                                 LBVP3580
      H=.5*H                                                            LBVP3590
      N=1                                                               LBVP3600
      ISW1=2                                                            LBVP3610
      GOTO 500                                                          LBVP3620
C                                                                       LBVP3630
  406 X=X+H                                                             LBVP3640
      ISW2=4                                                            LBVP3650
      GOTO 200                                                          LBVP3660
  407 N=2                                                               LBVP3670
      DO 408 I=1,NDIM                                                   LBVP3680
      AUX(2,I)=Y(I)                                                     LBVP3690
  408 AUX(9,I)=DERY(I)                                                  LBVP3700
      ISW1=3                                                            LBVP3710
      GOTO 500                                                          LBVP3720
C                                                                       LBVP3730
C     TEST ON SATISFACTORY ACCURACY                                     LBVP3740
  409 DO 414 I=1,NDIM                                                   LBVP3750
      Z=ABS(Y(I))                                                       LBVP3760
      IF(Z-1.)410,411,411                                               LBVP3770
  410 Z=1.                                                              LBVP3780
  411 DELT=.06666667*ABS(Y(I)-AUX(4,I))                                 LBVP3790
      IF(ISW)413,413,412                                                LBVP3800
  412 DELT=AUX(15,I)*DELT                                               LBVP3810
  413 IF(DELT-Z*PRMT(4))414,414,429                                     LBVP3820
  414 CONTINUE                                                          LBVP3830
C                                                                       LBVP3840
C     SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS               LBVP3850
      X=X+H                                                             LBVP3860
      ISW2=5                                                            LBVP3870
      GOTO 200                                                          LBVP3880
  415 DO 416 I=1,NDIM                                                   LBVP3890
      AUX(3,I)=Y(I)                                                     LBVP3900
  416 AUX(10,I)=DERY(I)                                                 LBVP3910
      N=3                                                               LBVP3920
      ISW1=4                                                            LBVP3930
      GOTO 500                                                          LBVP3940
C                                                                       LBVP3950
  417 N=1                                                               LBVP3960
      X=X+H                                                             LBVP3970
      ISW2=6                                                            LBVP3980
      GOTO 200                                                          LBVP3990
  418 X=XST                                                             LBVP4000
      DO 419 I=1,NDIM                                                   LBVP4010
      AUX(11,I)=DERY(I)                                                 LBVP4020
  419CY(I)=AUX(1,I)+H*(.375*AUX(8,I)+.7916667*AUX(9,I)                  LBVP4030
     1-.2083333*AUX(10,I)+.04166667*DERY(I))                            LBVP4040
  420 X=X+H                                                             LBVP4050
      N=N+1                                                             LBVP4060
      ISW2=11                                                           LBVP4070
      GOTO 200                                                          LBVP4080
  421 ISW3=0                                                            LBVP4090
      GOTO 300                                                          LBVP4100
  422 IF(N-4)423,600,600                                                LBVP4110
  423 DO 424 I=1,NDIM                                                   LBVP4120
      AUX(N,I)=Y(I)                                                     LBVP4130
  424 AUX(N+7,I)=DERY(I)                                                LBVP4140
      IF(N-3)425,427,600                                                LBVP4150
C                                                                       LBVP4160
  425 DO 426 I=1,NDIM                                                   LBVP4170
      DELT=AUX(9,I)+AUX(9,I)                                            LBVP4180
      DELT=DELT+DELT                                                    LBVP4190
  426 Y(I)=AUX(1,I)+.3333333*H*(AUX(8,I)+DELT+AUX(10,I))                LBVP4200
      GOTO 420                                                          LBVP4210
C                                                                       LBVP4220
  427 DO 428 I=1,NDIM                                                   LBVP4230
      DELT=AUX(9,I)+AUX(10,I)                                           LBVP4240
      DELT=DELT+DELT+DELT                                               LBVP4250
  428 Y(I)=AUX(1,I)+.375*H*(AUX(8,I)+DELT+AUX(11,I))                    LBVP4260
      GOTO 420                                                          LBVP4270
C                                                                       LBVP4280
C     NO SATISFACTORY ACCURACY. H MUST BE HALVED.                       LBVP4290
  429 IF(IHLF-10)404,430,430                                            LBVP4300
C                                                                       LBVP4310
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.      LBVP4320
  430 IHLF=11                                                           LBVP4330
      X=X+H                                                             LBVP4340
      IF(ISW)105,105,114                                                LBVP4350
C                                                                       LBVP4360
C     THIS PART OF LINEAR INITIAL VALUE PROBLEM COMPUTES                LBVP4370
C     STARTING VALUES BY MEANS OF RUNGE-KUTTA METHOD.                   LBVP4380
```

```
  500 Z=X                                                           L8VP4390
      DO 501 I=1,NDIM                                               L8VP4400
      X=H*AUX(N+7,I)                                                L8VP4410
      AUX(5,I)=X                                                    L8VP4420
  501 Y(I)=AUX(N,I)+.4*X                                            L8VP4430
C                                                                   L8VP4440
      X=Z+.4*H                                                      L8VP4450
      ISW2=1                                                        L8VP4460
      GOTO 200                                                      L8VP4470
  502 DO 503 I=1,NDIM                                               L8VP4480
      X=H*DERY(I)                                                   L8VP4490
      AUX(6,I)=X                                                    L8VP4500
  503 Y(I)=AUX(N,I)+.2969776*AUX(5,I)+.1587596*X                    L8VP4510
C                                                                   L8VP4520
      X=Z+.4557372*H                                                L8VP4530
      ISW2=2                                                        L8VP4540
      GOTO 200                                                      L8VP4550
  504 DO 505 I=1,NDIM                                               L8VP4560
      X=H*DERY(I)                                                   L8VP4570
      AUX(7,I)=X                                                    L8VP4580
  505 Y(I)=AUX(N,I)+.2181004*AUX(5,I)-3.050965*AUX(6,I)+3.832865*X  L8VP4590
C                                                                   L8VP4600
      X=Z+H                                                         L8VP4610
      ISW2=3                                                        L8VP4620
      GOTO 200                                                      L8VP4630
  506 DO 507 I=1,NDIM                                               L8VP4640
  5070Y(I)=AUX(N,I)+.1747603*AUX(5,I)-.5514807*AUX(6,I)             L8VP4650
     1+1.205536*AUX(7,I)+.1711848*H*DERY(I)                         L8VP4660
      X=Z                                                           L8VP4670
      GOTO(402,406,409,417),ISW1                                    L8VP4680
C                                                                   L8VP4690
C     POSSIBLE BREAK-POINT FOR LINKAGE                              L8VP4700
C                                                                   L8VP4710
C     STARTING VALUES ARE COMPUTED.                                 L8VP4720
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.       L8VP4730
  600 ISTEP=3                                                       L8VP4740
  601 IF(N-8)604,602,604                                            L8VP4750
C                                                                   L8VP4760
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS  L8VP4770
  602 DO 603 N=2,7                                                  L8VP4780
      DO 603 I=1,NDIM                                               L8VP4790
      AUX(N-1,I)=AUX(N,I)                                           L8VP4800
  603 AUX(N+6,I)=AUX(N+7,I)                                         L8VP4810
      N=7                                                           L8VP4820
C                                                                   L8VP4830
C     N LESS THAN 8 CAUSES N+1 TO GET N                             L8VP4840
  604 N=N+1                                                         L8VP4850
C                                                                   L8VP4860
C     COMPUTATION OF NEXT VECTOR Y                                  L8VP4870
      DO 605 I=1,NDIM                                               L8VP4880
      AUX(N-1,I)=Y(I)                                               L8VP4890
  605 AUX(N+6,I)=DERY(I)                                            L8VP4900
      X=X+H                                                         L8VP4910
  606 ISTEP=ISTEP+1                                                 L8VP4920
      DO 607 I=1,NDIM                                               L8VP4930
     ODELT=AUX(N-4,I)+1.333333*H*(AUX(N+6,I)+AUX(N+6,I)-AUX(N+5,I)+ L8VP4940
     1AUX(N+4,I)+AUX(N+4,I))                                        L8VP4950
      Y(I)=DELT-.9256198*AUX(16,I)                                  L8VP4960
  607 AUX(16,I)=DELT                                                L8VP4970
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX, MODIFIED PREDICTOR L8VP4980
C     IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.           L8VP4990
C                                                                   L8VP5000
      ISW2=7                                                        L8VP5010
      GOTO 200                                                      L8VP5020
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY.        L8VP5030
C                                                                   L8VP5040
  608 DO 609 I=1,NDIM                                               L8VP5050
     ODELT=.125*(9.*AUX(N-1,I)-AUX(N-3,I)+3.*H*(DERY(I)+AUX(N+6,I)+ L8VP5060
     1AUX(N+6,I)-AUX(N+5,I)))                                       L8VP5070
      AUX(16,I)=AUX(16,I)-DELT                                      L8VP5080
  609 Y(I)=DELT+.07438017*AUX(16,I)                                 L8VP5090
C                                                                   L8VP5100
C     TEST WHETHER H MUST BE HALVED OR DOUBLED                      L8VP5110
      DELT=0.                                                       L8VP5120
      DO 616 I=1,NDIM                                               L8VP5130
      Z=ABS(Y(I))                                                   L8VP5140
      IF(Z-1.)610,611,611                                           L8VP5150
  610 Z=1.                                                          L8VP5160
  611 Z=ABS(AUX(16,I))/Z                                            L8VP5170
      IF(ISW)613,613,612                                            L8VP5180
  612 Z=AUX(15,I)*Z                                                 L8VP5190
  613 IF(Z-PRMT(4))614,614,628                                      L8VP5200
  614 IF(DELT-Z)615,616,616                                         L8VP5210
  615 DELT=Z                                                        L8VP5220
  616 CONTINUE                                                      L8VP5230
C                                                                   L8VP5240
C     H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.               L8VP5250
      ISW2=8                                                        L8VP5260
      GOTO 200                                                      L8VP5270
  617 ISW3=1                                                        L8VP5280
      GOTO 300                                                      L8VP5290
  618 IF(H*(X-XEND))619,621,621                                     L8VP5300
  619 IF(ABS(X-XEND)-.1*ABS(H))621,620,620                          L8VP5310
  620 IF(DELT-.02*PRMT(4))622,622,601                               L8VP5320
  621 IF(ISW)105,105,117                                            L8VP5330
C                                                                   L8VP5340
C                                                                   L8VP5350
C     H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE     L8VP5360
C     AVAILABLE.                                                    L8VP5370
  622 IF(IHLF)601,601,623                                           L8VP5380
  623 IF(N-7)601,624,624                                            L8VP5390
  624 IF(ISTEP-4)601,625,625                                        L8VP5400
  625 IMOD=ISTEP/2                                                  L8VP5410
      IF(ISTEP-IMOD-IMOD)601,626,601                                L8VP5420
  626 H=H+H                                                         L8VP5430
      IHLF=IHLF-1                                                   L8VP5440
      ISTEP=0                                                       L8VP5450
      DO 627 I=1,NDIM                                               L8VP5460
      AUX(N-1,I)=AUX(N-2,I)                                         L8VP5470
      AUX(N-2,I)=AUX(N-4,I)                                         L8VP5480
      AUX(N-3,I)=AUX(N-6,I)                                         L8VP5490
      AUX(N+6,I)=AUX(N+5,I)                                         L8VP5500
      AUX(N+5,I)=AUX(N+3,I)                                         L8VP5510
      AUX(N+4,I)=AUX(N+1,I)                                         L8VP5520
      DELT=AUX(N+6,I)+AUX(N+5,I)                                    L8VP5530
      DELT=DELT+DELT+DELT                                           L8VP5540
  627OAUX(16,I)=8.962963*(Y(I)-AUX(N-3,I))-3.361111*H*(DERY(I)+DELT L8VP5550
     I+AUX(N+4,I))                                                  L8VP5560
      GOTO 601                                                      L8VP5570
C                                                                   L8VP5580
C                                                                   L8VP5590
C     H MUST BE HALVED                                              L8VP5600
  628 IHLF=IHLF+1                                                   L8VP5610
      IF(IHLF-10)630,630,629                                        L8VP5620
  629 IF(ISW)105,105,114                                            L8VP5630
  630 H=.5*H                                                        L8VP5640
      ISTEP=0                                                       L8VP5650
      DO 631 I=1,NDIM                                               L8VP5660
     OY(I)=.00390625*(80.*AUX(N-1,I)+135.*AUX(N-2,I)+40.*AUX(N-3,I)+ L8VP5670
     1AUX(N-4,I))-.1171875*(AUX(N+6,I)-6.*AUX(N+5,I)-AUX(N+4,I))*H  L8VP5680
     OAUX(N-4,I)=.0039062 5*(12.*AUX(N-1,I)+135.*AUX(N-2,I)+        L8VP5690
     1108.*AUX(N-3,I)+AUX(N-4,I))-.0234375*(AUX(N+6,I)+18.*AUX(N+5,I)- L8VP5700
     29.*AUX(N+4,I))*H                                              L8VP5710
      AUX(N-3,I)=AUX(N-2,I)                                         L8VP5720
  631 AUX(N+4,I)=AUX(N+5,I)                                         L8VP5730
      DELT=X-H                                                      L8VP5740
      X=DELT-(H+H)                                                  L8VP5750
      ISW2=9                                                        L8VP5760
      GOTO 200                                                      L8VP5770
  632 DO 633 I=1,NDIM                                               L8VP5780
      AUX(N-2,I)=Y(I)                                               L8VP5790
      AUX(N+5,I)=DERY(I)                                            L8VP5800
  633 Y(I)=AUX(N-4,I)                                               L8VP5810
      X=X-(H+H)                                                     L8VP5820
      ISW2=10                                                       L8VP5830
      GOTO 200                                                      L8VP5840
  634 X=DELT                                                        L8VP5850
      DO 635 I=1,NDIM                                               L8VP5860
      DELT=AUX(N+5,I)+AUX(N+4,I)                                    L8VP5870
      DELT=DELT+DELT+DELT                                           L8VP5880
     OAUX(16,I)=8.962963*(AUX(N-1,I)-Y(I))-3.361111*H*(AUX(N+6,I)+DELT L8VP5890
     1+DERY(I))                                                     L8VP5900
  635 AUX(N+3,I)=DERY(I)                                            L8VP5910
      GOTO 606                                                      L8VP5920
C                                                                   L8VP5930
C     END OF INITIAL VALUE PROBLEM                                  L8VP5940
      END                                                           L8VP5950
```

```
C                                                                   DLBV  10
C     ..................................................................DLBV  20
C                                                                   DLBV  30
C        SUBROUTINE DLBVP                                           DLBV  40
C                                                                   DLBV  50
C        PURPOSE                                                    DLBV  60
C           TO SOLVE A LINEAR BOUNDARY VALUE PROBLEM, WHICH CONSISTS OF DLBV  70
C           A SYSTEM OF NDIM LINEAR FIRST ORDER DIFFERENTIAL EQUATIONS DLBV  80
C                 DY/DX=A(X)*Y(X)+F(X)                              DLBV  90
C           AND NDIM LINEAR BOUNDARY CONDITIONS                     DLBV 100
C                 B*Y(XL)+C*Y(XU)=R.                                DLBV 110
C                                                                   DLBV 120
C        USAGE                                                      DLBV 130
C           CALL DLBVP (PRMT,B,C,R,Y,DERY,NDIM,IHLF,AFCT,FCT,DFCT,OUTP, DLBV 140
C                      AUX,A)                                       DLBV 150
C           PARAMETERS AFCT,FCT,DFCT,OUTP REQUIRE AN EXTERNAL STATEMENT.DLBV 160
C                                                                   DLBV 170
C        DESCRIPTION OF PARAMETERS                                  DLBV 180
C           PRMT    - DOUBLE PRECISION INPUT AND OUTPUT VECTOR WITH DLBV 190
C                     DIMENSION GREATER THAN OR EQUAL TO 5, WHICH   DLBV 200
C                     SPECIFIES THE PARAMETERS OF THE INTERVAL AND OF DLBV 210
C                     ACCURACY AND WHICH SERVES FOR COMMUNICATION BETWEENDLBV 220
C                     OUTPUT SUBROUTINE (FURNISHED BY THE USER) AND DLBV 230
C                     SUBROUTINE DLBVP. EXCEPT PRMT(5) THE COMPONENTS DLBV 240
C                     ARE NOT DESTROYED BY SUBROUTINE DLBVP AND THEY ARE DLBV 250
C           PRMT(1)- LOWER BOUND XL OF THE INTERVAL (INPUT),        DLBV 260
C           PRMT(1)- UPPER BOUND XU OF THE INTERVAL (INPUT),        DLBV 270
C           PRMT(3)- INITIAL INCREMENT OF THE INDEPENDENT VARIABLE  DLBV 280
C                     (INPUT),                                      DLBV 290
C           PRMT(4)- UPPER ERROR BOUND (INPUT). IF RELATIVE ERROR IS DLBV 300
C                     GREATER THAN PRMT(4), INCREMENT GETS HALVED.  DLBV 310
C                     IF INCREMENT IS LESS THAN PRMT(3) AND RELATIVE DLBV 320
C                     ERROR LESS THAN PRMT(4)/50, INCREMENT GETS DOUBLED.DLBV 330
C                     THE USER MAY CHANGE PRMT(4) BY MEANS OF HIS   DLBV 340
C                     OUTPUT SUBROUTINE.                            DLBV 350
C           PRMT(5)- NO INPUT PARAMETER. SUBROUTINE DLBVP INITIALIZES DLBV 360
C                     PRMT(5)=0. IF THE USER WANTS TO TERMINATE     DLBV 370
C                     SUBROUTINE DLBVP AT ANY OUTPUT POINT, HE HAS TO DLBV 380
C                     CHANGE PRMT(5) TO NON-ZERO BY MEANS OF SUBROUTINE DLBV 390
C                     OUTP. FURTHER COMPONENTS OF VECTOR PRMT ARE   DLBV 400
C                     FEASIBLE IF ITS DIMENSION IS DEFINED GREATER  DLBV 410
C                     THAN 5. HOWEVER SUBROUTINE DLBVP DOES NOT REQUIRE DLBV 420
C                     AND CHANGE THEM. NEVERTHELESS THEY MAY BE USEFUL DLBV 430
C                     FOR HANDING RESULT VALUES TO THE MAIN PROGRAM DLBV 440
C                     (CALLING DLBVP) WHICH ARE OBTAINED BY SPECIAL DLBV 450
C                     MANIPULATIONS WITH OUTPUT DATA IN SUBROUTINE OUTP. DLBV 460
C           B       - DOUBLE PRECISION NDIM BY NDIM INPUT MATRIX  DLBV 470
C                     (DESTROYED). IT IS THE COEFFICIENT MATRIX OF Y(XL) DLBV 480
C                     IN THE BOUNDARY CONDITIONS.                   DLBV 490
C           C       - DOUBLE PRECISION NDIM BY NDIM INPUT MATRIX  DLBV 500
C                     (POSSIBLY DESTROYED). IT IS THE COEFFICIENT MATRIX DLBV 510
C                     OF Y(XU) IN THE BOUNDARY CONDITIONS.          DLBV 520
C           R       - DOUBLE PRECISION INPUT VECTOR WITH DIMENSION NDIM DLBV 530
C                     (DESTROYED). IT SPECIFIES THE RIGHT HAND SIDE OF DLBV 540
C                     THE BOUNDARY CONDITIONS.                      DLBV 550
C           Y       - DOUBLE PRECISION AUXILIARY VECTOR WITH      DLBV 560
C                     DIMENSION NDIM. IT IS USED AS STORAGE LOCATION DLBV 570
C                     FOR THE RESULTING VALUES OF DEPENDENT VARIABLES DLBV 580
C                     COMPUTED AT INTERMEDIATE POINTS X.            DLBV 590
C           DERY    - DOUBLE PRECISION INPUT VECTOR OF ERROR WEIGHTS DLBV 600
C                     (DESTROYED). ITS MAXIMAL COMPONENT SHOULD BE  DLBV 610
C                     EQUAL TO 1. LATERON DERY IS THE VECTOR OF     DLBV 620
C                     DERIVATIVES, WHICH BELONG TO FUNCTION VALUES Y AT DLBV 630
C                     INTERMEDIATE POINTS X.                        DLBV 640
C           NDIM    - AN INPUT VALUE, WHICH SPECIFIES THE NUMBER OF DLBV 650
C                     DIFFERENTIAL EQUATIONS IN THE SYSTEM.         DLBV 660
C           IHLF    - AN OUTPUT VALUE, WHICH SPECIFIES THE NUMBER OF DLBV 670
C                     BISECTIONS OF THE INITIAL INCREMENT. IF IHLF GETS DLBV 680
C                     GREATER THAN 10, SUBROUTINE DLBVP RETURNS WITH DLBV 690
C                     ERROR MESSAGE IHLF=11 INTO MAIN PROGRAM.      DLBV 700
C                     ERROR MESSAGE IHLF=12 OR IHLF=13 APPEARS IN CASE DLBV 710
C                     PRMT(3)=0 OR IN CASE SIGN(PRMT(3)).NE.SIGN(PRMT(2)-DLBV 720
C                     PRMT(1)) RESPECTIVELY. FINALLY ERROR MESSAGE  DLBV 730
C                     IHLF=14 INDICATES, THAT THERE IS NO SOLUTION OR DLBV 740
C                     THAT THERE ARE MORE THAN ONE SOLUTION OF THE  DLBV 750
C                     PROBLEM.                                      DLBV 760
C                     A NEGATIVE VALUE OF IHLF HANDED TO SUBROUTINE OUTP DLBV 770
C                     TOGETHER WITH INITIAL VALUES OF FINALLY GENERATED DLBV 780
C                     INITIAL VALUE PROBLEM INDICATES, THAT THERE WAS DLBV 790
C                     POSSIBLE LOSS OF SIGNIFICANCE IN THE SOLUTION OF DLBV 800
C                     THE SYSTEM OF SIMULTANEOUS LINEAR EQUATIONS FOR DLBV 810
C                     THESE INITIAL VALUES. THE ABSOLUTE VALUE OF IHLF DLBV 820
C                     SHOWS, AFTER WHICH ELIMINATION STEP OF GAUSS  DLBV 830
C                     ALGORITHM POSSIBLE LOSS OF SIGNIFICANCE WAS   DLBV 840
C                     DETECTED.                                     DLBV 850
C           AFCT    - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT   DLBV 860
C                     COMPUTES THE COEFFICIENT MATRIX A OF VECTOR Y ON DLBV 870
C                     THE RIGHT HAND SIDE OF THE SYSTEM OF DIFFERENTIAL DLBV 880
C                     EQUATIONS FOR A GIVEN X-VALUE. ITS PARAMETER LIST DLBV 890
C                     MUST BE X,A. SUBROUTINE AFCT SHOULD NOT DESTROY X. DLBV 900
C           FCT     - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT   DLBV 910
C                     COMPUTES VECTOR F (INHOMOGENEOUS PART OF THE  DLBV 920
C                     RIGHT HAND SIDE OF THE SYSTEM OF DIFFERENTIAL DLBV 930
```

```
C                 EQUATIONS) FOR A GIVEN X-VALUE. ITS PARAMETER LIST  DLBV 940
C                 MUST BE X,F. SUBROUTINE FCT SHOULD NOT DESTROY X.   DLBV 950
C        DFCT   - THE NAME OF AN EXTERNAL SUBROUTINE USED. IT         DLBV 960
C                 COMPUTES VECTOR DF (DERIVATIVE OF THE INHOMOGENEOUS DLBV 970
C                 PART ON THE RIGHT HAND SIDE OF THE SYSTEM OF        DLBV 980
C                 DIFFERENTIAL EQUATIONS) FOR A GIVEN X-VALUE. ITS    DLBV 990
C                 PARAMETER LIST MUST BE X,DF. SUBROUTINE DFCT        DLBV1000
C                 SHOULD NOT DESTROY X.                               DLBV1010
C        OUTP   - THE NAME OF AN EXTERNAL OUTPUT SUBROUTINE USED.     DLBV1020
C                 ITS PARAMETER LIST MUST BE X,Y,DERY,IHLF,NDIM,PRMT. DLBV1030
C                 NONE OF THESE PARAMETERS (EXCEPT, IF NECESSARY,     DLBV1040
C                 PRMT(4),PRMT(5),...) SHOULD BE CHANGED BY           DLBV1050
C                 SUBROUTINE OUTP. IF PRMT(5) IS CHANGED TO NON-ZERO, DLBV1060
C                 SUBROUTINE DLBVP IS TERMINATED.                     DLBV1070
C        AUX    - DOUBLE PRECISION AUXILIARY STORAGE ARRAY WITH 20   DLBV1080
C                 ROWS AND NDIM COLUMNS.                              DLBV1090
C        A      - DOUBLE PRECISION NDIM BY NDIM MATRIX, WHICH IS USED DLBV1100
C                 AS AUXILIARY STORAGE ARRAY.                         DLBV1110
C                                                                     DLBV1120
C      REMARKS                                                        DLBV1130
C        THE PROCEDURE TERMINATES AND RETURNS TO CALLING PROGRAM, IF  DLBV1140
C        (1) MORE THAN 10 BISECTIONS OF THE INITIAL INCREMENT ARE     DLBV1150
C            NECESSARY TO GET SATISFACTORY ACCURACY (ERROR MESSAGE    DLBV1160
C            IHLF=11),                                                DLBV1170
C        (2) INITIAL INCREMENT IS EQUAL TO 0 OR IF IT HAS WRONG SIGN  DLBV1180
C            (ERROR MESSAGES IHLF=12 OR IHLF=13),                     DLBV1190
C        (3) THERE IS NO OR MORE THAN ONE SOLUTION OF THE PROBLEM     DLBV1200
C            (ERROR MESSAGE IHLF=14),                                 DLBV1210
C        (4) THE WHOLE INTEGRATION INTERVAL IS WORKED THROUGH,        DLBV1220
C        (5) SUBROUTINE OUTP HAS CHANGED PRMT(5) TO NON-ZERO.         DLBV1230
C                                                                     DLBV1240
C      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DLBV1250
C        SUBROUTINE DGELG     SYSTEM OF LINEAR EQUATIONS.             DLBV1260
C        THE EXTERNAL SUBROUTINES AFCT(X,A), FCT(X,F), DFCT(X,DF),    DLBV1270
C        AND OUTP(X,Y,DERY,IHLF,NDIM,PRMT) MUST BE FURNISHED          DLBV1280
C        BY THE USER.                                                 DLBV1290
C                                                                     DLBV1300
C      METHOD                                                         DLBV1310
C        EVALUATION IS DONE USING THE METHOD OF ADJOINT EQUATIONS.    DLBV1320
C        HAMMINGS FOURTH ORDER MODIFIED PREDICTOR-CORRECTOR METHOD    DLBV1330
C        IS USED TO SOLVE THE ADJOINT INITIAL VALUE PROBLEMS AND FI-  DLBV1340
C        NALLY TO SOLVE THE GENERATED INITIAL VALUE PROBLEM FOR Y(X). DLBV1350
C        THE INITIAL INCREMENT PRMT(3) IS AUTOMATICALLY ADJUSTED.     DLBV1360
C        FOR COMPUTATION OF INTEGRAL SUM, A FOURTH ORDER HERMITEAN    DLBV1370
C        INTEGRATION FORMULA IS USED.                                 DLBV1380
C        FOR REFERENCE, SEE                                           DLBV1390
C        (1) LANCE, NUMERICAL METHODS FOR HIGH SPEED COMPUTERS,       DLBV1400
C            ILIFFE, LONDON, 1960, PP.64-67.                          DLBV1410
C        (2) RALSTON/WILF, MATHEMATICAL METHODS FOR DIGITAL           DLBV1420
C            COMPUTERS, WILEY, NEW YORK/LONDON, 1960, PP.95-109.      DLBV1430
C        (3) RALSTON, RUNGE-KUTTA METHODS WITH MINIMUM ERROR BOUNDS,  DLBV1440
C            MTAC, VOL.16, ISS.80 (1962), PP.431-437.                 DLBV1450
C        (4) ZURMUEHL, PRAKTISCHE MATHEMATIK FUER INGENIEURE UND      DLBV1460
C            PHYSIKER, SPRINGER, BERLIN/GOETTINGEN/HEIDELBERG, 1963,  DLBV1470
C            PP.227-232.                                              DLBV1480
C                                                                     DLBV1490
C     ..................................................................DLBV1500
C                                                                     DLBV1510
      CSUBROUTINE DLBVP(PRMT,B,C,R,Y,DERY,NDIM,IHLF,AFCT,FCT,DFCT,OUTP, DLBV1520
     1AUX,A)                                                          DLBV1530
C                                                                     DLBV1540
C                                                                     DLBV1550
      DIMENSION PRMT(1),B(1),C(1),R(1),Y(1),DERY(1),AUX(20,1),A(1)    DLBV1560
      DOUBLE PRECISION PRMT,B,C,R,Y,DERY,AUX,A,H,X,Z,GL,HS,GU,SUM,    DLBV1570
     1DGL,DGU,XST,XEND,DELT                                           DLBV1580
C                                                                     DLBV1590
C     ERROR TEST                                                      DLBV1600
      IF(PRMT(3)*(PRMT(2)-PRMT(1)))2,1,3                              DLBV1610
    1 IHLF=12                                                         DLBV1620
      RETURN                                                          DLBV1630
    2 IHLF=13                                                         DLBV1640
      RETURN                                                          DLBV1650
C                                                                     DLBV1660
C     SEARCH FOR ZERO-COLUMNS IN MATRICES B AND C                     DLBV1670
    3 KK=-NDIM                                                        DLBV1680
      IB=0                                                            DLBV1690
      IC=0                                                            DLBV1700
      DO 7 K=1,NDIM                                                   DLBV1710
      AUX(15,K)=DERY(K)                                               DLBV1720
      AUX(1,K)=1.D0                                                   DLBV1730
      AUX(17,K)=1.D0                                                  DLBV1740
      KK=KK+NDIM                                                      DLBV1750
      DO 4 I=1,NDIM                                                   DLBV1760
      II=KK+I                                                         DLBV1770
      IF(B(II))5,4,5                                                  DLBV1780
    4 CONTINUE                                                        DLBV1790
      IB=IB+1                                                         DLBV1800
      AUX(1,K)=0.D0                                                   DLBV1810
    5 DO 6 I=1,NDIM                                                   DLBV1820
      II=KK+I                                                         DLBV1830
      IF(C(II))7,6,7                                                  DLBV1840
    6 CONTINUE                                                        DLBV1850
      IC=IC+1                                                         DLBV1860
      AUX(17,K)=0.D0                                                  DLBV1870
    7 CONTINUE                                                        DLBV1880
C                                                                     DLBV1890
C     DETERMINATION OF LOWER AND UPPER BOUND                          DLBV1900
      IF(IC-IB)8,11,11                                                DLBV1910
    8 H=PRMT(2)                                                       DLBV1920
      PRMT(2)=PRMT(1)                                                 DLBV1930
      PRMT(1)=H                                                       DLBV1940
      PRMT(3)=-PRMT(3)                                                DLBV1950
      DO 9 I=1,NDIM                                                   DLBV1960
    9 AUX(17,I)=AUX(1,I)                                              DLBV1970
      II=NDIM*NDIM                                                    DLBV1980
      DO 10 I=1,II                                                    DLBV1990
      H=B(I)                                                          DLBV2000
      B(I)=C(I)                                                       DLBV2010
   10 C(I)=H                                                          DLBV2020
C                                                                     DLBV2030
C     PREPARATIONS FOR CONSTRUCTION OF ADJOINT INITIAL VALUE PROBLEMS DLBV2040
   11 X=PRMT(2)                                                       DLBV2050
      CALL FCT(X,Y)                                                   DLBV2060
      CALL DFCT(X,DERY)                                               DLBV2070
      DO 12 I=1,NDIM                                                  DLBV2080
      AUX(18,I)=Y(I)                                                  DLBV2090
   12 AUX(19,I)=DERY(I)                                               DLBV2100
C                                                                     DLBV2110
C     POSSIBLE BREAK-POINT FOR LINKAGE                                DLBV2120
C                                                                     DLBV2130
C     THE FOLLOWING PART OF SUBROUTINE DLBVP UNTIL NEXT BREAK-POINT FOR DLBV2140
C     LINKAGE HAS TO REMAIN IN CORE DURING THE WHOLE REST OF THE      DLBV2150
C     COMPUTATIONS                                                    DLBV2160
C                                                                     DLBV2170
C     START LOOP FOR GENERATING ADJOINT INITIAL VALUE PROBLEMS        DLBV2180
      K=0                                                             DLBV2190
      KK=0                                                            DLBV2200
  100 K=K+1                                                           DLBV2210
      IF(AUX(17,K))108,108,101                                        DLBV2220
C                                                                     DLBV2230
C     INITIALIZATION OF ADJOINT INITIAL VALUE PROBLEM                 DLBV2240
  101 X=PRMT(2)                                                       DLBV2250
      CALL AFCT(X,A)                                                  DLBV2260
      SUM=0.D0                                                        DLBV2270
      GL=AUX(18,K)                                                    DLBV2280
      DGL=AUX(19,K)                                                   DLBV2290
      II=K                                                            DLBV2300
      DO 104 I=1,NDIM                                                 DLBV2310
      H=-A(II)                                                        DLBV2320
      DERY(I)=H                                                       DLBV2330
      AUX(20,I)=R(I)                                                  DLBV2340
      Y(I)=0.D0                                                       DLBV2350
      IF(I-K)103,102,103                                              DLBV2360
  102 Y(I)=1.D0                                                       DLBV2370
  103 DGL=DGL+H*AUX(18,I)                                             DLBV2380
  104 II=II+NDIM                                                      DLBV2390
      XEND=PRMT(1)                                                    DLBV2400
      H=.0625D0*(XEND-X)                                              DLBV2410
      ISW=0                                                           DLBV2420
      GOTO 400                                                        DLBV2430
C     THIS IS BRANCH TO ADJOINT LINEAR INITIAL VALUE PROBLEM          DLBV2440
C                                                                     DLBV2450
C     THIS IS RETURN FROM ADJOINT LINEAR INITIAL VALUE PROBLEM        DLBV2460
  105 IF(IHLF-10)106,106,117                                          DLBV2470
C                                                                     DLBV2480
C     UPDATING OF COEFFICIENT MATRIX B AND VECTOR R                   DLBV2490
  106 DO 107 I=1,NDIM                                                 DLBV2500
      KK=KK+1                                                         DLBV2510
      H=C(KK)                                                         DLBV2520
      R(I)=AUX(20,I)+H*SUM                                            DLBV2530
      II=1                                                            DLBV2540
      DO 107 J=1,NDIM                                                 DLBV2550
      B(II)=B(II)+H*Y(J)                                              DLBV2560
  107 II=II+NDIM                                                      DLBV2570
      GOTO 109                                                        DLBV2580
  108 KK=KK+NDIM                                                      DLBV2590
  109 IF(K-NDIM)100,110,110                                           DLBV2600
C                                                                     DLBV2610
C                                                                     DLBV2620
C     GENERATION OF LAST INITIAL VALUE PROBLEM                        DLBV2630
  110 EPS=PRMT(4)                                                     DLBV2640
      CALL DGELG(R,B,NDIM,1,EPS,I)                                    DLBV2650
      IF(I)111,112,112                                                DLBV2660
  111 IHLF=14                                                         DLBV2670
      RETURN                                                          DLBV2680
C                                                                     DLBV2690
  112 PRMT(5)=0.D0                                                    DLBV2700
      IHLF=-I                                                         DLBV2710
      X=PRMT(1)                                                       DLBV2720
      XEND=PRMT(2)                                                    DLBV2730
      H=PRMT(3)                                                       DLBV2740
      DO 113 I=1,NDIM                                                 DLBV2750
  113 Y(I)=R(I)                                                       DLBV2760
      ISW=1                                                           DLBV2770
  114 ISW2=12                                                         DLBV2780
      GOTO 200                                                        DLBV2790
  115 ISW3=-1                                                         DLBV2800
      GOTO 300                                                        DLBV2810
  116 IF(IHLF)400,400,117                                             DLBV2820
C     THIS WAS BRANCH INTO INITIAL VALUE PROBLEM                      DLBV2830
C                                                                     DLBV2840
C     THIS IS RETURN FROM INITIAL VALUE PROBLEM                       DLBV2850
  117 RETURN                                                          DLBV2860
C                                                                     DLBV2870
C     THIS PART OF LINEAR BOUNDARY VALUE PROBLEM COMPUTES THE RIGHT   DLBV2880
C     HAND SIDE DERY OF THE SYSTEM OF ADJOINT LINEAR DIFFERENTIAL     DLBV2890
C     EQUATIONS (IN CASE ISW=0) OR OF THE GIVEN SYSTEM (IN CASE ISW=1). DLBV2900
  200 CALL AFCT(X,A)                                                  DLBV2910
      IF(ISW)201,201,205                                              DLBV2920
C                                                                     DLBV2930
C     ADJOINT SYSTEM                                                  DLBV2940
  201 LL=0                                                            DLBV2950
      DO 203 M=1,NDIM                                                 DLBV2960
      HS=0.D0                                                         DLBV2970
      DO 202 L=1,NDIM                                                 DLBV2980
      LL=LL+1                                                         DLBV2990
  202 HS=HS-A(LL)*Y(L)                                                DLBV3000
  203 DERY(M)=HS                                                      DLBV3010
  204 GOTO(502,504,506,407,415,418,608,617,632,634,421,115),ISW2      DLBV3020
C                                                                     DLBV3030
C     GIVEN SYSTEM                                                    DLBV3040
  205 CALL FCT(X,DERY)                                                DLBV3050
      DO 207 M=1,NDIM                                                 DLBV3060
      LL=M-NDIM                                                       DLBV3070
      HS=0.D0                                                         DLBV3080
      DO 206 L=1,NDIM                                                 DLBV3090
      LL=LL+NDIM                                                      DLBV3100
  206 HS=HS+A(LL)*Y(L)                                                DLBV3110
  207 DERY(M)=HS+DERY(M)                                              DLBV3120
      GOTO 204                                                        DLBV3130
C                                                                     DLBV3140
C     THIS PART OF LINEAR BOUNDARY VALUE PROBLEM COMPUTES THE VALUE OF DLBV3150
C     INTEGRAL SUM, WHICH IS A PART OF THE OUTPUT OF ADJOINT INITIAL  DLBV3160
C     VALUE PROBLEM (IN CASE ISW=0) OR RECORDS RESULT VALUES OF THE   DLBV3170
C     FINAL INITIAL VALUE PROBLEM (IN CASE ISW=1).                    DLBV3180
  300 IF(ISW)301,301,305                                              DLBV3190
C                                                                     DLBV3200
C     ADJOINT PROBLEM                                                 DLBV3210
  301 CALL FCT(X,R)                                                   DLBV3220
      GU=0.D0                                                         DLBV3230
      DGU=0.D0                                                        DLBV3240
      DO 302 L=1,NDIM                                                 DLBV3250
      GU=GU+Y(L)*R(L)                                                 DLBV3260
  302 DGU=DGU+DERY(L)*R(L)                                            DLBV3270
      CALL DFCT(X,R)                                                  DLBV3280
      DO 303 L=1,NDIM                                                 DLBV3290
  303 DGU=DGU+Y(L)*R(L)                                               DLBV3300
      SUM=SUM+.5D0*H*((GL+GU)+.16666666666666667D0*H*(DGL-DGU))       DLBV3310
      GL=GU                                                           DLBV3320
      DGL=DGU                                                         DLBV3330
  304 IF(ISW3)116,422,618                                             DLBV3340
C                                                                     DLBV3350
C     GIVEN PROBLEM                                                   DLBV3360
  305 CALL OUTP(X,Y,DERY,IHLF,NDIM,PRMT)                              DLBV3370
      IF(PRMT(5))117,304,117                                          DLBV3380
C                                                                     DLBV3390
C     POSSIBLE BREAK-POINT FOR LINKAGE                                DLBV3400
C                                                                     DLBV3410
C     THE FOLLOWING PART OF SUBROUTINE DLBVP SOLVES IN CASE ISW=0 THE DLBV3420
C     ADJOINT INITIAL VALUE PROBLEM. IT COMPUTES INTEGRAL SUM AND     DLBV3430
C     THE VECTOR Y OF DEPENDENT VARIABLES AT THE LOWER BOUND PRMT(1). DLBV3440
C     IN CASE ISW=1 IT SOLVES FINALLY GENERATED INITIAL VALUE PROBLEM. DLBV3450
  400 N=1                                                             DLBV3460
      XST=X                                                           DLBV3470
      IHLF=0                                                          DLBV3480
      DO 401 I=1,NDIM                                                 DLBV3490
      AUX(16,I)=0.D0                                                  DLBV3500
      AUX(1,I)=Y(I)                                                   DLBV3510
```

```
  401 AUX(8,I)=DERY(I)                                                  DLBV3520
      ISW1=1                                                            DLBV3530
      GOTO 500                                                          DLBV3540
C                                                                       DLBV3550
  402 X=X+H                                                             DLBV3560
      DO 403 I=1,NDIM                                                   DLBV3570
  403 AUX(2,I)=Y(I)                                                     DLBV3580
C                                                                       DLBV3590
C     INCREMENT H IS TESTED BY MEANS OF BISECTION                       DLBV3600
  404 IHLF=IHLF+1                                                       DLBV3610
      X=X-H                                                             DLBV3620
      DO 405 I=1,NDIM                                                   DLBV3630
  405 AUX(4,I)=AUX(2,I)                                                 DLBV3640
      H=.5D0*H                                                          DLBV3650
      N=1                                                               DLBV3660
      ISW1=2                                                            DLBV3670
      GOTO 500                                                          DLBV3680
C                                                                       DLBV3690
  406 X=X+H                                                             DLBV3700
      ISW2=4                                                            DLBV3710
      GOTO 200                                                          DLBV3720
  407 N=2                                                               DLBV3730
      DO 408 I=1,NDIM                                                   DLBV3740
      AUX(2,I)=Y(I)                                                     DLBV3750
  408 AUX(9,I)=DERY(I)                                                  DLBV3760
      ISW1=3                                                            DLBV3770
      GOTO 500                                                          DLBV3780
C                                                                       DLBV3790
C     TEST ON SATISFACTORY ACCURACY                                     DLBV3800
  409 DO 414 I=1,NDIM                                                   DLBV3810
      Z=DABS(Y(I))                                                      DLBV3820
      IF(Z-1.D0)410,411,411                                             DLBV3830
  410 Z=1.D0                                                            DLBV3840
  411 DELT=.06666666666666667D0*DABS(Y(I)-AUX(4,I))                     DLBV3850
      IF(ISW)413,413,412                                                DLBV3860
  412 DELT=AUX(15,I)*DELT                                               DLBV3870
  413 IF(DELT-Z*PRMT(4))414,414,429                                     DLBV3880
  414 CONTINUE                                                          DLBV3890
C                                                                       DLBV3900
C     SATISFACTORY ACCURACY AFTER LESS THAN 11 BISECTIONS               DLBV3910
      X=X+H                                                             DLBV3920
      ISW2=5                                                            DLBV3930
      GOTO 200                                                          DLBV3940
  415 DO 416 I=1,NDIM                                                   DLBV3950
      AUX(3,I)=Y(I)                                                     DLBV3960
  416 AUX(10,I)=DERY(I)                                                 DLBV3970
      N=3                                                               DLBV3980
      ISW1=4                                                            DLBV3990
      GOTO 500                                                          DLBV4000
C                                                                       DLBV4010
  417 N=1                                                               DLBV4020
      X=X+H                                                             DLBV4030
      ISW2=6                                                            DLBV4040
      GOTO 200                                                          DLBV4050
  418 X=XST                                                             DLBV4060
      DO 419 I=1,NDIM                                                   DLBV4070
      AUX(11,I)=DERY(I)                                                 DLBV4080
4190Y(I)=AUX(1,I)+H*(.375D0*AUX(8,I)+.7916666666666667D0*AUX(9,I)       DLBV4090
     1-.2083333333333333D0*AUX(10,I)+.04166666666666667D0*DERY(I))      DLBV4100
  420 X=X+H                                                             DLBV4110
      N=N+1                                                             DLBV4120
      ISW2=11                                                           DLBV4130
      GOTO 200                                                          DLBV4140
  421 ISW3=0                                                            DLBV4150
      GOTO 300                                                          DLBV4160
  422 IF(N-4)423,600,600                                                DLBV4170
  423 DO 424 I=1,NDIM                                                   DLBV4180
      AUX(N,I)=Y(I)                                                     DLBV4190
  424 AUX(N+7,I)=DERY(I)                                                DLBV4200
      IF(N-3)425,427,600                                                DLBV4210
C                                                                       DLBV4220
  425 DO 426 I=1,NDIM                                                   DLBV4230
      DELT=AUX(9,I)+AUX(9,I)                                            DLBV4240
      DELT=DELT+DELT                                                    DLBV4250
  426 Y(I)=AUX(1,I)+.3333333333333333D0*H*(AUX(8,I)+DELT+AUX(10,I))     DLBV4260
      GOTO 420                                                          DLBV4270
C                                                                       DLBV4280
  427 DO 428 I=1,NDIM                                                   DLBV4290
      DELT=AUX(9,I)+AUX(10,I)                                           DLBV4300
      DELT=DELT+DELT+DELT                                               DLBV4310
  428 Y(I)=AUX(1,I)+.375D0*H*(AUX(8,I)+DELT+AUX(11,I))                  DLBV4320
      GOTO 420                                                          DLBV4330
C                                                                       DLBV4340
C     NO SATISFACTORY ACCURACY. H MUST BE HALVED.                       DLBV4350
  429 IF(IHLF-10)404,430,430                                            DLBV4360
C                                                                       DLBV4370
C     NO SATISFACTORY ACCURACY AFTER 10 BISECTIONS. ERROR MESSAGE.      DLBV4380
  430 IHLF=11                                                           DLBV4390
      X=X+H                                                             DLBV4400
      IF(ISW)105,105,114                                                DLBV4410
C                                                                       DLBV4420
C     THIS PART OF LINEAR INITIAL VALUE PROBLEM COMPUTES                DLBV4430
C     STARTING VALUES BY MEANS OF RUNGE-KUTTA METHOD.                   DLBV4440
  500 Z=X                                                               DLBV4450
      DO 501 I=1,NDIM                                                   DLBV4460
      X=H*AUX(N+7,I)                                                    DLBV4470
      AUX(5,I)=X                                                        DLBV4480
  501 Y(I)=AUX(N,I)+.4D0*X                                              DLBV4490
C                                                                       DLBV4500
      X=Z+.4D0*H                                                        DLBV4510
      ISW2=1                                                            DLBV4520
      GOTO 200                                                          DLBV4530
  502 DO 503 I=1,NDIM                                                   DLBV4540
      X=H*DERY(I)                                                       DLBV4550
      AUX(6,I)=X                                                        DLBV4560
  503 Y(I)=AUX(N,I)+.2969776092477536D0*AUX(5,I)+.1587596449710358 3D0*XDLBV4570
C                                                                       DLBV4580
      X=Z+.45573725421878943D0*H                                        DLBV4590
      ISW2=2                                                            DLBV4600
      GOTO 200                                                          DLBV4610
  504 DO 505 I=1,NDIM                                                   DLBV4620
      X=H*DERY(I)                                                       DLBV4630
      AUX(7,I)=X                                                        DLBV4640
  505 Y(I)=AUX(N,I)+.2181003882259204 7D0*AUX(5,I)-3.0509651486929308D0* DLBV4650
     1AUX(6,I)+3.8328647604670103D0*X                                   DLBV4660
C                                                                       DLBV4670
      X=Z+H                                                             DLBV4680
      ISW2=3                                                            DLBV4690
      GOTO 200                                                          DLBV4700
  506 DO 507 I=1,NDIM                                                   DLBV4710
5070Y(I)=AUX(N,I)+.1747602822626903 7D0*AUX(5,I)-.5514806628787329400*  DLBV4720
     1AUX(6,I)+1.2055355993965235D0*AUX(7,I)+.1711847812195190 3D0*     DLBV4730
     2H*DERY(I)                                                         DLBV4740
      X=Z                                                               DLBV4750
      GOTO(402,406,409,417),ISW1                                        DLBV4760
C                                                                       DLBV4770
C     POSSIBLE BREAK-POINT FOR LINKAGE                                  DLBV4780
C                                                                       DLBV4790
C     STARTING VALUES ARE COMPUTED.                                     DLBV4800
C     NOW START HAMMINGS MODIFIED PREDICTOR-CORRECTOR METHOD.           DLBV4810
  600 ISTEP=3                                                           DLBV4820
  601 IF(N-8)604,602,604                                                DLBV4830
C                                                                       DLBV4840
C     N=8 CAUSES THE ROWS OF AUX TO CHANGE THEIR STORAGE LOCATIONS      DLBV4850
  602 DO 603 N=2,7                                                      DLBV4860
      DO 603 I=1,NDIM                                                   DLBV4870
      AUX(N-1,I)=AUX(N,I)                                               DLBV4880
  603 AUX(N+6,I)=AUX(N+7,I)                                             DLBV4890
      N=7                                                               DLBV4900
C                                                                       DLBV4910
C     N LESS THAN 8 CAUSES N+1 TO GET N                                 DLBV4920
  604 N=N+1                                                             DLBV4930
C                                                                       DLBV4940
C     COMPUTATION OF NEXT VECTOR Y                                      DLBV4950
      DO 605 I=1,NDIM                                                   DLBV4960
      AUX(N-1,I)=Y(I)                                                   DLBV4970
  605 AUX(N+6,I)=DERY(I)                                                DLBV4980
      X=X+H                                                             DLBV4990
  606 ISTEP=ISTEP+1                                                     DLBV5000
      DO 607 I=1,NDIM                                                   DLBV5010
      ODELT=AUX(N-4,I)+1.3333333333333333D0*H*(AUX(N+6,I)+AUX(N+6,I)-   DLBV5020
     1AUX(N+5,I)+AUX(N+4,I)+AUX(N+4,I))                                 DLBV5030
      Y(I)=DELT-.9256198347107438D0*AUX(16,I)                           DLBV5040
  607 AUX(16,I)=DELT                                                    DLBV5050
C     PREDICTOR IS NOW GENERATED IN ROW 16 OF AUX. MODIFIED PREDICTOR   DLBV5060
C     IS GENERATED IN Y. DELT MEANS AN AUXILIARY STORAGE.               DLBV5070
C                                                                       DLBV5080
      ISW2=7                                                            DLBV5090
      GOTO 200                                                          DLBV5100
C     DERIVATIVE OF MODIFIED PREDICTOR IS GENERATED IN DERY.            DLBV5110
C                                                                       DLBV5120
  608 DO 609 I=1,NDIM                                                   DLBV5130
      ODELT=.125D0*(9.D0*AUX(N-1,I)-AUX(N-3,I)+3.D0*H*(DERY(I)+AUX(N+6,I)DLBV5140
     1+AUX(N+6,I)-AUX(N+5,I)))                                          DLBV5150
      AUX(16,I)=AUX(16,I)-DELT                                          DLBV5160
  609 Y(I)=DELT+.0743801652892562000*AUX(16,I)                          DLBV5170
C                                                                       DLBV5180
C     TEST WHETHER H MUST BE HALVED OR DOUBLED                          DLBV5190
      DELT=0.D0                                                         DLBV5200
      DO 616 I=1,NDIM                                                   DLBV5210
      Z=DABS(Y(I))                                                      DLBV5220
      IF(Z-1.D0)610,611,611                                             DLBV5230
  610 Z=1.D0                                                            DLBV5240
  611 Z=DABS(AUX(16,I))/Z                                               DLBV5250
      IF(ISW)613,613,612                                                DLBV5260
  612 Z=AUX(15,I)*Z                                                     DLBV5270
  613 IF(Z-PRMT(4))614,614,628                                          DLBV5280
  614 IF(DELT-Z)615,616,616                                             DLBV5290
  615 DELT=Z                                                            DLBV5300
  616 CONTINUE                                                          DLBV5310
C                                                                       DLBV5320
C     H MUST NOT BE HALVED. THAT MEANS Y(I) ARE GOOD.                   DLBV5330
      ISW2=8                                                            DLBV5340
      GOTO 200                                                          DLBV5350
  617 ISW3=1                                                            DLBV5360
      GOTO 300                                                          DLBV5370
  618 IF(H*(X-XEND))619,621,621                                         DLBV5380
  619 IF(DABS(X-XEND)-.1D0*DABS(H))621,620,620                          DLBV5390
  620 IF(DELT-.02D0*PRMT(4))622,622,601                                 DLBV5400
  621 IF(ISW)105,105,117                                                DLBV5410
C                                                                       DLBV5420
C     H COULD BE DOUBLED IF ALL NECESSARY PRECEEDING VALUES ARE         DLBV5430
C     AVAILABLE.                                                        DLBV5440
  622 IF(IHLF)601,601,623                                               DLBV5450
  623 IF(N-7)601,624,624                                                DLBV5460
  624 IF(ISTEP-4)601,625,625                                            DLBV5470
  625 IMOD=ISTEP/2                                                      DLBV5480
      IF(ISTEP-IMOD-IMOD)601,626,601                                    DLBV5490
  626 H=H+H                                                             DLBV5500
      IHLF=IHLF-1                                                       DLBV5510
      ISTEP=0                                                           DLBV5520
      DO 627 I=1,NDIM                                                   DLBV5530
      AUX(N-1,I)=AUX(N-2,I)                                             DLBV5540
      AUX(N-2,I)=AUX(N-4,I)                                             DLBV5550
      AUX(N-3,I)=AUX(N-6,I)                                             DLBV5560
      AUX(N+6,I)=AUX(N+5,I)                                             DLBV5570
      AUX(N+5,I)=AUX(N+3,I)                                             DLBV5580
      AUX(N+4,I)=AUX(N+1,I)                                             DLBV5590
      DELT=AUX(N+6,I)+AUX(N+5,I)                                        DLBV5600
      DELT=DELT+DELT+DELT                                               DLBV5610
  627 0AUX(16,I)=8.9629629629629630D0*(Y(I)-AUX(N-3,I))                 DLBV5620
     1-3.3611111111111111D0*H*(DERY(I)+DELT+AUX(N+4,I))                 DLBV5630
      GOTO 601                                                          DLBV5640
C                                                                       DLBV5650
C     H MUST BE HALVED                                                  DLBV5660
  628 IHLF=IHLF+1                                                       DLBV5670
      IF(IHLF-10)630,630,629                                            DLBV5680
  629 IF(ISW)105,105,114                                                DLBV5690
  630 H=.5D0*H                                                          DLBV5700
      ISTEP=0                                                           DLBV5710
      DO 631 I=1,NDIM                                                   DLBV5720
      0Y(I)=.390625D-2*(8.D1*AUX(N-1,I)+135.D0*AUX(N-2,I)+4.D1*AUX(N-3,I)DLBV5730
     1+AUX(N-4,I))-.11718750D0*(AUX(N+6,I)-6.D0*AUX(N+5,I)-AUX(N+4,I))*H DLBV5740
      0AUX(N-4,I)=.3906250D-2*(12.D0*AUX(N-1,I)+135.D0*AUX(N-2,I)+      DLBV5750
     1108.D0*AUX(N-3,I)+AUX(N-4,I))-.0234375D0*(AUX(N+6,I)+             DLBV5760
     218.D0*AUX(N+5,I)-9.D0*AUX(N+4,I))*H                               DLBV5770
      AUX(N-3,I)=AUX(N-2,I)                                             DLBV5780
  631 AUX(N+4,I)=AUX(N+5,I)                                             DLBV5790
      DELT=X-H                                                          DLBV5800
      X=DELT-(H+H)                                                      DLBV5810
      ISW2=9                                                            DLBV5820
      GOTO 200                                                          DLBV5830
  632 DO 633 I=1,NDIM                                                   DLBV5840
      AUX(N-2,I)=Y(I)                                                   DLBV5850
      AUX(N+5,I)=DERY(I)                                                DLBV5860
  633 Y(I)=AUX(N-4,I)                                                   DLBV5870
      X=X-(H+H)                                                         DLBV5880
      ISW2=10                                                           DLBV5890
      GOTO 200                                                          DLBV5900
  634 X=DELT                                                            DLBV5910
      DO 635 I=1,NDIM                                                   DLBV5920
      DELT=AUX(N+5,I)+AUX(N+4,I)                                        DLBV5930
      DELT=DELT+DELT+DELT                                               DLBV5940
      0AUX(16,I)=8.9629629629629630D0*(AUX(N-1,I)-Y(I))                 DLBV5950
     1-3.3611111111111111D0*H*(AUX(N+6,I)+DELT+DERY(I))                 DLBV5960
  635 AUX(N+3,I)=DERY(I)                                                DLBV5970
      GOTO 606                                                          DLBV5980
C     END OF INITIAL VALUE PROBLEM                                      DLBV5990
      END                                                               DLBV6000
```

Subroutine GMMMA

This subroutine computes the value of the gamma function for a given argument x.
The calculation of the gamma function, $\Gamma(x)$ is defined for $x > 0$ by:

$$\Gamma(x) = \int_0^{\infty} t^{x-1} \cdot e^{-t}\, dt \qquad (1)$$

This function satisfies the recurrence relation:

$$\Gamma(x) = (x-1) \cdot \Gamma(x-1) \qquad (2)$$

which defines $\Gamma(x)$ for any x not a negative integer.

Note that when x is a positive integer, $\Gamma(x) = (x-1)!$

To compute $\Gamma(x)$ for $x > 1$, apply the recurrence (2), r times until $1 < x - r = y \leq 2$. Thus, for $x > 1$:

$$\Gamma(x) = (x-1)\ (x-2)\ \ldots\ (x-r)\ \ \Gamma(y) \qquad (3)$$

$\Gamma(y)$ is computed from the following formula:

$$\begin{aligned}\Gamma(y) \approx 1 &- 0.57710166(y-1) + 0.98585399(y-1)^2 \\ &- 0.87642182(y-1)^3 + 0.83282120(y-1)^4 \\ &- 0.56847290(y-1)^5 + 0.25482049(y-1)^6 \\ &- 0.05149930(y-1)^7\end{aligned} \qquad (4)$$

For $x < 1$, the recurrence (2) is taken in the direction of decreasing n, giving:

$$\Gamma(x) = \frac{\Gamma(y)}{x(x+1)\ (x+2)\ \ldots\ (x+r-1)} \qquad (5)$$

where $1 < x + r = y \leq 2$.

As before, $\Gamma(y)$ is computed using equation (4).

```
C
C     ..................................................................
C
C        SUBROUTINE GMMMA
C
C        PURPOSE
C           COMPUTES THE GAMMA FUNCTION FOR A GIVEN ARGUMENT
C
C        USAGE
C           CALL GMMMA(XX,GX,IER)
C
C        DESCRIPTION OF PARAMETERS
C           XX -THE ARGUMENT FOR THE GAMMA FUNCTION
C           GX -THE RESULTANT GAMMA FUNCTION VALUE
C           IER-RESULTANT ERROR CODE WHERE
C               IER=0  NO ERROR
C               IER=1  XX IS WITHIN .000001 OF BEING A NEGATIVE INTEGER
C               IER=2  XX GT 57, OVERFLOW, GX SET TO 1.0E75
C
C        REMARKS
C           NONE
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           THE RECURSION RELATION AND POLYNOMIAL APPROXIMATION
C           BY C.HASTINGS,JR., 'APPROXIMATIONS FOR DIGITAL COMPUTERS',
C           PRINCETON UNIVERSITY PRESS, 1955
C
C     ..................................................................
C
      SUBROUTINE GMMMA(XX,GX,IER)
      IF(XX-57.)6,6,4
    4 IER=2
      GX=1.E75
      RETURN
    6 X=XX
      ERR=1.0E-6
      IER=0
      GX=1.0
      IF(X-2.0)50,50,15
   10 IF(X-2.0)110,110,15
   15 X=X-1.0
      GX=GX*X
      GO TO 10
   50 IF(X-1.0)60,120,110
C
C        SEE IF X IS NEAR NEGATIVE INTEGER OR ZERO
C
   60 IF(X-ERR)62,62,80
   62 Y=FLOAT(INT(X))-X
      IF(ABS(Y)-ERR)130,130,64
   64 IF(1.0-Y-ERR)130,130,70
C
C        X NOT NEAR A NEGATIVE INTEGER OR ZERO
C
   70 IF(X-1.0)80,80,110
   80 GX=GX/X
      X=X+1.0
      GO TO 70
  110 Y=X-1.0
      GY=1.0+Y*(-0.5771017+Y*(+0.9858540+Y*(-0.8764218+Y*(+0.8328212+
     1Y*(-0.5684729+Y*(+0.2548205+Y*(-0.0514993)))))))
      GX=GX*GY
  120 RETURN
  130 IER=1
      RETURN
      END
```

Subroutine DLGAM

This subroutine computes the double-precision natural logarithm of the gamma function of a given double-precision argument, x, where $10^{-9} < x < 10^{70}$. The Euler-McLaurin expansion, to the seventh derivative term, is used. For $x > 0$:

$$\log \Gamma(x) \simeq (x - 1/2) \log x + 1/2 \log 2\pi - x + 1/12\, x - 1/360\, x^3 + 1/1260\, x^5 - 1/1680\, x^7 \qquad (1)$$

This expression is very accurate for $x > 18$. If $x \leq 18$, x is replaced by $z = k + x$, where k is an integer such that $z > 18$. $\text{Log}\,\Gamma(z)$ is then evaluated by (1), and $\log x + \log (x + 1) + \ldots + \log (x + k - 1)$ is subtracted to obtain the desired result.

If x is between 10^{10} and 10^{70}, terms of lowest order in (1) are neglected, and $\log \Gamma(x)$ is computed as:

$$\log \Gamma(x) = x \; (\log (x) - 1) \qquad (2)$$

Subroutine DLGAM is available in a double-precision format only. If the single-precision value of the log gamma function of a given single-precision argument is desired, subroutine GMMMA should be used along with the FORTRAN-supplied natural logarithm function.

For reference see M. Abramowitz and I.A. Stegun, Handbook of Mathematical Functions. U.S. Department of Commerce, National Bureau of Standards Applied Mathematics Series, 1966, equation 6.1.41.

```
C                                                                       DLGA  10
C     ..................................................................DLGA  20
C                                                                       DLGA  30
C        SUBROUTINE DLGAM                                               DLGA  40
C                                                                       DLGA  50
C        PURPOSE                                                        DLGA  60
C           COMPUTES THE DOUBLE PRECISION NATURAL LOGARITHM OF THE      DLGA  70
C           GAMMA FUNCTION OF A GIVEN DOUBLE PRECISION ARGUMENT.        DLGA  80
C                                                                       DLGA  90
C        USAGE                                                          DLGA 100
C           CALL DLGAM(XX,DLNG,IER)                                     DLGA 110
C                                                                       DLGA 120
C        DESCRIPTION OF PARAMETERS                                      DLGA 130
C           XX   - THE DOUBLE PRECISION ARGUMENT FOR THE LOG GAMMA      DLGA 140
C                  FUNCTION.                                            DLGA 150
C           DLNG - THE RESULTANT DOUBLE PRECISION LOG GAMMA FUNCTION    DLGA 160
C                  VALUE.                                               DLGA 170
C           IER  - RESULTANT ERROR CODE WHERE                           DLGA 180
C                  IER= 0----NO ERROR.                                  DLGA 190
C                  IER=-1----XX IS WITHIN 10**(-9) OF BEING ZERO OR XX  DLGA 200
C                              IS NEGATIVE.  DLNG IS SET TO -1.0D75.    DLGA 210
C                  IER=+1----XX IS GREATER THAN 10**70. DLNG IS SET TO  DLGA 220
C                             +1.0D75.                                  DLGA 230
C                                                                       DLGA 240
C        REMARKS                                                        DLGA 250
C           NONE                                                        DLGA 260
C                                                                       DLGA 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DLGA 280
C           NONE                                                        DLGA 290
C                                                                       DLGA 300
C        METHOD                                                         DLGA 310
C           THE EULER-MCLAURIN EXPANSION TO THE SEVENTH DERIVATIVE TERM DLGA 320
C           IS USED, AS GIVEN BY M. ABRAMOWITZ AND I.A. STEGUN,         DLGA 330
C           'HANDBOOK OF MATHEMATICAL FUNCTIONS', U. S. DEPARTMENT OF   DLGA 340
C           COMMERCE, NATIONAL BUREAU OF STANDARDS APPLIED MATHEMATICS  DLGA 350
C           SERIES. 1966, EQUATION 6.1.41.                              DLGA 360
C                                                                       DLGA 370
C     ..................................................................DLGA 380
C                                                                       DLGA 390
      SUBROUTINE DLGAM(XX,DLNG,IER)                                     DLGA 400
      DOUBLE PRECISION XX,ZZ,TERM,RZZ,DLNG                              DLGA 410
      IER=0                                                             DLGA 420
      ZZ=XX                                                             DLGA 430
      IF(XX-1.D10) 2,2,1                                                DLGA 440
    1 IF(XX-1.D70) 8,9,9                                                DLGA 450
C                                                                       DLGA 460
C        SEE IF XX IS NEAR ZERO OR NEGATIVE                             DLGA 470
C                                                                       DLGA 480
    2 IF(XX-1.D-9) 3,3,4                                                DLGA 490
    3 IER=-1                                                            DLGA 500
      DLNG=-1.D75                                                       DLGA 510
      GO TO 10                                                          DLGA 520
C                                                                       DLGA 530
C        XX GREATER THAN ZERO AND LESS THAN OR EQUAL TO 1.D+10          DLGA 540
C                                                                       DLGA 550
    4 TERM=1.D0                                                         DLGA 560
    5 IF(ZZ-18.D0) 6,6,7                                                DLGA 570
    6 TERM=TERM*ZZ                                                      DLGA 580
      ZZ=ZZ+1.D0                                                        DLGA 590
      GO TO 5                                                           DLGA 600
    7 RZZ=1.D0/ZZ**2                                                    DLGA 610
      DLNG =(ZZ-0.5D0)*DLOG(ZZ)-ZZ +0.918938533204672 7 -DLOG(TERM)+    DLGA 620
     1(1.D0/ZZ)*(.8333333333333333D-1 -(RZZ*(.2777777777777777D-2 +(RZZ*DLGA 630
     2(.7936507936507936D-3 -(RZZ*(.5952380952380952D-3))))))))        DLGA 640
      GO TO 10                                                          DLGA 650
C                                                                       DLGA 660
C        XX GREATER THAN 1.D+10 AND LESS THAN 1.D+70                    DLGA 670
C                                                                       DLGA 680
    8 DLNG=ZZ*(DLOG(ZZ)-1.D0)                                           DLGA 690
      GO TO 10                                                          DLGA 700
C                                                                       DLGA 710
C        XX GREATER THAN OR EQUAL TO 1.D+70                             DLGA 720
C                                                                       DLGA 730
    9 IER=+1                                                            DLGA 740
      DLNG=1.D75                                                        DLGA 750
   10 RETURN                                                            DLGA 760
      END                                                               DLGA 770
```

Subroutine BESJ

This subroutine computes the J Bessel function for a given argument and integer order by using the recurrence relationship:

$$F_{n+1}(x) + F_{n-1}(x) = \left(\frac{2n}{x}\right) F_n(x) \qquad (1)$$

The desired Bessel function is:

$$J_n(x) = \frac{F_n(x)}{\alpha} \qquad (2)$$

where:

$$\alpha = F_0(x) + 2 \sum_{m=1}^{M-2} F_{2m}(x) \qquad (3)$$

M is initialized at M_0.

M_0 is the greater of M_A and M_B where;

$$M_A = \begin{cases} [x+6] & \text{if } x < 5 \\ [1.4x+60/x] & \text{if } x \geq 5 \end{cases}$$

$$M_B = [n+x/4+2]$$

$F_{M-2}, F_{M-3}, \ldots, F_2, F_1, F_0$ is evaluated using equation (1) with $F_M = 0$ and $F_{M-1} = 10^{-30}$. Values of α and $J_n(x)$ are then computed using equations (3) and (2) respectively.

The computation is repeated for M+3.

The values of $J_n(x)$ for M and M+3 are compared:

$$\text{If} \quad \left| J_n(x)_M - J_n(x)_{M+3} \right| \leq \delta \left| J_n(x)_{M+3} \right|$$

this value is accepted as $J_n(x)$; if not, the computation is repeated by adding 3 to M and using this as a new value for M. If M reaches M_{MAX} before the desired accuracy is obtained, execution is terminated. M_{MAX} is defined as:

$$M_{MAX} = \begin{cases} \left[20 + 10x - \frac{x^2}{3}\right] & \text{for } x \leq 15 \\ [90 + x/2] & \text{for } x > 15 \end{cases} \qquad (4)$$

```
C                                                                       BESJ  10
C     ...................................................................BESJ  20
C                                                                       BESJ  30
C        SUBROUTINE BESJ                                                BESJ  40
C                                                                       BESJ  50
C        PURPOSE                                                        BESJ  60
C           COMPUTE THE J BESSEL FUNCTION FOR A GIVEN ARGUMENT AND ORDERBESJ  70
C                                                                       BESJ  80
C        USAGE                                                          BESJ  90
C           CALL BESJ(X,N,BJ,D,IER)                                     BESJ 100
C                                                                       BESJ 110
C        DESCRIPTION OF PARAMETERS                                      BESJ 120
C           X  -THE ARGUMENT OF THE J BESSEL FUNCTION DESIRED           BESJ 130
C           N  -THE ORDER OF THE J BESSEL FUNCTION DESIRED              BESJ 140
C           BJ -THE RESULTANT J BESSEL FUNCTION                         BESJ 150
C           D  -REQUIRED ACCURACY                                       BESJ 160
C           IER-RESULTANT ERROR CODE WHERE                              BESJ 170
C              IER=0  NO ERROR                                          BESJ 180
C              IER=1  N IS NEGATIVE                                     BESJ 190
C              IER=2  X IS NEGATIVE OR ZERO                             BESJ 200
C              IER=3  REQUIRED ACCURACY NOT OBTAINED                    BESJ 210
C              IER=4  RANGE OF N COMPARED TO X NOT CORRECT (SEE REMARKS)BESJ 220
C                                                                       BESJ 230
C        REMARKS                                                        BESJ 240
C           N MUST BE GREATER THAN OR EQUAL TO ZERO, BUT IT MUST BE     BESJ 250
C           LESS THAN                                                   BESJ 260
C              20+10*X-X** 2/3    FOR X LESS THAN OR EQUAL TO 15        BESJ 270
C              90+X/2             FOR X GREATER THAN 15                 BESJ 280
C                                                                       BESJ 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  BESJ 300
C           NONE                                                        BESJ 310
C                                                                       BESJ 320
C        METHOD                                                         BESJ 330
C           RECURRENCE RELATION TECHNIQUE DESCRIBED BY H. GOLDSTEIN AND BESJ 340
C           R.M. THALER,'RECURRENCE TECHNIQUES FOR THE CALCULATION OF   BESJ 350
C           BESSEL FUNCTIONS',M.T.A.C.,V.13,PP.102-108 AND I.A. STEGUN  BESJ 360
C           AND M. ABRAMOWITZ,'GENERATION OF BESSEL FUNCTIONS ON HIGH   BESJ 370
C           SPEED COMPUTERS',M.T.A.C.,V.11,1957,PP.255-257              BESJ 380
C                                                                       BESJ 390
C     ...................................................................BESJ 400
C                                                                       BESJ 410
      SUBROUTINE BESJ(X,N,BJ,D,IER)                                     BESJ 420
C                                                                       BESJ 430
      BJ=.0                                                             BESJ 440
      IF(N)10,20,20                                                     BESJ 450
   10 IER=1                                                             BESJ 460
      RETURN                                                            BESJ 470
   20 IF(X)30,30,31                                                     BESJ 480
   30 IER=2                                                             BESJ 490
      RETURN                                                            BESJ 500
   31 IF(X-15.)32,32,34                                                 BESJ 510
   32 NTEST=20.+10.*X-X** 2/3                                           BESJ 520
      GO TO 36                                                          BESJ 530
   34 NTEST=90.+X/2.                                                    BESJ 540
   36 IF(N-NTEST)40,38,38                                               BESJ 550
   38 IER=4                                                             BESJ 560
      RETURN                                                            BESJ 570
   40 IER=0                                                             BESJ 580
      N1=N+1                                                            BESJ 590
      BPREV=.0                                                          BESJ 600
C                                                                       BESJ 610
C     COMPUTE STARTING VALUE OF M                                       BESJ 620
C                                                                       BESJ 630
      IF(X-5.)50,60,60                                                  BESJ 640
   50 MA=X+6.                                                           BESJ 650
      GO TO 70                                                          BESJ 660
   60 MA=1.4*X+60./X                                                    BESJ 670
   70 MB=N+IFIX(X)/4+2                                                  BESJ 680
      MZERO=MAX0(MA,MB)                                                 BESJ 690
C                                                                       BESJ 700
C     SET UPPER LIMIT OF M                                              BESJ 710
C                                                                       BESJ 720
      MMAX=NTEST                                                        BESJ 730
  100 DO 190 M=MZERO,MMAX,3                                             BESJ 740
C                                                                       BESJ 750
C     SET F(M),F(M-1)                                                   BESJ 760
C                                                                       BESJ 770
      FM1=1.0E-28                                                       BESJ 780
      FM=.0                                                             BESJ 790
      ALPHA=.0                                                          BESJ 800
      IF(M-(M/2)*2)120,110,120                                          BESJ 810
  110 JT=-1                                                             BESJ 820
      GO TO 130                                                         BESJ 830
  120 JT=1                                                              BESJ 840
  130 M2=M-2                                                            BESJ 850
      DO 160 K=1,M2                                                     BESJ 860
      MK=M-K                                                            BESJ 870
      BMK=2.*FLOAT(MK)*FM1/X-FM                                         BESJ 880
      FM=FM1                                                            BESJ 890
      FM1=BMK                                                           BESJ 900
      IF(MK-N-1)150,140,150                                             BESJ 910
  140 BJ=BMK                                                            BESJ 920
  150 JT=-JT                                                            BESJ 930
      S=1+JT                                                            BESJ 940
  160 ALPHA=ALPHA+BMK*S                                                 BESJ 950
      BMK=2.*FM1/X-FM                                                   BESJ 960
      IF(N)180,170,180                                                  BESJ 970
  170 BJ=BMK                                                            BESJ 980
  180 ALPHA=ALPHA+BMK                                                   BESJ 990
      BJ=BJ/ALPHA                                                       BESJ1000
      IF(ABS(BJ-BPREV)-ABS(D*BJ))200,200,190                            BESJ1010
  190 BPREV=BJ                                                          BESJ1020
      IER=3                                                             BESJ1030
  200 RETURN                                                            BESJ1040
      END                                                               BESJ1050
```

Subroutine BESY

This subroutine computes the Y Bessel function for a given argument x and order n. The recurrence relation:

$$Y_{n+1}(x) = \left(\frac{2n}{x}\right) \cdot Y_n(x) - Y_{n-1}(x) \tag{1}$$

is used for this evaluation.

For $x > 4$

$$Y_0(x) = \sqrt{\frac{2}{\pi x}} \left(P_0(x) \sin\left(x - \frac{\pi}{4}\right) + Q_0(x) \cos\left(x - \frac{\pi}{4}\right)\right) \tag{2}$$

$$Y_1(x) = \sqrt{\frac{2}{\pi x}} \left(-P_1(x) \cos\left(x - \frac{\pi}{4}\right) + Q_1(x) \sin\left(x - \frac{\pi}{4}\right)\right) \tag{3}$$

$P_0(x)$, $Q_0(x)$, $P_1(x)$, and $Q_1(x)$ are:

$$\frac{1}{\sqrt{2\pi}} P_0\left(\frac{4}{t}\right) = 0.3989422793 - 0.0017530620t^2 + 0.0001734300t^4 - 0.0000487613t^6 + 0.0000173565t^8 - 0.0000037043t^{10} \tag{4}$$

$$\frac{1}{t\sqrt{2\pi}} Q_0\left(\frac{4}{t}\right) = -0.0124669441 + 0.0004564324t^2 - 0.0000869791t^4 + 0.0000342468t^6 - 0.0000142078t^8 + 0.0000032312t^{10} \tag{5}$$

$$\frac{1}{\sqrt{2\pi}} P_1\left(\frac{4}{t}\right) = 0.3989422819 + 0.0029218256t^2 - 0.0002232030t^4 + 0.0000580759t^6 - 0.0000200920t^8 + 0.0000042414t^{10} \tag{6}$$

$$\frac{1}{t\sqrt{2\pi}} Q_1\left(\frac{4}{t}\right) = 0.0374008364 - 0.0006390400t^2 - 0.0001064741t^4 - 0.0000398708t^6 - 0.0000162200t^8 - 0.0000042414t^{10} \tag{7}$$

where $t = \frac{4}{x}$

For $x \leq 4$

$$Y_0(x) = \frac{2}{\pi} \sum_{m=0}^{15} (-1)^m \left(\frac{x}{2}\right)^{2m} \frac{1}{(m!)^2} \left[\log\frac{x}{2} + \gamma - H_m\right] \tag{8}$$

where:

$$H_m = \begin{cases} \sum_{r=1}^{m} \frac{1}{r} & \text{if } m \geq 1 \\ 0 & \text{if } m = 0 \end{cases} \tag{9}$$

and γ = Euler's constant = 0.5772156649

$$Y_1(x) = -\frac{2}{\pi x} + \frac{2}{\pi} \sum_{m=1}^{16} (-1)^{m+1} \left(\frac{x}{2}\right)^{2m-1} \frac{1}{m!\,(m-1)!} \cdot \left[\log\frac{x}{2} + \gamma - H_m + \frac{1}{2m}\right] \tag{10}$$

```
C                                                                       BESY  10
C     ...................................................................BESY  20
C                                                                       BESY  30
C        SUBROUTINE BESY                                                BESY  40
C                                                                       BESY  50
C        PURPOSE                                                        BESY  60
C           COMPUTE THE Y BESSEL FUNCTION FOR A GIVEN ARGUMENT AND ORDERBESY  70
C                                                                       BESY  80
C        USAGE                                                          BESY  90
C           CALL BESY(X,N,BY,IER)                                       BESY 100
C                                                                       BESY 110
C        DESCRIPTION OF PARAMETERS                                      BESY 120
C           X  -THE ARGUMENT OF THE Y BESSEL FUNCTION DESIRED           BESY 130
C           N  -THE ORDER OF THE Y BESSEL FUNCTION DESIRED              BESY 140
C           BY -THE RESULTANT Y BESSEL FUNCTION                         BESY 150
C           IER-RESULTANT ERROR CODE WHERE                              BESY 160
C              IER=0  NO ERROR                                          BESY 170
C              IER=1  N IS NEGATIVE                                     BESY 180
C              IER=2  X IS NEGATIVE OR ZERO                             BESY 190
C              IER=3  BY HAS EXCEEDED MAGNITUDE OF 10**70               BESY 200
C                                                                       BESY 210
C        REMARKS                                                        BESY 220
C           VERY SMALL VALUES OF X MAY CAUSE THE RANGE OF THE LIBRARY   BESY 230
C           FUNCTION ALOG TO BE EXCEEDED                                BESY 240
C           X MUST BE GREATER THAN ZERO                                 BESY 250
C           N MUST BE GREATER THAN OR EQUAL TO ZERO                     BESY 260
C                                                                       BESY 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  BESY 280
C           NONE                                                        BESY 290
C                                                                       BESY 300
C        METHOD                                                         BESY 310
C           RECURRENCE RELATION AND POLYNOMIAL APPROXIMATION TECHNIQUE  BESY 320
C           AS DESCRIBED BY A.J.M.HITCHCOCK,'POLYNOMIAL APPROXIMATIONS  BESY 330
C           TO BESSEL FUNCTIONS OF ORDER ZERO AND ONE AND TO RELATED    BESY 340
C           FUNCTIONS', M.T.A.C., V.11,1957,PP.86-88, AND G.N. WATSON,  BESY 350
C           'A TREATISE ON THE THEORY OF BESSEL FUNCTIONS', CAMBRIDGE   BESY 360
C           UNIVERSITY PRESS, 1958, P. 62                               BESY 370
C                                                                       BESY 380
C     ...................................................................BESY 390
```

```
C
      SUBROUTINE BESY(X,N,BY,IER)
C
C     CHECK FOR ERRORS IN N AND X
C
      IF(N)180,10,10
   10 IER=0
      IF(X)190,190,20
C
C     BRANCH IF X LESS THAN OR EQUAL 4
C
   20 IF(X-4.0)40,40,30
C
C       COMPUTE Y0 AND Y1 FOR X GREATER THAN 4
C
   30 T1=4.0/X
      T2=T1*T1
      P0=((((-.0000037043*T2+.0000173565)*T2-.0000487613)*T2
     1  +.00017343)*T2-.001753062)*T2+.3989423
      Q0=((((.0000032312*T2-.0000142078)*T2+.0000342468)*T2
     1  -.0000869791)*T2+.0004564324)*T2-.01246694
      P1=((((.0000042414*T2-.0000200920)*T2+.0000580759)*T2
     1  -.000223203)*T2+.002921826)*T2+.3989423
      Q1=((((-.0000036594*T2+.00001622)*T2-.0000398708)*T2
     1  +.0001064741)*T2-.0006390400)*T2+.03740084
      A=2.0/SQRT(X)
      B=A*T1
      C=X-.7853982
      Y0=A*P0*SIN(C)+B*Q0*COS(C)
      Y1=-A*P1*COS(C)+B*Q1*SIN(C)
      GO TO 90
C
C       COMPUTE Y0 AND Y1 FOR X LESS THAN OR EQUAL TO 4
C
   40 XX=X/2.
      X2=XX*XX
      T=ALOG(XX)+.5772157
      SUM=0.
      TERM=T
      Y0=T
      DO 70 L=1,15
      IF(L-1)50,60,50
   50 SUM=SUM+1./FLOAT(L-1)
   60 FL=L
      TS=T-SUM
      TERM=(TERM*(-X2)/FL**2)*(1.-1./(FL*TS))
   70 Y0=Y0+TERM
      TERM = XX*(T-.5)
      SUM=0.
      Y1=TERM
      DO 80 L=2,16
      SUM=SUM+1./FLOAT(L-1)
      FL=L
      FL1=FL-1.
      TS=T-SUM
      TERM=(TERM*(-X2)/(FL1*FL))*((TS-.5/FL)/(TS+.5/FL1))
   80 Y1=Y1+TERM
      PI2=.6366198
      Y0=PI2*Y0
      Y1=-PI2/X+PI2*Y1
C
C     CHECK IF ONLY Y0 OR Y1 IS DESIRED
C
   90 IF(N-1)100,100,130
C
C     RETURN EITHER Y0 OR Y1 AS REQUIRED
C
  100 IF(N)110,120,110
  110 BY=Y1
      GO TO 170
  120 BY=Y0
      GO TO 170
C
C     PERFORM RECURRENCE OPERATIONS TO FIND YN(X)
C
  130 YA=Y0
      YB=Y1
      K=1
  140 T=FLOAT(2*K)/X
      YC=T*YB-YA
      IF(ABS(YC)-1.0E70)145,145,141
  141 IER=3
      RETURN
  145 K=K+1
      IF(K-N)150,160,150
  150 YA=YB
      YB=YC
      GO TO 140
  160 BY=YC
  170 RETURN
  180 IER=1
      RETURN
  190 IER=2
      RETURN
      END
```

Subroutine I0

This subroutine computes the modified Bessel function $I_0(x)$ for a given argument (x).

For $-3.75 \leq x \leq 3.75$; $t = x/3.75$

$$I_0(x) = 1 + 3.515623t^2 + 3.089942t^4 + 1.206749t^6 + 0.2659732t^8 + 0.0360768t^{10} + 0.0045813t^{12} + \epsilon$$

with $|\epsilon| < 1.6 \cdot 10^{-7}$

For $3.75 \leq |x| < \infty$; $t = 3.75/|x|$

$$\sqrt{|x|}\, e^{-|x|} I_0(x) = 0.3989423 + 0.01328592t + 0.00225319t^2 - 0.00157565t^3 + 0.00916281t^4 - 0.02057706t^5 + 0.02635537t^6 - 0.01647633t^7 + 0.00392377t^8 + \epsilon$$

with $|\epsilon| < 1.9 \cdot 10^{-7}$

```
C
C     ..................................................................
C
C       SUBROUTINE IC
C
C       PURPOSE
C           COMPUTE THE MODIFIED BESSEL FUNCTION I OF ORDER ZERO
C
C       USAGE
C           CALL IC(X,RIC)
C
C       DESCRIPTION OF PARAMETERS
C           X    -GIVEN ARGUMENT OF THE BESSEL FUNCTION I OF ORDER 0
C           RIC  -RESULTANT VALUE OF THE BESSEL FUNCTION I OF ORDER 0
C
C       REMARKS
C           LARGE VALUES OF THE ARGUMENT MAY CAUSE OVERFLOW IN THE
C           EULITIN EXP-FUNCTION
C
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C          NONE
C
C       METHOD
C           POLYNOMIAL APPROXIMATIONS GIVEN BY E.E. ALLEN ARE USED FOR
C           CALCULATION.
C           FOR REFERENCE SEE
C           M. ABRAMOWITZ AND I.A. STEGUN,'HANDBOOK OF MATHEMATICAL
C           FUNCTIONS', U.S. DEPARTMENT OF COMMERCE, NATIONAL BUREAU OF
C           STANDARDS APPLIED MATHEMATICS SERIES, 1966, P.378.
C
C     ..................................................................
C
      SUBROUTINE IC(X,RIC)
      RIC=ABS(X)
      IF(RIC-3.75)1,1,2
    1 Z=X*X*7.111111E-2
      RIC=(((((( 4.5813E-3*Z+3.60768E-2)*Z+2.659732E-1)*Z+1.206749E0)*Z
     1+3.089942E0)*Z+3.515623E0)*Z+1.
      RETURN
    2 Z=3.75/RIC
      RIC= EXP(RIC)/SQRT(RIC)*(((((((3.92377E-3*Z-1.647633E-2)*Z
     1+2.635537E-2)*Z-2.057706E-2)*Z+5.16281E-3)*Z-1.57565E-3)*Z
     2+2.25319E-3)*Z+1.328592E-2)*Z+3.989423E-1)
      RETURN
      END
```

Subroutine INUE

This subroutine computes the modified Bessel functions $I_1(x)$, $I_2(x)$, . . . , $I_n(x)$ for a given argument x, where n is the given maximum order.

The recurrence relation

$$I_{k+1}(x) + \frac{2k}{x} I_k(x) - I_{k-1}(x) = 0 \qquad (1)$$

is used for this evaluation.

From (1) the values $G_k = I_k/I_{k-1}$ (2)

satisfy
$$G_k = \frac{1}{\frac{2k}{x} + G_{k+1}} \qquad (3)$$

Therefore, G_{n+1} may be calculated from the continued fraction

$$G_{n+1} = \frac{1}{\frac{2(n+1)}{x} +} \; \frac{1}{\frac{2(n+2)}{x} +} \; \frac{1}{\frac{2(n+3)}{x} +} \; \cdots \; . \qquad (4)$$

This continued fraction is evaluated using the following forward A-B process:

Let $A_{-1} = 1,\ A_0 = 0,\ A_k = \frac{2(n+k)}{x} A_{k-1} + A_{k-2}$ (5)

$B_{-1} = 0,\ B_0 = 1,\ B_k = \frac{2(n+k)}{x} B_{k-1} + B_{k-2}$

for k = 1, 2 . . . .

Then $G_{n+1} = \lim_{k \to \infty} \frac{A_k}{B_k}$ (6)

From G_{n+1} the values G_n, G_{n-1}, . . . , G_1 are obtained

applying (3) successively, with k = n, n-1, . . . , 1.

Finally the values I_1, I_2, . . . I_n are computed using $I_k = G_k \cdot I_{k-1}$, where I_0 is the value of (7)

Bessel function I of order zero, for example, as obtained using subroutine I0.

```
C                                                                       INUE  10
C     ..................................................................INUE  20
C                                                                       INUE  30
C        SUBROUTINE INUE                                                INUE  40
C                                                                       INUE  50
C        PURPOSE                                                        INUE  60
C           COMPUTE THE MODIFIED BESSEL FUNCTIONS I FOR ORDERS 1 TO N   INUE  70
C                                                                       INUE  80
C        USAGE                                                          INUE  90
C           CALL INUE(X,N,ZI,RI)                                        INUE 100
C                                                                       INUE 110
C        DESCRIPTION OF PARAMETERS                                      INUE 120
C           X     -GIVEN ARGUMENT OF THE BESSEL FUNCTIONS I             INUE 130
C           N     -GIVEN MAXIMUM ORDER OF BESSEL FUNCTIONS I            INUE 140
C           ZI    -GIVEN VALUE OF BESSEL FUNCTION I OF ORDER ZERO       INUE 150
C                  FOR ARGUMENT X                                       INUE 160
C           RI    -RESULTANT VECTOR OF DIMENSION N, CONTAINING THE      INUE 170
C                  VALUES OF THE FUNCTIONS I FOR ORDERS 1 TO N          INUE 180
C                                                                       INUE 190
C        REMARKS                                                        INUE 200
C           THE VALUE OF ZI MAY BE CALCULATED USING SUBROUTINE I0.      INUE 210
C           USING A DIFFERENT VALUE HAS THE EFFECT THAT ALL VALUES OF   INUE 220
C           BESSEL FUNCTIONS I ARE MULTIPLIED BY THE  FACTOR ZI/I(0,X)  INUE 230
C           WHERE I(0,X) IS THE VALUE OF I FOR ORDER 0 AND ARGUMENT X.  INUE 240
C           THIS MAY BE USED DISADVANTAGEOUSLY IF ONLY THE RATIOS OF I  INUE 250
C           FOR DIFFERENT ORDERS ARE REQUIRED.                          INUE 260
C                                                                       INUE 270
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  INUE 280
C           NONE                                                        INUE 290
C                                                                       INUE 300
C        METHOD                                                         INUE 310
C           THE VALUES ARE OBTAINED USING BACKWARD RECURRENCE RELATION  INUE 320
C           TECHNIQUE. THE RATIO I(N+1,X)/I(N,X) IS OBTAINED FROM A     INUE 330
C           CONTINUED FRACTION.                                         INUE 340
C           FOR REFERENCE SEE                                           INUE 350
C           G. BLANCH,'NUMERICAL EVALUATION OF CONTINUED FRACTIONS',    INUE 360
C           SIAM REVIEW, VOL.6-NO.4,1964,PP.383-421.                    INUE 370
C                                                                       INUE 380
C     ..................................................................INUE 390
C                                                                       INUE 400
      SUBROUTINE INUE(X,N,ZI,RI)                                        INUE 410
      DIMENSION RI(1)                                                   INUE 420
      IF(N)10,10,1                                                      INUE 430
    1 FN=N+N                                                            INUE 440
      Q1=X/FN                                                           INUE 450
      IF(ABS(X)-5.E-416.6.2                                             INUE 460
    2 A0=1.                                                             INUE 470
      A1=0.                                                             INUE 480
      B0=0.                                                             INUE 490
      B1=1.                                                             INUE 500
      FI=FN                                                             INUE 510
    3 FI=FI+2.                                                          INUE 520
      AN=FI/ABS(X)                                                      INUE 530
      A=AN*A1+A0                                                        INUE 540
      B=AN*B1+B0                                                        INUE 550
      A0=A1                                                             INUE 560
      B0=B1                                                             INUE 570
      A1=A                                                              INUE 580
      B1=B                                                              INUE 590
      Q0=Q1                                                             INUE 600
      Q1=A/B                                                            INUE 610
      IF(ABS((Q1-Q0)/Q1)-1.E-6)4,4,3                                    INUE 620
    4 IF(X)5,6,6                                                        INUE 630
    5 Q1=-Q1                                                            INUE 640
    6 K=N                                                               INUE 650
    7 Q1=X/(FN+X*Q1)                                                    INUE 660
      RI(K)=Q1                                                          INUE 670
      FN=FN-2.                                                          INUE 680
      K=K-1                                                             INUE 690
      IF(K)8,8,7                                                        INUE 700
    8 FI=ZI                                                             INUE 710
      DO 9 I=1,N                                                        INUE 720
      FI=FI*RI(I)                                                       INUE 730
    9 RI(I)=FI                                                          INUE 740
   10 RETURN                                                            INUE 750
      END                                                               INUE 760
```

Subroutine BESK

This subroutine computes the K Bessel function for a given argument x and order n. The recurrence relation:

$$K_{n+1}(x) = \frac{2n}{x} K_n(x) + K_{n-1}(x) \tag{1}$$

is used for this evaluation.

The initial values K_0 and K_1 are found as follows:

For x > 1:

$$K_0(x) = e^{-x}\sqrt{\frac{\pi}{2x}}\; G_0(x) \tag{2}$$

$$K_1(x) = e^{-x}\sqrt{\frac{\pi}{2x}}\; G_1(x) \tag{3}$$

where x = 1/t for t < 1:

$$\begin{aligned} G_0\left(\frac{1}{t}\right)\cdot\sqrt{\frac{\pi}{2}} = {} & 1.2533141 - 0.1566642t \\ & +0.08811128t^2 - 0.09139095t^3 \\ & +0.1344596t^4 - 0.2299850t^5 \\ & +0.3792410t^6 - 0.5247277t^7 \\ & +0.5575368t^8 - 0.4262633t^9 \\ & +0.2184518t^{10} - 0.06680977t^{11} \\ & +0.009189383t^{12} \end{aligned} \tag{4}$$

$$\begin{aligned} G_1\left(\frac{1}{t}\right)\cdot\sqrt{\frac{\pi}{2}} = {} & 1.2533141 + 0.4699927t \\ & -0.1468583t^2 + 0.1280427t^3 \\ & -0.1736432t^4 + 0.2847618t^5 \\ & -0.4594342t^6 + 0.6283381t^7 \\ & -0.6632295t^8 + 0.5050239t^9 \\ & -0.2581304t^{10} + 0.07880001t^{11} \\ & -0.0108242t^{12} \end{aligned} \tag{5}$$

For x ≤ 1:

$$\gamma = \text{Euler's constant} = 0.5772156649 \tag{6}$$

$$K_0(x) = -\left(\gamma + \log\frac{x}{2}\right) + \sum_{s=1}^{6}\left(\frac{x}{2}\right)^{2s}\frac{1}{(s!)^2}\left[H_s - \left(\gamma + \log\frac{x}{2}\right)\right] \tag{7}$$

where:

$$H_s = \sum_{r=1}^{s}\frac{1}{r} \tag{8}$$

$$K_1(x) = \frac{1}{x} + \sum_{s=1}^{8}\left(\frac{x}{2}\right)^{2s-1}\frac{1}{(s!)^2}\left[\frac{1}{2} + s\cdot\left(\gamma + \log\frac{x}{2} - H_s\right)\right] \tag{9}$$

```
C                                                                       BESK  10
C     ..................................................................BESK  20
C                                                                       BESK  30
C        SUBROUTINE BESK                                                BESK  40
C                                                                       BESK  50
C           COMPUTE THE K BESSEL FUNCTION FOR A GIVEN ARGUMENT AND ORDERBESK  60
C                                                                       BESK  70
C        USAGE                                                          BESK  80
C           CALL BESK(X,N,BK,IER)                                       BESK  90
C                                                                       BESK 100
C        DESCRIPTION OF PARAMETERS                                      BESK 110
C           X  -THE ARGUMENT OF THE K BESSEL FUNCTION DESIRED           BESK 120
C           N  -THE ORDER OF THE K BESSEL FUNCTION DESIRED              BESK 130
C           BK -THE RESULTANT K BESSEL FUNCTION                         BESK 140
C           IER-RESULTANT ERROR CODE WHERE                              BESK 150
C              IER=0  NO ERROR                                          BESK 160
C              IER=1  N IS NEGATIVE                                     BESK 170
C              IER=2  X IS ZERO OR NEGATIVE                             BESK 180
C              IER=3  X .GT. 170, MACHINE RANGE EXCEEDED                BESK 190
C              IER=4  BK .GT. 10**70                                    BESK 200
C                                                                       BESK 210
C        REMARKS                                                        BESK 220
C           N MUST BE GREATER THAN OR EQUAL TO ZERO                     BESK 230
C                                                                       BESK 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  BESK 250
C           NONE                                                        BESK 260
C                                                                       BESK 270
C        METHOD                                                         BESK 280
C           COMPUTES ZERO ORDER AND FIRST ORDER BESSEL FUNCTIONS USING  BESK 290
C           SERIES APPROXIMATIONS AND THEN COMPUTES N TH ORDER FUNCTION BESK 300
C           USING RECURRENCE RELATION.                                  BESK 310
C           RECURRENCE RELATION AND POLYNOMIAL APPROXIMATION TECHNIQUE  BESK 320
C           AS DESCRIBED BY A.J.M.HITCHCOCK,'POLYNOMIAL APPROXIMATIONS  BESK 330
C           TO BESSEL FUNCTIONS OF ORDER ZERO AND ONE AND TO RELATED    BESK 340
C           FUNCTIONS', M.T.A.C., V.11,1957,PP.86-88, AND G.N. WATSON,  BESK 350
C           'A TREATISE ON THE THEORY OF BESSEL FUNCTIONS', CAMBRIDGE   BESK 360
C           UNIVERSITY PRESS, 1958, P. 62                               BESK 370
C                                                                       BESK 380
C     ..................................................................BESK 390
C                                                                       BESK 400
      SUBROUTINE BESK(X,N,BK,IER)                                       BESK 410
      DIMENSION T(12)                                                   BESK 420
      BK=0                                                              BESK 430
      IF(N)10,11,11                                                     BESK 440
   10 IER=1                                                             BESK 450
      RETURN                                                            BESK 460
   11 IF(X)12,12,20                                                     BESK 470
   12 IER=2                                                             BESK 480
      RETURN                                                            BESK 490
   20 IF(X-170.0)22,22,21                                               BESK 500
   21 IER=3                                                             BESK 510
      RETURN                                                            BESK 520
   22 IER=0                                                             BESK 530
      IF(X-1.)26,26,25                                                  BESK 540
   25 A=EXP(-X)                                                         BESK 550
      E=1./X                                                            BESK 560
      C=SQRT(E)                                                         BESK 570
      T(1)=E                                                            BESK 580
      DO 26 L=2,12                                                      BESK 590
   26 T(L)=T(L-1)*E                                                     BESK 600
      IF(N-1)27,29,27                                                   BESK 610
C                                                                       BESK 620
C     COMPUTE K0 USING POLYNOMIAL APPROXIMATION                         BESK 630
C                                                                       BESK 640
   27 BK=A*(1.2533141-.1566642*T(1)+.08811128*T(2)-.09139095*T(3)       BESK 650
     2+.1344596*T(4)-.2299850*T(5)+.3792410*T(6)-.5247277*T(7)          BESK 660
     3+.5575368*T(8)-.4262633*T(9)+.2184518*T(10)-.06680977*T(11)       BESK 670
     4+.009189383*T(12))*C                                              BESK 680
      IF(N)28,26,29                                                     BESK 690
   28 BK=BK                                                             BESK 700
      RETURN                                                            BESK 710
C                                                                       BESK 720
C     COMPUTE K1 USING POLYNOMIAL APPROXIMATION                         BESK 730
```

```
C
   25 G1=A*(1.2533141+.46555279*T(1)-.146856639*T(2)+.128C427*T(3)
     2-.1726432*T(4)+.2847618*T(5)-.4594362*T(6)+.6283261*T(7)
     3-.6632255*T(8)+.5C5C239*T(9)-.25E13C4*T(1C)+.07880CC1*T(11)
     4-.C1CE241E*T(12))*C
      IF(N-132C,3C,31
   3C EK=G1
      RETURN
C
C     FRCM KC,K1 CCMPUTE KA USING RECURRENCE RELATICN
C
   31 CC 35 J=2,N
      GJ=2.*(FLCAT(J)-1.)*G1/X+GC
      IF(GJ-1.CE7C)33,33,32
   32 IER=4
      GC TC 34
   33 GC=G1
   35 G1=GJ
   34 EK=GJ
      RETURN
   36 B=X/2.
      A=.5772157+ALCG(B)
      C=B*B
      IF(N-1)37,43,37
C
C     CCMPUTE KC USING SERIES EXPANSICN
C
   37 GC=-A
      X2J=1.
      FACT=1.
      HJ=.C
      CC 4C J=1,6
      RJ=1./FLCAT(J)
      X2J=X2J*C
      FACT=FACT*RJ*RJ
      HJ=HJ+RJ
   4C GC=GC+X2J*FACT*(HJ-A)
      IF(N)43,42,43
   42 EK=GC
      RETURN
C
C     CCMPUTE K1 USING SERIES EXPANSICN
C
   43 X2J=B
      FACT=1.
      HJ=1.
      G1=1./X+X2J*(.5+A-HJ)
      CC 5C J=2,8
      X2J=X2J*C
      RJ=1./FLCAT(J)
      FACT=FACT*RJ*RJ
      HJ=HJ+RJ
   5C G1=G1+X2J*FACT*(.5+(A-HJ)*FLCAT(J))
      IF(N-1)31,52,31
   52 EK=G1
      RETURN
      END
```

Subroutine EXPI

This subroutine computes the exponential integral for a given value of the argument x.

The exponential integral is defined as:

$$E_1(x) = -Ei(-x) = \int_x^{\infty} \frac{e^{-t}}{t}\,dt = \int_0^x \frac{1-e^{-t}}{t}\,dt - C - \ln(x)$$

where C = 0.57 ... is Euler's constant.

The integral

$\int_x^{\infty} \frac{e^{-t}}{t}\,dt$ has a logarithmic singularity at x = 0.

Therefore, it must be evaluated in the Cauchy Principal Value sense for negative values of x.

For x = 0 the routine returns -1.E75 as value of $E_1(x)$.

For $x \geq 1$:

$$\begin{aligned} AUX &= xe^x E_1(x) \\ &= (x^4 + 8.5733287401x^3 + 18.0590169730x^2 \\ &\quad + 8.6347608925x + 0.2677737343)/ \\ &\quad (x^4 + 9.5733223454x^3 + 25.6329561486x^2 \\ &\quad + 21.0996530827x + 3.9584969228) + \epsilon \end{aligned}$$

with $|\epsilon| < 2.E-8$.

For $-3 \leq x \leq 1$:

$$\begin{aligned} AUX &= (E_1(x) + \ln|x| + C)/x \\ &= 9.999999E-1 - 2.500001E-1 \cdot x \\ &+ 5.555682E-2 \cdot x^2 - 1.041576E-2 \cdot x^3 \\ &+ 1.664156E-3 \cdot x^4 - 2.335379E-4 \cdot x^5 \\ &+ 2.928433E-5 \cdot x^6 - 1.766345E-6 \cdot x^7 \\ &+ 7.122452E-7 \cdot x^7 + \epsilon \end{aligned}$$

with $|\epsilon| < 7.3E-8$.

For $-9 \leq x \leq -3$:

$$AUX = xe^{x}E_1(x)$$

$$= 1-(248.6697 + 224.4234x + 32.43665x^2 + 3.061037x^3 + 0.05176245x^4)/(180.7837 + 22.63818x + 38.93944x^2 + 3.995161x^3 + x^4) + \epsilon$$

with $|\epsilon| < 1.3E-7$.

For $-174 \leq x \leq -9$:

$$AUX = xe^{x}E_1(x)$$

$$= 1-(674.567029 + 57.4118330x - 6.05529232x^2 - x^3)/(1699.06552 + 841.654932x + 49.3133893x^2 - 8.01957683x^3 - 0.999792040x^4) + \epsilon$$

with $|\epsilon| < 3.8E-7$.

```
C                                                                       EXPI  10
C     ..................................................................EXPI  20
C                                                                       EXPI  30
C        SUBROUTINE EXPI                                                EXPI  40
C                                                                       EXPI  50
C        PURPOSE                                                        EXPI  60
C           COMPUTES THE EXPONENTIAL INTEGRAL -EI(-X)                   EXPI  70
C                                                                       EXPI  80
C        USAGE                                                          EXPI  90
C           CALL EXPI(X,RES)                                            EXPI 100
C                                                                       EXPI 110
C        DESCRIPTION OF PARAMETERS                                      EXPI 120
C           X     - ARGUMENT OF EXPONENTIAL INTEGRAL                    EXPI 130
C           RES   - RESULT VALUE                                        EXPI 140
C           AUX   - RESULTANT AUXILIARY VALUE                           EXPI 150
C                                                                       EXPI 160
C        REMARKS                                                        EXPI 170
C           X GT 170 (X LT -174) MAY CAUSE UNDERFLOW (OVERFLOW)         EXPI 180
C           WITH THE EXPONENTIAL FUNCTION                               EXPI 190
C           FOR X = 0 THE RESULT VALUE IS SET TO -1.E75                 EXPI 200
C                                                                       EXPI 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  EXPI 220
C           NONE                                                        EXPI 230
C                                                                       EXPI 240
C        METHOD                                                         EXPI 250
C           DEFINITION                                                  EXPI 260
C           RES=INTEGRAL(EXP(-T)/T, SUMMED OVER T FROM X TO INFINITY).  EXPI 270
C           EVALUATION                                                  EXPI 280
C           THREE DIFFERENT RATIONAL APPROXIMATIONS ARE USED IN THE     EXPI 290
C           RANGES 1 LE X, X LE -9 AND -9 LT X LE -3 RESPECTIVELY,      EXPI 300
C           A POLYNOMIAL APPROXIMATION IS USED IN -3 LT X LT 1.         EXPI 310
C                                                                       EXPI 320
C     ..................................................................EXPI 330
C                                                                       EXPI 340
      SUBROUTINE EXPI(X,RES,AUX)                                        EXPI 350
      IF(X-1.)2,1,1                                                     EXPI 360
    1 Y=1./X                                                            EXPI 370
      AUX=1.-Y*(((Y+3.277356E0)*Y+2.052156E0)*Y+2.709479E-1)/((((Y*     EXPI 380
     1)0.072553E0+5.716543E0)*Y+6.945235E0)*Y+2.593888E0)*Y+2.709456E-1) EXPI 390
      RES=AUX*Y*EXP(-X)                                                 EXPI 400
      RETURN                                                            EXPI 410
    2 IF(X+3.)6,6,3                                                     EXPI 420
    3 AUX=((((((7.122452E-7*X-1.766345E-6)*X+2.928433E-5)*X-2.335379E-4EXPI 430
     1)*X+1.664156E-3)*X-1.041576E-2)*X+5.555682E-2)*X-2.500001E-1)*X   EXPI 440
     2+5.555555E-1                                                      EXPI 450
      RES=-1.E75                                                        EXPI 460
      IF(X)4,5,4                                                        EXPI 470
    4 RES=X*AUX-ALOG(ABS(X))-5.772157E-1                                EXPI 480
    5 RETURN                                                            EXPI 490
    6 IF(X+5.)8,8,7                                                     EXPI 500
    7 AUX=1.-((((5.176245E-2*X+3.061037E0)*X+3.243665E1)*X+2.244234E2)*XEXPI 510
     1+2.486667E2)/((((X+3.595161E0)*X+3.893944E1)*X+2.263818E1)*X      EXPI 520
                                                                        EXPI 530
     2+1.E07E37E2)                                                      EXPI 540
      GOTO 9                                                            EXPI 550
    8 Y=9./X                                                            EXPI 560
      AUX=1.-Y*(((Y+7.659824E-1)*Y-7.271015E-1)*Y-1.080693E0)/((((Y     EXPI 570
     1+2.518755E0+1.122927E1)*Y+5.921405E0)*Y-8.666702E0)*Y-9.724216E0) EXPI 580
    9 RES=AUX*EXP(-X)/X                                                 EXPI 590
      RETURN                                                            EXPI 600
      END
```

Subroutine SICI

This subroutine computes the sine and cosine integrals for a given value of the argument x.

These integrals are defined as:

$$si(x) = S_i(x) - \pi/2 = \int_0^x \frac{\sin(t)}{t}\,dt - \pi/2$$

$$ci(x) = C + \ln(x) + \int_0^x \frac{\cos(t)-1}{t}\,dt$$

where C is Euler's constant 0.57 . . . .

The integral $\int \frac{\cos(t)}{t}\,dt$ has a logarithmic singularity at x = 0. Therefore, it must be evaluated in the Cauchy Principal Value sense for negative values of x.

For x = 0 the routine returns -1.E75 as value of ci(x).

For $x \leq 4$ and $t = 16-x^2$:

$$(si(x) + \pi/2)/x = 4.395509E-1 + 1.964882E-2\cdot t + 6.939889E-4\cdot t^2 + 1.374168E-5\cdot t^3 + 1.568988E-7\cdot t^4 + 1.753141E-9\cdot t^5 + \epsilon$$

with $|\epsilon| < 8.2E-8$

$$-(ci(x) - C - \ln|x|)/x^2 = 1.315308E-1 + 4.990920E-3\cdot t + 1.185999E-4\cdot t^2 + 1.725752E-6\cdot t^3 + 1.584996E-8\cdot t^4 + 1.386985E-10\cdot t^5 + \epsilon$$

with $|\epsilon| < 5.6E-9$.

For $x > 4$ and $t = 4/x$:

$$si(x) = -\left(\frac{4}{x}\right)\,(\cos(x)\cdot P(x) + \sin(x)\cdot Q(x)) + \epsilon$$

$$ci(x) = \left(\frac{4}{x}\right)\,(\sin(x)\cdot P(x) - \cos(x)\cdot Q(x)) + \epsilon$$

where:

$$|\epsilon| < \frac{1}{x}\cdot 2.3E-8 \text{ and}$$

$$P(x) = 2.500000E-1 - 6.646441E-7\cdot t - 3.122418E-2\cdot t^2 - 3.764000E-4\cdot t^3 + 2.601293E-2\cdot t^4 - 7.945556E-3\cdot t^5 - 4.400416E-2\cdot t^6 + 7.902034E-2\cdot t^7 - 6.537283E-2\cdot t^8 + 2.819179E-2\cdot t^9 - 5.108699E-3\cdot t^{10}$$

$$Q(x) = 2.583989E-10 + 6.250011E-2\cdot t - 1.134958E-5\cdot t^2 - 2.314617E-2\cdot t^3 - 3.332519E-3\cdot t^4 + 4.987716E-2\cdot t^5 - 7.261642E-2\cdot t^6 + 5.515070E-2\cdot t^7 - 2.279143E-2\cdot t^8 + 4.048069E-3\cdot t^9$$

```
C                                                                       SICI  10
C     ...................................................................SICI  20
C                                                                       SICI  30
C        SUBROUTINE SICI                                                SICI  40
C                                                                       SICI  50
C        PURPOSE                                                        SICI  60
C           COMPUTES THE SINE AND COSINE INTEGRAL                       SICI  70
C                                                                       SICI  80
C        USAGE                                                          SICI  90
C           CALL SICI(SI,CI,X)                                          SICI 100
C                                                                       SICI 110
C        DESCRIPTION OF PARAMETERS                                      SICI 120
C           SI    - THE RESULTANT VALUE SI(X)                           SICI 130
C           CI    - THE RESULTANT VALUE CI(X)                           SICI 140
C           X     - THE ARGUMENT OF SI(X) AND CI(X)                     SICI 150
C                                                                       SICI 160
C        REMARKS                                                        SICI 170
C           THE ARGUMENT VALUE REMAINS UNCHANGED                        SICI 180
C                                                                       SICI 190
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SICI 200
C           NONE                                                        SICI 210
C                                                                       SICI 220
C        METHOD                                                         SICI 230
C           DEFINITION                                                  SICI 240
C           SI(X)=INTEGRAL(SIN(T)/T)                                    SICI 250
C           CI(X)=INTEGRAL(COS(T)/T)                                    SICI 260
C           EVALUATION                                                  SICI 270
C           REDUCTION OF RANGE USING SYMMETRY.                          SICI 280
C           DIFFERENT APPROXIMATIONS ARE USED FOR ABS(X) GREATER        SICI 290
C           THAN 4 AND FOR ABS(X) LESS THAN 4.                          SICI 300
C           REFERENCE                                                   SICI 310
C           LUKE AND WIPF, 'POLYNOMIAL APPROXIMATIONS TO INTEGRAL       SICI 320
C           TRANSFORMS', MATHEMATICAL TABLES AND OTHER AIDS TO          SICI 330
C           COMPUTATION, VOL. 15, 1961, ISSUE 74, PP. 174-178.          SICI 340
C                                                                       SICI 350
C     ...................................................................SICI 360
C                                                                       SICI 370
      SUBROUTINE SICI(SI,CI,X)                                          SICI 380
      Z=ABS(X)                                                          SICI 390
      IF(Z-4.)1,1,4                                                     SICI 400
    1 Y=(4.-Z)*(4.+Z)                                                   SICI 410
      SI=-1.5707963E                                                    SICI 420
      IF(Z)3,2,3                                                        SICI 430
    2 CI=-1.E75                                                         SICI 440
      RETURN                                                            SICI 450
    3 SI=X*((((((1.753141E-9*Y+1.568988E-7)*Y+1.374168E-5)*Y+6.939889E-4)SICI 460
     1*Y+1.964882E-2)*Y+4.3955C9E-1+SI/X)                               SICI 470
      CI=((5.772156E-1+ALOG(Z))/Z-Z*((((((1.386985E-10*Y+1.584996E-8)*Y  SICI 480
     1+1.725752E-6)*Y+1.185999E-4)*Y+4.990920E-3)*Y+1.315308E-1))*Z     SICI 490
      RETURN                                                            SICI 500
    4 SI=SIN(Z)                                                         SICI 510
      Y=COS(Z)                                                          SICI 520
      Z=4./Z                                                            SICI 530
      L=(((((((4.048069E-3*Z-2.279143E-2)*Z+5.515070E-2)*Z-7.261642E-2)SICI 540
     1*Z+4.587716E-2)*Z-3.332519E-3)*Z-2.314617E-2)*Z-1.134958E-5)*Z    SICI 550
     2+6.250011E-2)*Z+2.583989E-10                                      SICI 560
      M=(((((((1-5.108699E-3*Z+2.819179E-2)*Z-6.537283E-2)*Z            SICI 570
     1+7.902034E-2)*Z-4.400416E-2)*Z-7.945556E-3)*Z+2.601293E-2)*Z      SICI 580
     2-3.764000E-4)*Z-3.122418E-2)*Z-6.646441E-7)*Z+2.500000E-1         SICI 590
      CI=Z*(SI*M-Y*L)                                                   SICI 600
      SI=-Z*(SI*L+Y*M)                                                  SICI 610
      IF(X)5,6,6                                                        SICI 620
    5 SI=3.141593E-SI                                                   SICI 630
    6 RETURN                                                            SICI 640
      END                                                               SICI 650
```

Subroutine CS

This subroutine computes the Fresnel integrals for a given value of the argument x.

The Fresnel integrals are defined as:

$$C(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \frac{\cos(t)}{\sqrt{t}}\,dt = \int_0^{\sqrt{\frac{2}{\pi}x}} \cos\left(\frac{\pi}{2}u^2\right)du$$

and

$$S(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \frac{\sin(t)}{\sqrt{t}}\,dt = \int_0^{\sqrt{\frac{2}{\pi}x}} \sin\left(\frac{\pi}{2}u^2\right)du$$

The subroutine CS calculates both C(x) and S(x) simultaneously.

If x is negative, $|x|$ is used in CS.

For $0 \leq x \leq 4$ and $t = 16 - x^2$:

$$C(x)/\sqrt{x} = 1.840962E\text{-}1 + 1.102544E\text{-}2\cdot t + 1.020418E\text{-}3\cdot t^2 + 3.273308E\text{-}5\cdot t^3 + 5.451182E\text{-}7\cdot t^4 + 5.244297E\text{-}9\cdot t^5 + 5.100785E\text{-}11\cdot t^6 +$$

with $|\epsilon| < 8.5E\text{-}9$

$$S(x)/\sqrt{x^3} = 8.026490E\text{-}2 + 6.121300E\text{-}3\cdot t + 2.441816E\text{-}4\cdot t^2 + 5.051141E\text{-}6\cdot t^3 + 5.883158E\text{-}8\cdot t^4 + 6.677681E\text{-}10\cdot t^5 + \epsilon$$

with $|\epsilon| < 3.5E\text{-}8$.

For $x > 4$ and $t = 4/x$:

$$C(x) = \frac{1}{2} + \sqrt{\frac{4}{x}}\,(\sin(x)\cdot P(x) + \cos(x)\cdot Q(x)) + \epsilon$$

$$S(x) = \frac{1}{2} + \sqrt{\frac{4}{x}}\,(-\cos(x)\cdot P(x) + \sin(x)\cdot Q(x)) + \epsilon$$

where $|\epsilon| < 1.3E\text{-}8$ and

$$P(x) = 1.994712E\text{-}1 - 1.207998E\text{-}6\cdot t - 9.314911E\text{-}3\cdot t^2 - 4.027145E\text{-}4\cdot t^3 + 7.428246E\text{-}3\cdot t^4 - 7.271690E\text{-}3\cdot t^5 + 3.401409E\text{-}3\cdot t^6 - 6.633926E\text{-}4\cdot t^7$$

and

$$Q(x) = 4.444091E\text{-}9 - 2.493322E\text{-}2\cdot t - 1.606428E\text{-}5\cdot t^2 + 5.972151E\text{-}3\cdot t^3 - 3.095341E\text{-}4\cdot t^4 - 6.792801E\text{-}3\cdot t^5 + 7.970943E\text{-}3\cdot t^6 - 4.169289E\text{-}3\cdot t^7 + 8.768258E\text{-}4\cdot t^8$$

```
C                                                                       CS   10
C   ...................................................................CS   20
C                                                                       CS   30
C     SUBROUTINE CS                                                     CS   40
C                                                                       CS   50
C     PURPOSE                                                           CS   60
C        COMPUTES THE FRESNEL INTEGRALS.                                CS   70
C                                                                       CS   80
C     USAGE                                                             CS   90
C        CALL CS (C,S,X)                                                CS  100
C                                                                       CS  110
C     DESCRIPTION OF PARAMETERS                                         CS  120
C        C     - THE RESULTANT VALUE C(X).                              CS  130
C        S     - THE RESULTANT VALUE S(X).                              CS  140
C        X     - THE ARGUMENT OF FRESNEL INTEGRALS                      CS  150
C                IF X IS NEGATIVE, THE ABSOLUTE VALUE IS USED.          CS  160
C                                                                       CS  170
C     REMARKS                                                           CS  180
C        THE ARGUMENT VALUE X REMAINS UNCHANGED.                        CS  190
C                                                                       CS  200
C     SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                     CS  210
C        NONE                                                           CS  220
C                                                                       CS  230
C     METHOD                                                            CS  240
C                                                                       CS  250
C        DEFINITION                                                     CS  260
C        C(X)=INTEGRAL(COS(T)/SQRT(2*PI*T) SUMMED OVER T FROM 0 TO X)CS  270
C        S(X)=INTEGRAL(SIN(T)/SQRT(2*PI*T) SUMMED OVER T FROM 0 TO X)CS  270
C                                                                       CS  280
C        EVALUATION                                                     CS  290
C        USING DIFFERENT APPROXIMATIONS FOR X LESS THAN 4 AND X         CS  300
C        GREATER THAN 4.                                                CS  310
C        REFERENCE                                                      CS  320
C        'COMPUTATION OF FRESNEL INTEGRALS' BY BOERSMA,                 CS  330
C        MATHEMATICAL TABLES AND OTHER AIDS TO COMPUTATION, VOL. 14,    CS  340
C        1960, NO. 72, P. 380.                                          CS  350
C                                                                       CS  360
C   ...................................................................CS  370
      SUBROUTINE CS(C,S,X)                                              CS  380
      Z=ABS(X)                                                          CS  390
      IF(Z-4.)1,1,2                                                     CS  400
    1 C=SQRT(Z)                                                         CS  410
      S=Z*C                                                             CS  420
      Z=(4.-Z)*(4.+Z)                                                   CS  430
      C=C*((((((5.10C785E-11*Z+5.244297E-9)*Z+5.451182E-7)*Z            CS  440
     1+3.2733CEE-5)*Z+1.02C41EE-3)*Z+1.102544E-2)*Z+1.84C565E-1)        CS  450
      S=S*((((((6.677681E-1C*Z+5.883158E-8)*Z+5.C51141E-6)*Z            CS  460
     1+2.441816E-4)*Z+6.12132CE-3)*Z+8.026450E-2)                       CS  470
      RETURN                                                            CS  480
    2 C=COS(Z)                                                          CS  490
      S=SIN(Z)                                                          CS  500
      Z=4./Z                                                            CS  510
      A=(((((((E.768258E-4*Z-4.169289E-3)*Z+7.970943E-3)*Z-6.792801E-3) CS  520
     1*Z-3.C95341E-4)*Z+5.972151E-3)*Z-1.6CE428E-5)*Z-2.493322E-2)*Z    CS  530
     2-4.444C51E-9                                                      CS  540
      B=(((((E-6.633926E-4*Z+3.4014C9E-3)*Z-7.271690E-3)*Z+7.428246E-3) CS  550
     1*Z-4.C27145E-4)*Z-9.314910E-3)*Z-1.2C7998E-6)*Z+1.994711E-1       CS  560
      Z=SQRT(Z)                                                         CS  570
      C=C.5+Z*(C*A+S*B)                                                 CS  580
      S=C.5+Z*(S*A-C*B)                                                 CS  590
      RETURN                                                            CS  600
      END                                                               CS  610
```

Subroutines CEL1 and DCEL1

These subroutines compute the complete elliptic integral of the first kind. This is defined as:

$$K(k) = \int_0^{\pi/2} \frac{dt}{\sqrt{1-k^2 \sin^2 t}}, \quad 0 \le k < 1$$

An equivalent definition is:

$$K(k) = \int_0^{\infty} \frac{dx}{\sqrt{(1+x^2)(1+k_c^2 x^2)}}$$

where k_c is the complementary modulus:

$$k_c^2 + k^2 = 1, \quad 0 < k_c^2 \le 1$$

The subroutines CEL1, DCEL1 calculate K(k) for given modulus k. The calculation of RES = K(k) is based on the process of the arithmetic-geometric mean.

Starting with the pair of numbers:

$$a_0 = 2, \quad g_0 = 2 \cdot k_c$$

the sequences of numbers (a_n), (g_n) are generated using the definition:

$$a_n = (a_{n-1} + g_{n-1}), \quad g_n = 2\sqrt{a_{n-1} \cdot g_{n-1}}$$

This iterative process is stopped at the Nth step, when $a_N = g_N$.

If D is the number of decimal digits in the mantissa of floating-point numbers, then the equality $a_N = g_N$ must be interpreted as $|a_N - g_N|$ is less than $a_N \cdot 10^{-D}$.

Since the sequences $(2^{-(n+1)} a_n)$, $(2^{-(n+1)} g_n)$ converge quadratically to the same limit (arithmetic-geometric mean) the test for the end of iteration may be replaced by comparing $|a_{N-1} - g_{N-1}|$ against $a_{N-1} \cdot 10^{-D/2}$, thus saving one calculation of the geometric mean.

The value of $K(k) = \frac{2^N \pi}{a_N}$.

```
C                                                                       CEL1  10
C     ..................................................................CEL1  20
C                                                                       CEL1  30
C       SUBROUTINE CEL1                                                 CEL1  40
C                                                                       CEL1  50
C       PURPOSE                                                         CEL1  60
C          CALCULATE COMPLETE ELLIPTIC INTEGRAL OF FIRST KIND           CEL1  70
C                                                                       CEL1  80
C       USAGE                                                           CEL1  90
C          CALL CEL1(RES,AK,IER)                                        CEL1 100
C                                                                       CEL1 110
C       DESCRIPTION OF PARAMETERS                                       CEL1 120
C          RES   - RESULT VALUE                                         CEL1 130
C          AK    - MODULUS (INPUT)                                      CEL1 140
C          IER   - RESULTANT ERROR CODE WHERE                           CEL1 150
C                  IER=0  NO ERROR                                      CEL1 160
C                  IER=1  AK NOT IN RANGE -1 TO +1                      CEL1 170
C                                                                       CEL1 180
C       REMARKS                                                         CEL1 190
C          THE RESULT IS SET TO 1.E75 IF ABS(AK) GE 1                   CEL1 200
C          FOR MODULUS AK AND COMPLEMENTARY MODULUS CK,                 CEL1 210
C          RELATION AK*AK+CK*CK=1.0 IS USED.                            CEL1 220
C          AK MUST BE IN THE RANGE -1 TO +1                             CEL1 230
C                                                                       CEL1 240
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   CEL1 250
C          NONE                                                         CEL1 260
C                                                                       CEL1 270
C       METHOD                                                          CEL1 280
C          DEFINITION                                                   CEL1 290
C          CEL1(AK)=INTEGRAL(1/SQRT((1+T*T)*(1+(CK*T)**2)), SUMMED      CEL1 300
C          OVER T FROM 0 TO INFINITY).                                  CEL1 310
C          EQUIVALENT ARE THE DEFINITIONS                               CEL1 320
C          CEL1(AK)=INTEGRAL(1/(COS(T)SQRT(1+(CK*TAN(T))**2)),SUMMED    CEL1 330
C          OVER T FROM 0 TO PI/2),                                      CEL1 340
C          CEL1(AK)=INTEGRAL(1/SQRT(1-(AK*SIN(T))**2),SUMMED OVER T     CEL1 350
C          FROM 0 TO PI/2), WHERE K=SQRT(1.-CK*CK).                     CEL1 360
C          EVALUATION                                                   CEL1 370
C          LANDENS TRANSFORMATION IS USED FOR CALCULATION.              CEL1 380
C          REFERENCE                                                    CEL1 390
C          R.BULIRSCH, 'NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS     CEL1 400
C          AND ELLIPTIC FUNCTIONS', HANDBOOK SERIES SPECIAL FUNCTIONS,  CEL1 410
C          NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.               CEL1 420
C                                                                       CEL1 430
C     ..................................................................CEL1 440
      SUBROUTINE CEL1(RES,AK,IER)                                       CEL1 450
      IER=0                                                             CEL1 460
      ARI=2.                                                            CEL1 470
      CEC=(0.5-AK)+0.5                                                  CEL1 480
      CEC=CEC+CEC*AK                                                    CEL1 490
      RES=0.5                                                           CEL1 500
      IF(CEC)1,2,4                                                      CEL1 510
    1 IER=1                                                             CEL1 520
    2 RES=1.E75                                                         CEL1 530
      RETURN                                                            CEL1 540
    3 CEC=CEC*AARI                                                      CEL1 550
    4 CEC=SQRT(CEC)                                                     CEL1 560
      CEC=CEC+CEC                                                       CEL1 570
      AARI=ARI                                                          CEL1 580
      ARI=ARI+CEC                                                       CEL1 590
      RES=RES+RES                                                       CEL1 600
      IF(CEC/AARI-0.99999)3,5,5                                         CEL1 610
    5 RES=RES/ARI*6.283185E0                                            CEL1 620
      RETURN                                                            CEL1 630
      END                                                               CEL1 640
                                                                        CEL1 650
```

```
C                                                                       DCEL1  10
C     ..................................................................DCEL1  20
C                                                                       DCEL1  30
C       SUBROUTINE DCEL1                                                DCEL1  40
C                                                                       DCEL1  50
C       PURPOSE                                                         DCEL1  60
C          CALCULATE COMPLETE ELLIPTIC INTEGRAL OF FIRST KIND           DCEL1  70
C                                                                       DCEL1  80
C       USAGE                                                           DCEL1  90
C          CALL DCEL1(RES,AK,IER)                                       DCEL1 100
C                                                                       DCEL1 110
C       DESCRIPTION OF PARAMETERS                                       DCEL1 120
C          RES   - RESULT VALUE IN DOUBLE PRECISION                     DCEL1 130
C          AK    - MODULUS (INPUT) IN DOUBLE PRECISION                  DCEL1 140
C          IER   - RESULTANT ERROR CODE WHERE                           DCEL1 150
C                  IER=0  NO ERROR                                      DCEL1 160
C                  IER=1  AK NOT IN RANGE -1 TO +1                      DCEL1 170
C                                                                       DCEL1 180
C       REMARKS                                                         DCEL1 190
C          THE RESULT IS SET TO 1.E75 IF ABS(AK) GE 1                   DCEL1 200
C          FOR MODULUS AK AND COMPLEMENTARY MODULUS CK,                 DCEL1 210
C          RELATION AK*AK+CK*CK=1.0 IS USED.                            DCEL1 220
C          AK MUST BE IN THE RANGE -1 TO +1                             DCEL1 230
C                                                                       DCEL1 240
C       SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   DCEL1 250
C          NONE                                                         DCEL1 260
C                                                                       DCEL1 270
C       METHOD                                                          DCEL1 280
C          DEFINITION                                                   DCEL1 290
C          CEL1(AK)=INTEGRAL(1/SQRT((1+T*T)*(1+(CK*T)**2)), SUMMED      DCEL1 300
C          OVER T FROM 0 TO INFINITY).                                  DCEL1 310
C          EQUIVALENT ARE THE DEFINITIONS                               DCEL1 320
C          CEL1(AK)=INTEGRAL(1/(COS(T)SQRT(1+(CK*TAN(T))**2)),SUMMED    DCEL1 330
C          OVER T FROM 0 TO PI/2),                                      DCEL1 340
C          CEL1(AK)=INTEGRAL(1/SQRT(1-(AK*SIN(T))**2),SUMMED OVER T     DCEL1 350
C          FROM 0 TO PI/2), WHERE K=SQRT(1.-CK*CK).                     DCEL1 360
C          EVALUATION                                                   DCEL1 370
C          LANDENS TRANSFORMATION IS USED FOR CALCULATION.              DCEL1 380
C          REFERENCE                                                    DCEL1 390
C          R.BULIRSCH, 'NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS     DCEL1 400
C          AND ELLIPTIC FUNCTIONS', HANDBOOK SERIES SPECIAL FUNCTIONS,  DCEL1 410
C          NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.               DCEL1 420
C                                                                       DCEL1 430
C     ..................................................................DCEL1 440
      SUBROUTINE DCEL1(RES,AK,IER)                                      DCEL1 450
      DOUBLE PRECISION RES,AK,GEC,ARI,AARI                              DCEL1 460
      IER=0                                                             DCEL1 470
      ARI=2.D0                                                          DCEL1 480
      GEC=(0.5D0-AK)+0.5D0                                              DCEL1 490
      GEC=GEC+GEC*AK                                                    DCEL1 500
      RES=0.5D0                                                         DCEL1 510
      IF(GEC)1,2,4                                                      DCEL1 520
    1 IER=1                                                             DCEL1 530
    2 RES=1.D75                                                         DCEL1 540
      RETURN                                                            DCEL1 550
    3 GEC=GEC*AARI                                                      DCEL1 560
    4 GEC=DSQRT(GEC)                                                    DCEL1 570
      GEC=GEC+GEC                                                       DCEL1 580
      AARI=ARI                                                          DCEL1 590
      ARI=ARI+GEC                                                       DCEL1 600
      RES=RES+RES                                                       DCEL1 610
      IF(GEC/AARI-0.999999999D0)3,5,5                                   DCEL1 620
    5 RES=RES/ARI*6.2831853071795865D0                                  DCEL1 630
      RETURN                                                            DCEL1 640
      END                                                               DCEL1 650
                                                                        DCEL1 660
```

Form H20-0205-3
Page revised 2/14/69
By TNL N20-1944

intentionally left blank

'orm H20-0205-3
'age revised 2/14/69
3y TNL N20-1944

intentionally left blank

Form H20-0205-3
Page revised 2/14/69
By TNL N20-1944

intentionally left blank

Subroutines CEL2 and DCEL 2

These subroutines compute the generalized complete elliptic integral of the second kind. This is defined as:

$$\text{cel 2 } (k;\ A, B) = \int_0^{\pi/2} \frac{A + (B - A)\sin^2 t}{\sqrt{1 - k^2 \sin^2 t}}\,dt$$

Equivalent is the definition:

$$\text{cel 2 } (k;\ A, B) = \int_0^{\infty} \frac{A + Bx^2}{(1 + x^2)\sqrt{(1 + x^2)(1 + k_c^2 x^2)}}\,dx$$

where k_c is the complementary modulus:

$$k_c^2 + k^2 = 1,\ 0 < k_c^2 \leq 1$$

The subroutines CEL2, DCEL2 calculate cel 2 (k; A, B) for given modulus k and constants A, B. The calculation of RES = cel 2 (k; A, B) is based on the process of the arithmetic-geometric mean.

Starting with the pair of numbers:

$$a_0 = 2,\ g_0 = 2 \cdot k_c$$

The sequences of numbers (a_n), (g_n) are generated using for definition:

$$a_n = (a_{n-1} + g_{n-1}),\ g_n = 2\sqrt{a_{n-1}\, g_{n-1}}$$

This iteration process is stopped at the Nth step when $a_N = g_N$.

Further needed are the sequences (A_i), (B_i), defined by means of:

$$A_0 = A,\ B_0 = B$$

$$A_n = B_{n-1}/a_{n-1} + A_{n-1}$$

$$B_n = 2(B_{n-1} + g_{n-1} \cdot A_{n-1})$$

If D is the number of decimal digits in the mantissa of floating-point numbers, the iteration process is stopped as soon as $(a_{N-1} - g_{N-1})$ is less than $a_{N-1} \cdot 10^{-D/2}$.

Since $(2^{-(n+1)}a_n)$, $(2^{-(n+1)}g_n)$ converge quadratically to the same limit (arithmetic-geometric mean), this implies that $(a_N - g_N)$ is less than $a_N \cdot 10^{-D}$

The value of cel 2 (k, A, B) $= \frac{\pi}{4} \cdot \frac{A_{N+1}}{a_N}$

```
C
C     ..................................................................
C
C        SUBROUTINE CEL2
C
C        PURPOSE
C           COMPUTES THE GENERALIZED COMPLETE ELLIPTIC INTEGRAL OF
C           SECOND KIND.
C
C        USAGE
C           CALL CEL2(RES,AK,A,B,IER)
C
C        DESCRIPTION OF PARAMETERS
C           RES   - RESULT VALUE
C           AK    - MODULUS (INPUT)
C           A     - CONSTANT TERM IN NUMERATOR
C           B     - FACTOR OF QUADRATIC TERM IN NUMERATOR
C           IER   - RESULTANT ERROR CODE WHERE
C                   IER=0  NO ERROR
C                   IER=1  AK NOT IN RANGE -1 TO +1
C
C        REMARKS
C           FOR ABS(AK) GE 1 THE RESULT IS SET TO 1.E75 IF B IS
C           POSITIVE, TO -1.E75 IF B IS NEGATIVE.
C           SPECIAL CASES ARE
C           K(K) OBTAINED WITH A = 1, B = 1
C           E(K) OBTAINED WITH A = 1, B = 1-K*K WHERE KK IS
C           COMPLEMENTARY MODULUS.
C           E(K) OBTAINED WITH A = 1, B = 0
C           D(K) OBTAINED WITH A = 0, B = 1
C           WHERE K, E, B, D DEFINE SPECIAL CASES OF THE GENERALIZED
C           COMPLETE ELLIPTIC INTEGRAL OF SECOND KIND IN THE USUAL
C           NOTATION, AND THE ARGUMENT K OF THESE FUNCTIONS MEANS
C           THE MODULUS.
C
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
C           NONE
C
C        METHOD
C           DEFINITION
C           RES=INTEGRAL((A+B*T*T)/(SQRT((1+T*T)*(1+(CK*T)**2)))*(1+T*T))
C           SUMMED OVER T FROM 0 TO INFINITY).
C           EVALUATION
C           LANDENS TRANSFORMATION IS USED FOR CALCULATION.
C           REFERENCE
C           R.BULIRSCH, 'NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS
C           AND ELLIPTIC FUNCTIONS', HANDBOOK SERIES SPECIAL FUNCTIONS,
C           NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.
C
C     ..................................................................
C
      SUBROUTINE CEL2(RES,AK,A,B,IER)
      IER=0
      AMI=2.
      GEO=(0.5-AK)+0.5
      GEO=GEO+GEO*AK
      RES=A
      A1=A+B
      BO=B+B
      IF(GEO)1,2,6
    1 IER=1
    2 IF(B)3,8,4
    3 RES=-1.E75
      RETURN
    4 RES=1.E75
      RETURN
    5 GEO=GEO*AMI
    6 GEO=SQRT(GEO)
      GEO=GEO+GEO
      AMI=AMI
      AMI=AMI+GEO
      BO=BO+RES*GEO
      RES=A1
      BO=BO+BO
      A1=BO/AMI+A1
      IF(GEO/AMI-0.99999)5,7,7
    7 RES=A1/AMI
      RES=RES+0.57079632E0*RES
    8 RETURN
      END
```

```
C                                                                       DCE2  10
C     ..................................................................DCE2  20
C                                                                       DCE2  30
C        SUBROUTINE CCEL2                                               DCE2  40
C                                                                       DCE2  50
C        PURPOSE                                                        DCE2  60
C           COMPUTES THE GENERALIZED COMPLETE ELLIPTIC INTEGRAL OF      DCE2  70
C           SECOND KIND.                                                DCE2  80
C                                                                       DCE2  90
C        USAGE                                                          DCE2 100
C           CALL CCEL2(RES,AK,A,B,IER)                                  DCE2 110
C                                                                       DCE2 120
C        DESCRIPTION OF PARAMETERS                                      DCE2 130
C           RES   - RESULT VALUE IN DOUBLE PRECISION                    DCE2 140
C           AK    - MODULUS (INPUT) IN DOUBLE PRECISION                 DCE2 150
C           A     - DOUBLE PRECISION CONSTANT TERM IN NUMERATOR         DCE2 160
C           B     - DOUBLE PRECISION FACTOR OF QUADRATIC TERM           DCE2 170
C                   IN NUMERATOR                                        DCE2 180
C           IER   - RESULTANT ERROR CODE WHERE                          DCE2 190
C                   IER=0  NO ERROR                                     DCE2 200
C                   IER=1  AK NOT IN RANGE -1 TO +1                     DCE2 210
C                                                                       DCE2 220
C        REMARKS                                                        DCE2 230
C           FOR ABS(AK) GE 1 THE RESULT IS SET TO 1.E75 IF B IS         DCE2 240
C           POSITIVE, TO -1.D75 IF B IS NEGATIVE.                       DCE2 250
C           SPECIAL CASES ARE                                           DCE2 260
C           K(K) OBTAINED WITH A = 1, B = 1                             DCE2 270
C           E(K) OBTAINED WITH A = 1, B = CK*CK WHERE CK IS             DCE2 280
C           COMPLEMENTARY MODULUS.                                      DCE2 290
C           B(K) OBTAINED WITH A = 1, B = C                             DCE2 300
C           D(K) OBTAINED WITH A = C, B = 1                             DCE2 310
C           WHERE K, E, B, D DEFINE SPECIAL CASES OF THE GENERALIZED    DCE2 320
C           COMPLETE ELLIPTIC INTEGRAL OF SECOND KIND IN THE USUAL      DCE2 330
C           NOTATION, AND THE ARGUMENT K OF THESE FUNCTIONS MEANS       DCE2 340
C           THE MODULUS.                                                DCE2 350
C                                                                       DCE2 360
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DCE2 370
C           NONE                                                        DCE2 380
C                                                                       DCE2 390
C        METHOD                                                         DCE2 400
C           DEFINITION                                                  DCE2 410
C           RES=INTEGRAL((A+B*T*T)/(SQRT((1+T*T)*(1+(CK*T)**2))*(1+T*T))DCE2 420
C           SUMMED OVER T FROM 0 TO INFINITY).                          DCE2 430
C           EVALUATION                                                  DCE2 440
C           LANDENS TRANSFORMATION IS USED FOR CALCULATION.             DCE2 450
C                                                                       DCE2 460
C           REFERENCE                                                   DCE2 470
C           R.BULIRSCH, "NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS    DCE2 480
C           AND ELLIPTIC FUNCTIONS", NUMERISCHE MATHEMATIK VOL. 7, 1965, DCE2 490
C           NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.              DCE2 500
C                                                                       DCE2 510
C     ..................................................................DCE2 520
C                                                                       DCE2 530
      SUBROUTINE CCEL2(RES,AK,A,B,IER)                                  DCE2 540
      DOUBLE PRECISION RES,AK,A,B,GEC,ARI,AARI,BO,A1                    DCE2 550
      IER=0                                                             DCE2 560
      ARI=2.D0                                                          DCE2 570
      GEC=(0.5D0-AK)+0.5D0                                              DCE2 580
      GEC=GEC+GEC*AK                                                    DCE2 590
      RES=A                                                             DCE2 600
      A1=A+B                                                            DCE2 610
      BO=B+B                                                            DCE2 620
      IF(GEC)1,2,6                                                      DCE2 630
    1 IER=1                                                             DCE2 640
    2 IF(B)3,B,4                                                        DCE2 650
    3 RES=-1.D75                                                        DCE2 660
      RETURN                                                            DCE2 670
    4 RES=1.D75                                                         DCE2 680
      RETURN                                                            DCE2 690
    5 GEC=GEC*AARI                                                      DCE2 700
    6 GEC=DSQRT(GEC)                                                    DCE2 710
      GEC=GEC+GEC                                                       DCE2 720
      AARI=ARI                                                          DCE2 730
      ARI=ARI+GEC                                                       DCE2 740
      BO=BO+RES*GEC                                                     DCE2 750
      RES=A1                                                            DCE2 760
      BO=BO+BO                                                          DCE2 770
      A1=BO/ARI+A1                                                      DCE2 780
      IF(GEC/AARI-0.999999999D0)5,7,7                                   DCE2 790
    7 RES=A1/ARI                                                        DCE2 800
      RES=RES+0.57079632679489662D0*RES                                 DCE2 810
    8 RETURN                                                            DCE2 820
      END
```

Subroutines ELI1 and DELI1

These subroutines compute the elliptic integral of the first kind, where:

$$\text{eli 1 } (x,\ ck) = \int_0^x \frac{dt}{\sqrt{(1+t^2)\,(1+ck^2t^2)}}$$

Equivalent definitions are:

$$\text{eli 1 } (x,\ ck) = \int_0^{\arctan x} \frac{dt}{\cos t\sqrt{1+ck^2\tan^2 t}}$$

$$= \int_0^{\arctan x} \frac{dt}{\sqrt{1-k^2\sin^2 t}}$$

where k is the modulus and ck the complementary modulus.

For reason of numerical stability, ck is chosen as input parameter instead of modulus k:

$$0 \leqq ck^2 < \infty \quad \text{and } k^2 + ck^2 = 1.$$

This allows k to be any pure imaginary or real number such that $k^2 \leqq 1$. In standard notation:

$$F\ (\varphi,\ k) = \text{eli 1 } (\tan\varphi,\ ck)$$

Computation is based on the process of the arithmetic-geometric mean together with Landen's transformation:

$$\text{eli 1 } (x,\ ck) = \frac{1}{1+ck}\ \text{eli 1 } (x_1,\ ck_1)$$

where:

$$ck_1 = \frac{2\sqrt{ck}}{1+ck}$$

$$x_1 = \frac{(1+ck)\,x}{1-ck\cdot x^2}$$

$$\phi_1 = \arctan x_1$$

$$\sin\phi_1 = \frac{x_1}{\sqrt{1+x_1^2}}$$

The iteration process begins with $ari_0 = 1$, $geo_0 = |ck|$, and the sequences (ari_n), (geo_n) are generated using for definition:

$$ari_{n+1} = ari_n + geo_n$$

$$geo_{n+1} = 2\sqrt{ari_n \cdot geo_n}$$

The iteration process is stopped at the Nth step when $ari_N = geo_N$.

Further needed is an iteration scheme for transformation of the argument x.

Setting $ang_i = \dfrac{ari_i}{x_i}$

then $ang_0 = 1/|x|$

$$ang_{n+1} = ang_n - \frac{ari_n \cdot geo_n}{ang_n}$$

Doing the iteration up to the Nth step (characterized by $ari_N = geo_N$):

$$\text{eli 1 } (x,\ ck) = \frac{ari_0}{ari_N}\ \text{eli 1 } (x_N,\ ck_N)$$

with $$\sin\phi_N = \frac{ari_N}{\pm\sqrt{ang_N^2 + ari_N^2}}$$

The sign determination of $\sin\phi_N$ must be done so that $\phi_i = \arctan x_i$ is continuously increasing with index i.

With $ari_N = geo_N$, then $ck_N = 1$, $k_N = 0$ and therefore:

$$\text{eli 1 } (x_N,\ ck_N) = \int_0^{\phi_N} dt = \phi_N$$

and the final result is:

$$\text{eli 1 } (x,\ ck) = \frac{\phi_N}{ari_N}$$

Degenerate cases of argument and modulus:

$x = 0$ gives result eli 1 = 0

$ck = 0$ gives result eli 1 = $\operatorname{sgn} x \ln\left(|x| + \sqrt{1+x^2}\right)$

```
C                                                                       ELI1  10
C     ..................................................................ELI1  20
C                                                                       ELI1  30
C        SUBROUTINE ELI1                                                ELI1  40
C                                                                       ELI1  50
C        PURPOSE                                                        ELI1  60
C           COMPUTES THE ELLIPTIC INTEGRAL OF FIRST KIND                ELI1  70
C                                                                       ELI1  80
C        USAGE                                                          ELI1  90
C           CALL ELI1(RES,X,CK)                                         ELI1 100
C                                                                       ELI1 110
C        DESCRIPTION OF PARAMETERS                                      ELI1 120
C           RES   - RESULT VALUE                                        ELI1 130
C           X     - UPPER INTEGRATION BOUND (ARGUMENT OF ELLIPTIC       ELI1 140
C                   INTEGRAL OF FIRST KIND)                             ELI1 150
C           CK    - COMPLEMENTARY MODULUS                               ELI1 160
C                                                                       ELI1 170
C                                                                       ELI1 180
C        REMARKS                                                        ELI1 190
C           MODULUS K = SQRT(1.-CK*CK).                                 ELI1 200
C                                                                       ELI1 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ELI1 220
C           NONE                                                        ELI1 230
C                                                                       ELI1 240
C        METHOD                                                         ELI1 250
C           DEFINITION                                                  ELI1 260
C           RES=INTEGRAL(1/SQRT((1+T*T)*(1+(CK*T)**2)), SUMMED          ELI1 270
C           OVER T FROM 0 TO X).                                        ELI1 280
C           EQUIVALENT ARE THE DEFINITIONS                              ELI1 290
C           RES=INTEGRAL(1/(COS(T)*SQRT(1+(CK*TAN(T))**2)), SUMMED      ELI1 300
C           OVER T FROM 0 TO ATAN(X)),                                  ELI1 310
C           RES=INTEGRAL(1/SQRT(1-(K*SIN(T))**2), SUMMED OVER           ELI1 320
C           T FROM 0 TO ATAN(X)).                                       ELI1 330
C           EVALUATION                                                  ELI1 340
C           LANDENS TRANSFORMATION IS USED FOR CALCULATION.             ELI1 350
C           REFERENCE                                                   ELI1 360
C           R. BULIRSCH, NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS ANDELI1 370
C                  ELLIPTIC FUNCTIONS.                                  ELI1 380
C                  HANDBOOK SERIES OF SPECIAL FUNCTIONS                 ELI1 390
C                  NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.       ELI1 400
C                                                                       ELI1 410
C     ..................................................................ELI1 420
C                                                                       ELI1 430
      SUBROUTINE ELI1(RES,X,CK)                                         ELI1 440
C                                                                       ELI1 450
      IF(X)2,1,2                                                        ELI1 460
    1 RES=0.                                                            ELI1 470
      RETURN                                                            ELI1 480
    2 IF(CK)4,3,4                                                       ELI1 490
    3 RES=ALOG(ABS(X)+SQRT(1.+X*X))                                     ELI1 500
      GOTO 13                                                           ELI1 510
    4 ANGLE=ABS(1./X)                                                   ELI1 520
      GEO=ABS(CK)                                                       ELI1 530
      ARI=1.                                                            ELI1 540
      PIM=0.                                                            ELI1 550
    5 SQGEO=ARI*GEO                                                     ELI1 560
      AARI=ARI                                                          ELI1 570
      ARI=GEO+ARI                                                       ELI1 580
      ANGLE=-SQGEO/ANGLE+ANGLE                                          ELI1 590
      SQGEO=SQRT(SQGEO)                                                 ELI1 600
      IF(ANGLE)7,6,7                                                    ELI1 610
C     REPLACE 0 BY SMALL VALUE                                          ELI1 620
    6 ANGLE=SQGEO*1.E-8                                                 ELI1 630
    7 TEST=AARI*1.E-4                                                   ELI1 640
      IF(ABS(AARI-GEO)-TEST)10,10,8                                     ELI1 650
    8 GEO=SQGEO+SQGEO                                                   ELI1 660
      PIM=PIM+PIM                                                       ELI1 670
      IF(ANGLE)9,5,5                                                    ELI1 680
    9 PIM=PIM+3.1415927                                                 ELI1 690
      GOTO 5                                                            ELI1 700
   10 IF(ANGLE)11,12,12                                                 ELI1 710
   11 PIM=PIM+3.1415927                                                 ELI1 720
   12 RES=(ATAN(ARI/ANGLE)+PIM)/ARI                                     ELI1 730
   13 IF(X)14,15,15                                                     ELI1 740
   14 RES=-RES                                                          ELI1 750
   15 RETURN                                                            ELI1 760
      END
```

```
C                                                                       DEL1  10
C     ..................................................................DEL1  20
C                                                                       DEL1  30
C        SUBROUTINE DELI1                                               DEL1  40
C                                                                       DEL1  50
C        PURPOSE                                                        DEL1  60
C           COMPUTES THE ELLIPTIC INTEGRAL OF FIRST KIND                DEL1  70
C                                                                       DEL1  80
C        USAGE                                                          DEL1  90
C           CALL DELI1(RES,X,CK)                                        DEL1 100
C                                                                       DEL1 110
C        DESCRIPTION OF PARAMETERS                                      DEL1 120
C           RES   - RESULT VALUE IN DOUBLE PRECISION                    DEL1 130
C           X     - UPPER INTEGRATION BOUND (ARGUMENT OF ELLIPTIC       DEL1 140
C                   INTEGRAL OF FIRST KIND) IN DOUBLE PRECISION         DEL1 150
C           CK    - COMPLEMENTARY MODULUS IN DOUBLE PRECISION           DEL1 160
C                                                                       DEL1 170
C                                                                       DEL1 180
C        REMARKS                                                        DEL1 190
C           DOUBLE PRECISION MODULUS K = DSQRT(1.D0-CK*CK).             DEL1 200
C                                                                       DEL1 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DEL1 220
C           NONE                                                        DEL1 230
C                                                                       DEL1 240
C        METHOD                                                         DEL1 250
C           DEFINITION                                                  DEL1 260
C           RES=INTEGRAL(1/SQRT((1+T*T)*(1+(CK*T)**2)), SUMMED          DEL1 270
C           OVER T FROM 0 TO X).                                        DEL1 280
C           EQUIVALENT ARE THE DEFINITIONS                              DEL1 290
C           RES=INTEGRAL(1/(COS(T)*SQRT(1+(CK*TAN(T))**2)), SUMMED      DEL1 300
C           OVER T FROM 0 TO ATAN(X)),                                  DEL1 310
C           RES=INTEGRAL(1/SQRT(1-(K*SIN(T))**2), SUMMED OVER           DEL1 320
C           T FROM 0 TO ATAN(X)).                                       DEL1 330
C           EVALUATION                                                  DEL1 340
C           LANDENS TRANSFORMATION IS USED FOR CALCULATION.             DEL1 350
C           REFERENCE                                                   DEL1 360
C           R. BULIRSCH, NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS ANDDEL1 370
C                  ELLIPTIC FUNCTIONS.                                  DEL1 380
C                  HANDBOOK SERIES OF SPECIAL FUNCTIONS                 DEL1 390
C                  NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.       DEL1 400
C                                                                       DEL1 410
C     ..................................................................DEL1 420
C                                                                       DEL1 430
      SUBROUTINE DELI1(RES,X,CK)                                        DEL1 440
C                                                                       DEL1 450
      DOUBLE PRECISION RES,X,CK,ANGLE,GEO,ARI,PIM,SQGEO,AARI,TEST       DEL1 460
C
      IF(X)2,1,2                                                        DEL1 470
    1 RES=0.D0                                                          DEL1 480
      RETURN                                                            DEL1 490
C                                                                       DEL1 500
    2 IF(CK)4,3,4                                                       DEL1 510
    3 RES=DLOG(DABS(X)+DSQRT(1.D0+X*X))                                 DEL1 520
      GOTO 13                                                           DEL1 530
C                                                                       DEL1 540
    4 ANGLE=DABS(1.D0/X)                                                DEL1 550
      GEO=DABS(CK)                                                      DEL1 560
      ARI=1.D0                                                          DEL1 570
      PIM=0.D0                                                          DEL1 580
    5 SQGEO=ARI*GEO                                                     DEL1 590
      AARI=ARI                                                          DEL1 600
      ARI=GEO+ARI                                                       DEL1 610
      ANGLE=-SQGEO/ANGLE+ANGLE                                          DEL1 620
      SQGEO=DSQRT(SQGEO)                                                DEL1 630
      IF(ANGLE)7,6,7                                                    DEL1 640
C                                                                       DEL1 650
C        REPLACE 0 BY SMALL VALUE                                       DEL1 660
C                                                                       DEL1 670
    6 ANGLE=SQGEO*1.D-17                                                DEL1 680
    7 TEST=AARI*1.D-9                                                   DEL1 690
      IF(DABS(AARI-GEO)-TEST)10,10,8                                    DEL1 700
    8 GEO=SQGEO+SQGEO                                                   DEL1 710
      PIM=PIM+PIM                                                       DEL1 720
      IF(ANGLE)9,5,5                                                    DEL1 730
    9 PIM=PIM+3.1415926535897932                                        DEL1 740
      GOTO 5                                                            DEL1 750
   10 IF(ANGLE)11,12,12                                                 DEL1 760
   11 PIM=PIM+3.1415926535897932                                        DEL1 770
   12 RES=(DATAN(ARI/ANGLE)+PIM)/ARI                                    DEL1 780
   13 IF(X)14,15,15                                                     DEL1 790
   14 RES=-RES                                                          DEL1 800
   15 RETURN                                                            DEL1 810
      END                                                               DEL1 820
```

'ubroutines ELI2 and DELI2

'hese subroutines compute the generalized elliptic ntegral of the second kind, where:

$$\text{li 2 } (x,\ ck,\ a,\ b) = \int_0^x \frac{(a + bt^2)\,dt}{(1 + t^2)\sqrt{(1 + t^2)(1 + ck^2 t^2)}}$$

'quivalent are the definitions:

$$\text{li 2 } (x,\ ck,\ a,\ b) = \int_0^{\arctan x} \frac{(a + b\tan^2 t)\,dt}{\sqrt{(1+\tan^2 t)(1+ck^2\tan^2 t)}}$$

$$= \int_0^{\arctan x} \frac{(a + (b-a)\sin^2 t)\,dt}{\sqrt{1 - k^2\sin^2 t}}$$

'here k is the modulus and ck the complementary nodulus.

For reasons of numerical stability, ck is chosen s input parameter instead of modulus k:

$$0 \leqq ck^2 < \infty \text{ and } k^2 + ck^2 = 1$$

'his allows k to be any pure imaginary or real num- er such that $k^2 \leqq 1$.

In standard notation:

$$E\ (\varphi, k) = \text{eli 2 } (\tan\varphi,\ ck,\ 1,\ ck^2)$$
$$F\ (\varphi,\ k) = \text{eli 2 } (\tan\varphi,\ ck,\ 1,\ 1)$$

'omputation is based on the process of the rithmetic-geometric mean together with Landen's ransformation:

$$\text{li 2 } (x, ck, a, b) = \frac{1}{1 + ck}\left(\text{eli 2 } (x_1,\ ck_1,\ a_1,\ b_1) + \frac{1}{2}(a - b)\frac{x_1}{\sqrt{1 + x_1^2}}\right)$$

here:

$$a_1 = \frac{1}{2}(a + b)$$

$$b_1 = \frac{1}{1 + ck}(b + a \cdot ck)$$

$$ck_1 = \frac{2\sqrt{ck}}{1 + ck}$$

$$a_1 - b_1 = \frac{1}{2}(a - b)\frac{1 - ck}{1 + ck}$$

$$x_1 = \frac{(1 + ck)x}{1 - ck \cdot x^2}$$

$$k_1 = \frac{1 - ck}{1 + ck}$$

$$\phi_1 = \arctan x_1$$

$$\sin\phi_1 = \frac{x_1}{\sqrt{1 + x_1^2}}$$

The iteration process begins with $ari_0 = 1$, $geo_0 = |ck|$, and the sequences (ari_n), (geo_n) are generated, using for definition:

$$ari_{n+1} = ari_n + geo_n$$

$$geo_{n+1} = 2\sqrt{ari_n \cdot geo_n}$$

The iteration process is stopped at the Nth step when $ari_N = geo_N$.

Further needed are the sequences (A_i), (B_i), defined by means of:

$A_i = a_i$, $B_i = b_i \cdot ari_i$; that is, by the iteration scheme

$$A_0 = a,\ B_0 = b$$

$$A_{n+1} = \frac{1}{2}\left(A_n + \frac{B_n}{ari_n}\right)$$

$$B_{n+1} = B_n + geo_n \cdot A_n$$

Finally an iteration scheme is needed for transformation of the argument x.

Setting $ang_i = \dfrac{ari_i}{x_i}$

gives $ang_0 = 1/|x|$

$$ang_{n+1} = ang_n - \frac{ari_n \cdot geo_n}{ang_n}$$

Doing the iteration scheme up to the Nth step (characterized by $ari_N = geo_N$):

$$\text{eli } 2\ (x, ck, a, b) = \frac{ari_0}{ari_N}\ \text{eli } 2\ (x_N, ck_N, a_N, b_N) + \text{SUM}$$

$$\text{where SUM} = \frac{a-b}{2}\left(\frac{1}{ari_1}\sin\phi_1 + \frac{1}{ari_2}\,\frac{ari_0 - geo_0}{ari_1}\,\frac{\sin\phi_2}{2}\right.$$

$$+ \ldots + \frac{1}{ari_N}\cdot\frac{ari_0 - geo_0}{ari_1}\cdots\frac{ari_{N-2} - geo_{N-2}}{ari_{N-1}}$$

$$\left.\frac{\sin\phi_N}{2^{N-1}}\right)$$

$$\text{with } \sin\phi_n = \frac{ari_n}{\pm\sqrt{ang_n^2 + ari_n^2}}$$

The sign determination of $\sin\phi_n$ must be done so that $\phi_i = \arctan x_i$ is continuously increasing with index i.

With $ari_N = geo_N$, then $ck_N = 1$, $k_N = 0$, and therefore

$$\text{eli } 2\ (x_N, ck_N, a_N, b_N) = \int_0^{\phi_n}\left[a_N + (b_N - a_N)\sin^2 t\right]dt$$

$$= \frac{a_N + b_N}{2}\,\phi_N - \frac{1}{2}(b_N - a_N)\sin\phi_N\cos\phi_N$$

The final result is:

$$\text{eli } 2\ (x, ck, a, b) = \frac{A_{N+1}}{ari_N}\cdot\arctan\frac{ari_N}{ang_N} + \text{SUM} +$$

$$\frac{1}{ari_N}\left(\frac{a_N - b_N}{2}\right)\sin\phi_N\cos\phi_N$$

Degenerate cases of argument and modulus:

$x = 0$ gives result eli 2 = 0

$ck = 0$ gives result $\text{eli } 2 = \left[b\cdot\ln\left(|x| + \sqrt{1+x^2}\right) + (a-b)\frac{|x|}{\sqrt{1+x^2}}\right]\text{sgn } x$

```
C                                                                       ELI2  10
C     ...................................................................ELI2  20
C                                                                       ELI2  30
C        SUBROUTINE ELI2                                                ELI2  40
C                                                                       ELI2  50
C        PURPOSE                                                        ELI2  60
C           COMPUTES THE GENERALIZED ELLIPTIC INTEGRAL OF SECOND KIND   ELI2  70
C                                                                       ELI2  80
C        USAGE                                                          ELI2  90
C           CALL ELI2(R,X,CK,A,B)                                       ELI2 100
C                                                                       ELI2 110
C        DESCRIPTION OF PARAMETERS                                      ELI2 120
C           R      - RESULT VALUE                                       ELI2 130
C           X      - UPPER INTEGRATION BOUND (ARGUMENT OF ELLIPTIC      ELI2 140
C                    INTEGRAL OF SECOND KIND)                           ELI2 150
C           CK     - COMPLEMENTARY MODULUS                              ELI2 160
C           A      - CONSTANT TERM IN NUMERATOR                         ELI2 170
C           B      - QUADRATIC TERM IN NUMERATOR                        ELI2 180
C                                                                       ELI2 190
C        REMARKS                                                        ELI2 200
C           MODULUS K = SQRT(1.-CK*CK).                                 ELI2 210
C           SPECIAL CASES OF THE GENERALIZED ELLIPTIC INTEGRAL OF       ELI2 220
C           SECOND KIND ARE                                             ELI2 230
C           F(ATAN(X),K) OBTAINED WITH A=1., B=1.                       ELI2 240
C           E(ATAN(X),K) OBTAINED WITH A=1., B=CK*CK.                   ELI2 250
C           B(ATAN(X),K) OBTAINED WITH A=1., B=0.                       ELI2 260
C           D(ATAN(X),K) OBTAINED WITH A=0., B=1.                       ELI2 270
C                                                                       ELI2 280
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ELI2 290
C           NONE                                                        ELI2 300
C                                                                       ELI2 310
C        METHOD                                                         ELI2 320
C           DEFINITION                                                  ELI2 330
C           R=INTEGRAL((A+B*T*T)/(SQRT((1+T*T)*(1+(CK*T)**2))*(1+T*T)), ELI2 340
C                 SUMMED OVER T FROM 0 TO X).                           ELI2 350
C           EQUIVALENT IS THE DEFINITION                                ELI2 360
C           R=INTEGRAL((A+(B-A)*(SIN(T))**2)/SQRT(1-(K*SIN(T))**2),     ELI2 370
C                 SUMMED OVER T FROM 0 TO ATAN(X)).                     ELI2 380
C           EVALUATION                                                  ELI2 390
C           LANDENS TRANSFORMATION IS USED FOR CALCULATION.             ELI2 400
C           REFERENCE                                                   ELI2 410
C           R. BULIRSCH, NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS ANDELI2 420
C                 ELLIPTIC FUNCTIONS                                    ELI2 430
C                 HANDBOOK SERIES OF SPECIAL FUNCTIONS                  ELI2 440
C                 NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.        ELI2 450
C                                                                       ELI2 460
C     ...................................................................ELI2 470
C                                                                       ELI2 480
      SUBROUTINE ELI2(R,X,CK,A,B)                                       ELI2 490
C        TEST ARGUMENT                                                  ELI2 500
      IF(X)2,1,2                                                        ELI2 510
    1 R=0.                                                              ELI2 520
      RETURN                                                            ELI2 530
C        TEST MODULUS                                                   ELI2 540
    2 C=0.                                                              ELI2 550
      D=0.5                                                             ELI2 560
      IF(CK)7,3,7                                                       ELI2 570
    3 R=SQRT(1.+X*X)                                                    ELI2 580
      R=(A-B)*ABS(X)/R+B*ALOG(ABS(X)+R)                                 ELI2 590
C        TEST SIGN OF ARGUMENT                                          ELI2 600
    4 R=R+C*(A-B)                                                       ELI2 610
      IF(X)5,6,6                                                        ELI2 620
    5 R=-R                                                              ELI2 630
    6 RETURN                                                            ELI2 640
C        INITIALIZATION                                                 ELI2 650
    7 AN=(B+A)*0.5                                                      ELI2 660
      AA=A                                                              ELI2 670
      R=B                                                               ELI2 680
      ANG=ABS(1./X)                                                     ELI2 690
      PIM=0.                                                            ELI2 700
      ISI=0                                                             ELI2 710
      ARI=1.                                                            ELI2 720
      GEO=ABS(CK)                                                       ELI2 730
C        LANDEN TRANSFORMATION                                          ELI2 740
    8 R=AA*GEO+R                                                        ELI2 750
      SGEO=ARI*GEO                                                      ELI2 760
      AA=AN                                                             ELI2 770
      AARI=ARI                                                          ELI2 780
C        ARITHMETIC MEAN                                                ELI2 790
      ARI=GEO+ARI                                                       ELI2 800
C        SUM OF SINE VALUES                                             ELI2 810
      AN=(R/ARI+AA)*0.5                                                 ELI2 820
      AANG=ABS(ANG)                                                     ELI2 830
      ANG=-SGEO/ANG+ANG                                                 ELI2 840
      PIMA=PIM                                                          ELI2 850
      IF(ANG)10,9,11                                                    ELI2 860
    9 ANG=-1.E-8*AANG                                                   ELI2 870
   10 PIM=PIM+3.1415927                                                 ELI2 880
      ISI=ISI+1                                                         ELI2 890
   11 AANG=ARI*ARI+ANG*ANG                                              ELI2 900
      P=D/SQRT(AANG)                                                    ELI2 910
      IF(ISI-4)13,12,12                                                 ELI2 920
   12 ISI=ISI-4                                                         ELI2 930
   13 IF(ISI-2)15,14,14                                                 ELI2 940
   14 P=-P                                                              ELI2 950
   15 C=C+P                                                             ELI2 960
      D=D*(AARI-GEO)*0.5/ARI                                            ELI2 970
      IF(ABS(AARI-GEO)-1.E-4*AARI)17,17,16                              ELI2 980
   16 SGEO=SQRT(SGEO)                                                   ELI2 990
C        GEOMETRIC MEAN                                                 ELI21000
      GEO=SGEO+SGEO                                                     ELI21010
      PIM=PIM+PIMA                                                      ELI21020
      ISI=ISI+ISI                                                       ELI21030
      GOTO 8                                                            ELI21040
C        ACCURACY WAS SUFFICIENT                                        ELI21050
   17 R=(ATAN(ARI/ANG)+PIM)*AN/ARI                                      ELI21060
      C=C+D*ANG/AANG                                                    ELI21070
      GOTO 4                                                            ELI21080
      END                                                               ELI21090
```

```
*********************************************************************
      SUBROUTINE DELI2

      PURPOSE
         COMPUTES THE GENERALIZED ELLIPTIC INTEGRAL OF SECOND KIND

      USAGE
         CALL DELI2(R,X,CK,A,B)

      DESCRIPTION OF PARAMETERS
         R      - RESULT VALUE IN DOUBLE PRECISION
         X      - UPPER INTEGRATION BOUND (ARGUMENT OF ELLIPTIC
                  INTEGRAL OF SECOND KIND) IN DOUBLE PRECISION
         CK     - COMPLEMENTARY MODULUS IN DOUBLE PRECISION
         A      - DOUBLE PRECISION CONSTANT TERM IN NUMERATOR
         B      - DOUBLE PRECISION QUATRATIC TERM IN NUMERATOR

      REMARKS
         DOUBLE PRECISION MODULUS K = DSQRT(1.D0-CK*CK).
         SPECIAL CASES OF THE GENERALIZED ELLIPTIC INTEGRAL OF
         SECOND KIND ARE
         F(DATAN(X),K) OBTAINED WITH A=1.D0, B=1.D0
         E(DATAN(X),K) OBTAINED WITH A=1.D0, B=CK*CK
         B(DATAN(X),K) OBTAINED WITH A=1.D0, B=0.D0
         D(DATAN(X),K) OBTAINED WITH A=0.D0, B=1.D0.

      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED
         NONE

      METHOD
         DEFINITION
         R=INTEGRAL((A+B*T*T)/(SQRT((1+T*T)*(1+(CK*T)**2))*(1+T*T)),
                 SUMMED OVER T FROM 0 TO X).
         EQUIVALENT IS THE DEFINITION
         R=INTEGRAL((A+(B-A)*(SIN(T))**2)/SQRT(1-(K*SIN(T))**2),
                 SUMMED OVER T FROM 0 TO ATAN(X)).
         EVALUATION
         LANDENS TRANSFORMATION IS USED FOR CALCULATION.
         REFERENCE
         R. BULIRSCH, NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS AND
                 ELLIPTIC FUNCTIONS
                 HANDBOOK SERIES OF SPECIAL FUNCTIONS
                 NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.
*********************************************************************
      SUBROUTINE DELI2(R,X,CK,A,B)

      DOUBLE PRECISION R,X,A,B,AN,AA,ANG,AANG,PIM,PIMA,ARI,AARI
      DOUBLE PRECISION GEO,SGEO,C,D,P,CK

         TEST ARGUMENT

      IF(X)2,1,2
    1 R=0.D0
      RETURN

         TEST MODULUS

    2 C=0.D0
      D=0.5D0
      IF(CK)7,3,7
    3 R=DSQRT(1.D0+X*X)
      R=(A-B)*DABS(X)/R+B*DLOG(DABS(X)+R)
    4 R=R+C*(A-B)

         TEST SIGN OF ARGUMENT

      IF(X)5,6,6
    5 R=-R
    6 RETURN

         INITIALIZATION

    7 AN=(B+A)*0.5D0
      AA=A
      R=B
      ANG=DABS(1.D0/X)
      PIM=0.D0
      ISI=0
      ARI=1.D0
      GEO=DABS(CK)

         LANDEN TRANSFORMATION

    8 R=AA*GEO+R
      SGEO=ARI*GEO
      AA=AN
      AARI=ARI

         ARITHMETIC MEAN

      ARI=GEO+ARI

         SUM OF SINE VALUES

      AN=(R/ARI+AA)*0.5D0
      AANG=DABS(ANG)
      ANG=-SGEO/ANG+ANG
      PIMA=PIM
      IF(ANG)10,9,11

         REPLACE 0 BY SMALL VALUE

    9 ANG=-1.D-17*AANG
   10 PIM=PIM+3.1415926535897932
      ISI=ISI+1
   11 AANG=ARI*ARI+ANG*ANG
      P=D/DSQRT(AANG)
      IF(ISI-4)13,12,12
   12 ISI=ISI-4
   13 IF(ISI-2)15,14,14
   14 P=-P
   15 C=C+P
      D=D*(AARI-GEO)*0.5D0/ARI
      IF(DABS(AARI-GEO)-1.D-9*AARI)17,17,16
   16 SGEO=DSQRT(SGEO)

         GEOMETRIC MEAN

      GEO=SGEO+SGEO
      PIM=PIM+PIMA
      ISI=ISI+ISI
      GOTO 8

         ACCURACY WAS SUFFICIENT

   17 R=(DATAN(ARI/ANG)+PIM)*AN/ARI
      C=C+D*ANG/AANG
      GOTO 4
      END
```

Subroutines JELF and DJELF

These subroutines compute the Jacobian elliptic functions. The three Jacobian elliptic functions arise as inverse functions of elliptic integrals.

$$\text{Set } x = \int_0^{\mu} dt/\sqrt{(1-t^2)(1-k^2t^2)}$$

$$= \int_0^{\varphi} dt/\sqrt{1-k^2\sin^2 t}$$

$$= F(\varphi, k)$$

Then:

$$\mathrm{sn}\ (x, k) = \sin\varphi = \mu$$

$$\mathrm{cn}\ (x, k) = \cos\varphi = \sqrt{1-\mu^2}$$

$$\mathrm{dn}\ (x, k) = \sqrt{1-k^2\sin^2\varphi} = \sqrt{1-k^2\mu^2}$$

The value k is the modulus, ck the complementary modulus, and sck the square of the complementary modulus:

$$\mathrm{sck} = \mathrm{ck}^2 = 1 - k^2, \quad -\infty < \mathrm{sck} < \infty$$

The value sck is chosen as input parameter in order to allow for complex values of ck [k is not restricted to the range (0, 1)]. The values sn, cn, dn are frequently needed together. Therefore, the subroutines JELF and DJELF compute all three of them. This is no disadvantage since computation of all three values is no more complicated than computation of any one of them. The calculation is based on the process of the arithmetic-geometric mean together with Gauss transformation.

The degenerate case sck = 0 (that is, k = 1) must be treated separately:

$$\mathrm{sn}\ (x, 1) = \tanh x$$

$$\mathrm{cn}\ (x, 1) = \mathrm{dn}\ (x, 1) = 1/\cosh x$$

For a negative value of sck, Jacobi's modulus transformation gives:

$$\mathrm{sn}\ (x, k) = 1/k \cdot \mathrm{sn}\ (kx, 1/k)$$

$$\mathrm{cn}\ (x, k) = \mathrm{dn}\ (kx, 1/k)$$

$$\mathrm{dn}\ (x, k) = \mathrm{cn}\ (kx, 1/k)$$

This means:

1. The internal argument y is set to kx instead of x.
2. The internal square of the complementary modulus cm is set to - sck/(1-sck) instead of sck.

The process of the arithmetic-geometric mean is started with $ari_1 = 1$, $geo_1 = \sqrt{cm}$. The sequences (ari_n), (geo_n) are generated, using for definition:

$$ari_{n+1} = (ari_n + geo_n)/2$$

$$geo_{n+1} = \sqrt{ari_n \cdot geo_n}$$

Numerical experience shows that eleven iterations are sufficient to obtain satisfactory convergence for all values of sck which may be represented in floating point. The iteration process is stopped at the Nth step when $ari_{N+1} - geo_{N+1}$ is sufficiently small.

Gauss transformation gives:

$$F(\varphi, k) = (1 + k_1)\, F(\varphi_1, k_1)$$

where: $$k = \frac{2\sqrt{k_1}}{1+k_1}$$

$$\sin\varphi = \frac{(1 + k_1)\sin\varphi_1}{1 + k_1 \sin^2\varphi_1}$$

Thereby $k = \frac{2\sqrt{k_1}}{1+k_1}$ implies $ck = \frac{1 - k_1}{1 + k_1}$,

$$k_1 = \frac{1 - ck}{1 + ck}, \quad ck_1 = \frac{2\sqrt{ck}}{1 + ck}$$

Setting $ck = \frac{geo_1}{ari_1}$, $ck_{n-1} = \frac{geo_n}{ari_n}$, then

$$k_n = \frac{ari_n - ari_{n+1}}{ari_{n+1}}$$

$$1 + k_n = \frac{ari_n}{ari_{n+1}}$$

From the definition $y = F(\varphi, k)$ it follows immediately through Gauss transformation that

$$y = (1 + k_1) \ldots (1 + k_N)\, F(\varphi_N, k_N).$$

Thus, since $k_N = 0$ and therefore $F(\varphi_N, k_N) = \varphi_N$:

$$y = ari_1/ari_{N+1} \cdot \varphi_N \text{ or } \varphi_N = ari_{N+1} \cdot y$$

To obtain the Jacobian elliptic functions, the inverse transformation must be performed on φ_N.

The formula $\sin\varphi = \frac{(1 + k_1)\sin\varphi_1}{1 + k_1 \sin^2\varphi_1}$

implies $$\cos\varphi = \frac{\cos\varphi_1}{1 + k_1 \sin^2\varphi_1}\sqrt{1 - k_1^2 \sin^2\varphi_1}$$

and $$\cot\varphi = \frac{1}{1 + k_1}\cot\varphi_1\sqrt{1 - k_1^2 \sin^2\varphi_1}$$

From the last equation:

$$ari_n \cdot \cot\varphi_{n-1} = ari_{n+1}\cot\varphi_n\sqrt{1 - k_n^2 \sin^2\varphi_n}$$

Since: $$\sin\varphi_n = \frac{(1 + k_{n+1})\sin\varphi_{n+1}(1 + \cot^2\varphi_{n+1})}{(1 + k_{n+1}) + \cot^2\varphi_{n+1}}$$

$$\sin^2\varphi_n = \frac{ari_{n+1}^2(1 + \cot^2\varphi_{n+1})}{(ari_{n+1} + ari_{n+2}\cot^2\varphi_{n+1})^2}$$

and $$\sqrt{1 - k_n^2 \sin^2\varphi_n} = \frac{geo_{n+1} + ari_{n+2}\cot^2\varphi_{n+1}}{ari_{n+1} + ari_{n+2}\cot^2\varphi_{n+1}}$$

e final result is the iteration scheme:

$$c_{N+1} = ari_{N+1} \cot (y \cdot ari_{N+1})$$

$$d_{N+1} = 1$$

$$c_n = d_{n+1} \cdot c_{n+1}$$

$$d_n = \frac{c^2_{n+1} / ari_{n+2} + geo_{n+1}}{c^2_{n+1} / ari_{n+2} + ari_{n+1}}$$

r $\quad n = N, N-1, \ldots, 1.$

hen $c_1 = \cot\varphi$, and:

$$sn\ (x, k) = \frac{1}{\sqrt{1 + c_1^2}} = \frac{1}{\sqrt{1 + \cot^2\varphi}} = \sin\varphi$$

$$cn\ (x, k) = sn \cdot c_1 = \sin\varphi \cdot \cot\varphi = \cos\varphi$$

$$dn\ (x, k) = d_1 = \sqrt{1 - k^2 \sin^2\varphi}$$

```
                                                                      JELF  10
*********************************************************************JELF  20
      SUBROUTINE JELF                                                 JELF  30
                                                                      JELF  40
      PURPOSE                                                         JELF  50
         COMPUTES THE THREE JACOBIAN ELLIPTIC FUNCTIONS SN, CN, DN.   JELF  60
                                                                      JELF  70
      USAGE                                                           JELF  80
         CALL JELF(SN,CN,DN,X,SCK)                                    JELF  90
                                                                      JELF 100
      DESCRIPTION OF PARAMETERS                                       JELF 110
         SN    - RESULT VALUE SN(X)                                   JELF 120
         CN    - RESULT VALUE CN(X)                                   JELF 130
         DN    - RESULT VALUE DN(X)                                   JELF 140
         X     - ARGUMENT OF JACOBIAN ELLIPTIC FUNCTIONS              JELF 150
         SCK   - SQUARE OF COMPLEMENTARY MODULUS                      JELF 160
                                                                      JELF 170
      REMARKS                                                         JELF 180
         NONE                                                         JELF 190
                                                                      JELF 200
      SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                   JELF 210
         NONE                                                         JELF 220
                                                                      JELF 230
      METHOD                                                          JELF 240
         DEFINITION                                                   JELF 250
         X=INTEGRAL(1/SQRT((1-T*T)*(1-(K*T)**2)), SUMMED OVER         JELF 260
         T FROM 0 TO SN), WHERE K=SQRT(1-SCK).                        JELF 270
         SN*SN + CN*CN = 1                                            JELF 280
         (K*SN)**2 + DN**2 = 1.                                       JELF 290
         EVALUATION                                                   JELF 300
         CALCULATION IS DONE USING THE PROCESS OF THE ARITHMETIC      JELF 310
         GEOMETRIC MEAN TOGETHER WITH GAUSS DESCENDING TRANSFORMATION JELF 320
         BEFORE INVERSION OF THE INTEGRAL TAKES PLACE.                JELF 330
         REFERENCE                                                    JELF 340
         R. BULIRSCH, NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS AND JELF 350
                 ELLIPTIC FUNCTIONS.                                  JELF 360
                 HANDBOOK SERIES OF SPECIAL FUNCTIONS                 JELF 370
                 NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.       JELF 380
                                                                      JELF 390
*********************************************************************JELF 400
                                                                      JELF 410
      SUBROUTINE JELF(SN,CN,DN,X,SCK)                                 JELF 420
                                                                      JELF 430
                                                                      JELF 440
      DIMENSION ARI(12),GEO(12)                                       JELF 450
      TEST MODULUS                                                    JELF 460
      CM=SCK                                                          JELF 470
      Y=X                                                             JELF 480
      IF(SCK)3,1,4                                                    JELF 490
                                                                      JELF 500
    1 D=EXP(X)                                                        JELF 510
      A=1./D                                                          JELF 520
      B=A+D                                                           JELF 530
      CN=2./B                                                         JELF 540
      DN=CN                                                           JELF 550
      SN=TANH(X)                                                      JELF 560
C        DEGENERATE CASE SCK=0 GIVES RESULTS                          JELF 570
C           CN X = DN X = 1/COSH X                                    JELF 580
C           SN X = TANH X                                             JELF 590
    2 RETURN                                                          JELF 600
C        JACOBIS MODULUS TRANSFORMATION                               JELF 610
    3 D=1.-SCK                                                        JELF 620
      CM=-SCK/D                                                       JELF 630
      D=SQRT(D)                                                       JELF 640
      Y=D*X                                                           JELF 650
    4 A=1.                                                            JELF 660
      DN=1.                                                           JELF 670
      DO 6 I=1,12                                                     JELF 680
      L=I                                                             JELF 690
      ARI(I)=A                                                        JELF 700
      CM=SQRT(CM)                                                     JELF 710
      GEO(I)=CM                                                       JELF 720
      C=(A+CM)*.5                                                     JELF 730
      IF(ABS(A-CM)-1.E-4*A)7,7,5                                      JELF 740
    5 CM=A*CM                                                         JELF 750
    6 A=C                                                             JELF 760
C                                                                     JELF 770
C     START BACKWARD RECURSION                                        JELF 780
    7 Y=C*Y                                                           JELF 790
      SN=SIN(Y)                                                       JELF 800
      CN=COS(Y)                                                       JELF 810
      IF(SN)8,13,8                                                    JELF 820
    8 A=CN/SN                                                         JELF 830
      C=A*C                                                           JELF 840
      DO 9 I=1,L                                                      JELF 850
      K=L-I+1                                                         JELF 860
      B=ARI(K)                                                        JELF 870
      A=C*A                                                           JELF 880
      C=DN*C                                                          JELF 890
      DN=(GEO(K)+A)/(B+A)                                             JELF 900
    9 A=C/B                                                           JELF 910
      A=1./SQRT(C*C+1.)                                               JELF 920
      IF(SN)10,11,11                                                  JELF 930
   10 SN=-A                                                           JELF 940
      GOTO 12                                                         JELF 950
   11 SN=A                                                            JELF 960
   12 CN=C*SN                                                         JELF 970
   13 IF(SCK)14,2,2                                                   JELF 980
   14 A=DN                                                            JELF 990
      DN=CN                                                           JELF1000
      CN=A                                                            JELF1010
      SN=SN/D                                                         JELF1020
      RETURN                                                          JELF1030
      END                                                             JELF1040
```

```
C                                                                     DJEL  10
C     ******************************************************************DJEL  20
C                                                                     DJEL  30
C        SUBROUTINE DJELF                                             DJEL  40
C                                                                     DJEL  50
C        PURPOSE                                                      DJEL  60
C           COMPUTES THE THREE JACOBIAN ELLIPTIC FUNCTIONS SN, CN, DN. DJEL  70
C                                                                     DJEL  80
C        USAGE                                                        DJEL  90
C           CALL DJELF(SN,CN,DN,X,SCK)                                DJEL 100
C                                                                     DJEL 110
C        DESCRIPTION OF PARAMETERS                                    DJEL 120
C           SN    - RESULT VALUE SN(X) IN DOUBLE PRECISION            DJEL 130
C           CN    - RESULT VALUE CN(X) IN DOUBLE PRECISION            DJEL 140
C           DN    - RESULT VALUE DN(X) IN DOUBLE PRECISION            DJEL 150
C           X     - DOUBLE PRECISION ARGUMENT OF JACOBIAN ELLIPTIC    DJEL 160
C                   FUNCTIONS                                         DJEL 170
C           SCK   - SQUARE OF COMPLEMENTARY MODULUS IN DOUBLE PRECISION DJEL 180
C                                                                     DJEL 190
C        REMARKS                                                      DJEL 200
C           NONE                                                      DJEL 210
C                                                                     DJEL 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                DJEL 230
C           NONE                                                      DJEL 240
C                                                                     DJEL 250
C        METHOD                                                       DJEL 260
C           DEFINITION                                                DJEL 270
C           X=INTEGRAL(1/SQRT((1-T*T)*(1-(K*T)**2)), SUMMED OVER      DJEL 280
C           T FROM 0 TO SN), WHERE K=SQRT(1-SCK).                     DJEL 290
C           SN*SN + CN*CN = 1                                         DJEL 300
C           (K*SN)**2 + DN**2 = 1.                                    DJEL 310
C           EVALUATION                                                DJEL 320
C           CALCULATION IS DONE USING THE PROCESS OF THE ARITHMETIC   DJEL 330
C           GEOMETRIC MEAN TOGETHER WITH GAUSS DESCENDING TRANSFORMATIONDJEL 340
C           BEFORE INVERSION OF THE INTEGRAL TAKES PLACE.             DJEL 350
C           REFERENCE                                                 DJEL 360
C           R. BULIRSCH, NUMERICAL CALCULATION OF ELLIPTIC INTEGRALS ANDDJEL 370
C                   ELLIPTIC FUNCTIONS.                               DJEL 380
C                   HANDBOOK SERIES OF SPECIAL FUNCTIONS              DJEL 390
C                   NUMERISCHE MATHEMATIK VOL. 7, 1965, PP. 78-90.    DJEL 400
C                                                                     DJEL 410
C     ******************************************************************DJEL 420
C                                                                     DJEL 430
      SUBROUTINE DJELF(SN,CN,DN,X,SCK)                                DJEL 440
C                                                                     DJEL 450
      DIMENSION ARI(12),GEO(12)                                       DJEL 460
      DOUBLE PRECISION SN,CN,DN,X,SCK,ARI,GEO,CM,Y,A,B,C,D            DJEL 470
C                                                                     DJEL 480
C        TEST MODULUS                                                 DJEL 490
C                                                                     DJEL 500
      CM=SCK                                                          DJEL 510
      Y=X                                                             DJEL 520
      IF(SCK)3,1,4                                                    DJEL 530
    1 D=DEXP(X)                                                       DJEL 540
      A=1.D0/D                                                        DJEL 550
      B=A+D                                                           DJEL 560
      CN=2.D0/B                                                       DJEL 570
      DN=CN                                                           DJEL 580
      A=(D-A)/2.D0                                                    DJEL 590
      SN=A*CN                                                         DJEL 600
C        DEGENERATE CASE SCK=0 GIVES RESULTS                          DJEL 610
C           CN X = DN X = 1/COSH X                                    DJEL 620
C           SN X = TANH X                                             DJEL 630
    2 RETURN                                                          DJEL 640
C                                                                     DJEL 650
C        JACOBIS MODULUS TRANSFORMATION                               DJEL 660
C                                                                     DJEL 670
    3 D=1.D0-SCK                                                      DJEL 680
      CM=-SCK/D                                                       DJEL 690
      D=DSQRT(D)                                                      DJEL 700
      Y=D*X                                                           DJEL 710
```

```
    4 A=1.D0                                                            DJEL 720
      DN=1.D0                                                           DJEL 730
      DO 6 I=1,12                                                       DJEL 740
      L=I                                                               DJEL 750
      ARI(I)=A                                                          DJEL 760
      CM=DSQRT(CM)                                                      DJEL 770
      GEO(I)=CM                                                         DJEL 780
      C=(A+CM)*.5D0                                                     DJEL 790
      IF(DABS(A-CM)-1.D-9*A)7,7,5                                       DJEL 800
    5 CM=A*CM                                                           DJEL 810
    6 A=C                                                               DJEL 820
C                                                                       DJEL 830
C         START BACKWARD RECURSION                                      DJEL 840
C                                                                       DJEL 850
    7 Y=C*Y                                                             DJEL 860
      SN=DSIN(Y)                                                        DJEL 870
      CN=DCOS(Y)                                                        DJEL 880
      IF(SN)8,13,8                                                      DJEL 890
    8 A=CN/SN                                                           DJEL 900
      C=A*C                                                             DJEL 910
      DO 9 I=1,L                                                        DJEL 920
      K=L-I+1                                                           DJEL 930
      B=ARI(K)                                                          DJEL 940
      A=C*A                                                             DJEL 950
      C=DN*C                                                            DJEL 960
      DN=(GEO(K)+A)/(B+A)                                               DJEL 970
    9 A=C/B                                                             DJEL 980
      A=1.D0/DSQRT(C*C+1.D0)                                            DJEL 990
      IF(SN)10,11,11                                                    DJEL1000
   10 SN=-A                                                             DJEL1010
      GOTO 12                                                           DJEL1020
   11 SN=A                                                              DJEL1030
   12 CN=C*SN                                                           DJEL1040
   13 IF(SCK)14,2,2                                                     DJEL1050
   14 A=DN                                                              DJEL1060
      DN=CN                                                             DJEL1070
      CN=A                                                              DJEL1080
      SN=SN/D                                                           DJEL1090
      RETURN                                                            DJEL1100
      END                                                               DJEL1110
```

APPENDIX A: ACCURACY OF SUBROUTINES

The subroutines in SSP can be broken down into three major categories from the standpoint of accuracy. They are (1) subroutines having little or no effect on accuracy, (2) subroutines whose accuracy is dependent on the characteristics of the input data, and (3) subroutines in which definite statements on accuracy can be made.

SUBROUTINES HAVING LITTLE OR NO EFFECT ON ACCURACY

The following subroutines do not materially affect the accuracy of the results, either because of the simple nature of the computation or because they do not modify the data:

ABSNT -- detection of missing data

ARRAY -- vector storage/double-dimensioned conversion

ATSE, DATSE -- table selection out of an equidistant table

ATSG, DATSG -- table selection out of a general table

ATSM, DATSM -- table selection out of a monotonic table

AVDAT -- data storage allocation

BISER -- biserial correlation coefficient

BOUND -- selection of observations within bounds

CADD -- add column of one matrix to column of another matrix

CCPY -- copy column of matrix into vector

CCUT -- partition by column

CHISQ -- χ^2 test for a contingency table

CINT -- interchange two columns

CONVT -- single-precision/double-precision conversion

CSRT -- sort matrix columns

CSUM -- sum the columns of a matrix

CTAB -- tabulate the columns of a matrix

CTIE -- adjoin two matrices by column

DCLA -- replace diagonal with scalar

DCPY -- copy diagonal of matrix into vector

FRAT, DFRAT -- used with ARAT
used with DARAT

GAUSS -- normal random numbers

GMADD -- add two general matrices

GMSUB -- subtract two general matrices

GMTRA -- transpose of a general matrix

KOLM2 -- Kolmogorov - Smirnov two-sample test

KRANK -- Kendall rank correlation

LOC -- location in compressed-stored matrix

MADD -- add two matrices

MCPY -- matrix copy

MFUN -- matrix transformation by a function

MOMEN -- first four moments

MPAIR -- Wilcoxin's signed rank test

MPRC, DMPRC -- permutations of rows or columns of a matrix

MSTR -- storage conversion

MSUB -- subtract two matrices

MTRA -- transpose a matrix

ORDER -- rearrangement of intercorrelations

PADD -- add two polynomials

PADDM -- multiply polynomial by constant and add to another polynomial

PCLA -- replace one polynomial by another

PERM -- inverse of permutation - transposition

PHI -- phi coefficient

PMPY -- multiply two polynomials

PNORM -- normalize coefficient vector of polynomial

PPRCN -- operations with permutations

PSUB -- subtract one polynomial from another

PVSUB -- substitute variable of polynomial by another polynomial

QTEST -- Cochran Q-test

RADD -- add row of one matrix to row of another matrix

RANDU -- uniform random numbers

RANK -- rank observations

RCPY -- copy row of matrix into vector

RCUT -- partition by row

RECP -- reciprocal function for MFUN

RINT -- interchange two rows

RSRT -- sort matrix rows

RSUM -- sum the rows of a matrix

RTAB -- tabulate the rows of a matrix

RTIE -- adjoin two matrices by row

SADD -- add scalar to matrix

SCLA -- matrix clear and add scalar

SCMA -- scalar-multiply column and add to another column

SDIV -- matrix divided by a scalar

SIGNT -- sign test

SMPY -- matrix multiplied by a scalar

SRANK -- Spearman rank correlation

SRATE -- survival rate

SRMA -- scalar-multiply row and add to another row

SSUB -- subtract scalar from a matrix

SUBMX -- build subset matrix

SUBST -- subset selection from observation matrix

TAB1 -- tabulation of data (one variable)

TAB2 -- tabulation of data (two variables)

TIE -- calculation of ties in ranked observations

TRACE -- cumulative percentage of eigenvalues

TTEST -- tests on population means

TWOAV -- Friedman two-way analysis of variance

UTEST -- Mann-Whitney U-test

WTEST -- Kendall coefficient of concordance

XCPY -- copy submatrix from given matrix

SUBROUTINES WHOSE ACCURACY IS DATA-DEPENDENT

The accuracy of the following subroutines cannot be predicted because it is dependent on the characteristics of the input data and on the size of the problem. The programmer using these subroutines must be aware of the limitations dictated by numerical analyses considerations. It cannot be assumed that the results are accurate simply because subroutine execution is completed. Subroutines in this category are:

ACFI, DACFI -- continued fraction interpolation

AHI, DAHI -- Aitken-Hermite interpolation

ALI, DALI -- Aitken-Lagrange interpolation

APCH, DAPCH -- construction of normal equations for a polynomial least-squares fit to a tabulated function

APFS, DAPFS -- solution of normal equations for least-squares fit

APLL, DAPLL -- construction of normal equations for least-squares fit to tabulated function

APMM, DAPMM -- Chebyshev approximation of tabulated functions

ARAT, DARAT -- least-squares rational approximation of tabulated functions

ATEIG -- eigenvalues of a real Hessenberg matrix

AUTO -- autocovariances

AVCAL -- Σ and Δ operation

CANOR -- canonical correlation

CNPS, DCNPS -- value of series expansion in Chebyshev polynomials

CORRE -- means, standard deviations, and correlations

CROSS -- cross covariances

CSPS, DCSPS -- value of series expansion in shifted Chebyshev polynomials

DBAR, DDBAR -- first derivative of a function at the border of the definition interval

DCAR, DDCAR -- first derivative of a function at the center of the definition interval

DET3, DDET3 -- differentiation of equidistantly tabulated functions using three points interpolation

DET5, DDET5 -- differentiation of equidistantly tabulated functions using five points interpolation

DGT3, DDGT3 -- differentiation of tabulated functions using three points interpolation

DISCR -- discriminant functions

DMATX -- means and dispersion matrix

EIGEN -- eigenvalues and eigenvectors of a real, symmetric matrix

EXSMO -- triple exponential smoothing

FACTR -- factorization of a matrix into a product of triangular matrices

FMCG, DFMCG -- unconstrained minimum of a function of several variables--conjugate gradient method

FMFP, DFMFP -- unconstrained minimum of a function of several variables--Davidon method

FORIF -- Fourier analysis of a given function

FORIT -- Fourier analysis of a tabulated function

GDATA -- data generation

GELB, DGELB -- system of general simultaneous linear equations with band-structured coefficients

GELG, DGELG -- system of general simultaneous linear equations by Gauss elimination

GELS, DGELS -- system of general simultaneous linear equations with symmetric coefficients

GMPRD -- product of two general matrices

GTPRD -- transpose product, general matrices

HARM, DHARM -- complex three-dimensional analysis

HEPS, DHEPS -- value of series expansion in Hermite polynomials

HPCG, DHPCG -- solution of general system of first-order ordinary differential equations with given initial values by Hamming's modified predictor-corrector method

HPCL, DHPCL -- solution of linear system of first-order ordinary differential equations with given initial values by Hamming's modified predictor-corrector method

HSBG -- reduction of a real matrix to Hessenberg form

KOLMO -- Kolmogorov-Smirnov one-sample test

LAPS, DLAPS -- value of series expansion in Laguerre

LBVP, DLBVP -- solution of system of linear first-order ordinary differential equations with linear boundary conditions by method of adjoint equations

LEPS, DLEPS -- value of series expansion in Legendre polynomials

LLSQ, DLLSQ -- solution of linear least-squares problems

LOAD -- factor loading

MCHB, DMCHB -- factorization of a symmetric positive definite band matrix

MATA -- transpose product of a matrix

MEANQ -- mean square operation

MFGR, DMFGR -- matrix factorization and rank determination

MFSD, DMFSD -- factorizes a symmetric positive definite matrix (square root method)

MFSS, DMFSS -- rank determination and factorization of symmetric positive semidefinite matrix

MINV -- matrix inversion

MISR -- missing value correlations

MLSS, DMLSS -- solve a system of linear equations with symmetric positive semidefinite matrix

MPRD -- matrix product

MTDS, DMTDS -- division of a matrix by a triangular matrix

MULTR -- multiple regression and correlation

NROOT -- eigenvalues and eigenvectors of a special nonsymmetric matrix

PCLD -- complete linear division

PDER -- derivative of a polynomial

PDIV -- divide polynomials

PECN, DPECN -- economization of a polynomial for symmetric range

PECS, DPECS -- economization of a polynomial for unsymmetric range

PFQB, DPFQB -- determination of a quadratic factor of a polynomial

PGCD -- greatest common divisor of two polynomials

PILD -- evaluate polynomial and its first derivative

PINT -- integral of a polynomial

POINT -- point biserial correlation

POLRT -- real and complex roots of a real polynomial

PQSD -- quadratic synthetic division of a polynomial

PRBM, DPRBM -- roots of a polynomial with real coefficients--Bairstow's method

PROBT -- probit analysis

PRQD, DPRQD -- roots of a real polynomial by QD algorithm with displacement

PVAL -- evaluate polynomial

QA2-QA10, DQA4-DQA32 -- integration of a given function by associated Gaussian-Laguerre quadrature formulas

QATR, DQATR -- integration of a given function by trapezoidal rule together with Romberg's extrapolation method

QG2-QG10, DQG4-DQG32 -- integration of a given function by Gaussian quadrature formulas

QH2 QH10, DQH8-DQH64 -- integration of a given function by Gaussian - Hermite quadrature formulas

QHFE, DQHFE -- integration of equidistantly tabulated function with first derivative by Hermitian formula of first order

QHFG, DQHFG -- integration of monotonically tabulated function with first derivative by Hermitian formula of first order

QHSG, DQHSG -- integration of monotonically tabulated function with first and second derivatives by Hermitian formula of first order

QHSE, DQHSE -- integration of equidistantly tabulated function with first and second derivatives by Hermitian formula of second order

QL2-QL10, DQL4-DQL32 -- integration of a given function by Gaussian-Laguerre quadrature formulas

QSF, DQSF -- integration of equidistantly tabulated function by Simpson's rule

QTFE, DQTFE integration of equidistantly tabulated function by trapezoidal rule

QTFG, DQTFG -- integration of monotonically tabulated function by trapezoidal rule

RHARM, DRHARM -- real one-dimensional analysis

RKI -- solution of first-order differential equation by Runge-Kutta method

RK2 -- tabulated solution of first-order differential equation by Runge-Kutta method

RKGS, DRKGS -- solution of system of first-order ordinary differential equations with given initial values by the Runge-Kutta method

RSLMC -- solution of a system of linear equations

RTMI, DRTMI -- determine root within a range by Mueller's iteration

RTNI, DRTNI -- refine estimate of root by Newton's iteration

RTWI, DRTWI -- refine estimate of root by Wegstein's iteration

SE13, DSE13 -- local least-squares smoothing of equidistantly tabulated functions

SE15, DSE15 local least-squares smoothing of equidistantly tabulated functions

SE35, DSE35 -- local least-squares smoothing of equidistantly tabulated functions

SG13, DSG13 -- local least-squares smoothing of tabulated functions

SIMQ -- solution of simultaneous linear, algebraic equations

SINV, DSINV -- inversion of a symmetric positive definite matrix

SMO -- application of filter coefficients (weights)

STPRG -- stepwise regression

TALLY -- totals, means, standard deviations, minimums, and maximums

TCNP, DTCNP -- transform series expansion in Chebyshev polynomials to a polynomial

TCSP, DTCSP -- transform series expansion in shifted Chebyshev polynomials to a polynomial

TETRA -- tetrachoric correlation

THEP, DTHEP -- transform series expansion in Hermite polynomials to a polynomial

TLAP, DTLAP -- transform series expansion in Laguerre polynomials to a polynomial

TLEP, DTLEP -- transform a series expansion in Legendre polynomials to a polynomial

TPRD -- transpose product

VARMX -- varimax rotation

UBROUTINES WITH DEFINITE ACCURACY HARACTERISTICS

'he subroutines in this section have accuracy charcteristics that can be specified on an individual asis. The mathematical descriptions for many of ıese subroutines contain information on truncation rror of a strictly theoretical nature. The actual nplementation of these subroutines on System/360 esults in the accuracy noted in Table 1. The tandard reference for comparing the accuracy of ıese subroutines is M. Abramowitz, I. A. Stegun, Handbook of Mathematical Functions, National Bureau f Standards, Washington, D. C., March 1965. However, in certain cases, other tables were used, as oted below. It should be remembered that in System /360 single-precision floating point, there are just over six significant figures.

Maximum differences below are given in terms of number of decimal places (D. P.) and/or number of significant digits (S. D.) which agree. The number of digits tabled should be considered when accuracy statements are viewed; that is, certain tables are given to only five places, whereas the algorithms used may be more accurate. In compiling maximum differences, the maximum was taken over the set of points indicated in the table. The average difference was normally much smaller.

The notation x = a (b) c implies that a, a + b, a + 2b, . . . , c were the arguments (x) used.

TABLE 1

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
BDTR	$p=I_x(a,b)$	$I_x^{-1}(a,b)$ Tables by Leon H. Harter: New Tables of the Incomplete Gamma Function Ratio and of Percentage Points of the χ^2 and Beta Distribution, 1964	p=.0001, .0005 a=1 (1) 40 b=5(5) 40 p=.0100, .0500 a= 2 (2) 10 b=5 (5) 30	correct to 5 D. P.
		"t" distribution: $t(x,n)=$ $[1+I_z(c,d)]/2$ where $z=x^2/(x^2+n)$ $c=1/2$ $d=n/2$ Tables: Biometrika vol. 1	x= .01, and x= .5 (.5) 4 for n= 1 (1) 24, and n= 30, 40, 60, 120	.5 in 5th D. P.

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
BESJ J Bessel function Parameter D=1 · E - 6	$J_n(x)$	$J_n(x)$	n=0(1)2 x=0(0.2)17.4 n=3(1)9 x=0(0.2)20 n=10(1)11 x= 10(0.2)20 n = 0(1)20 20 (10) 50 100 x= 1,2,5,10,50, 100	2 in 5th S.D. 1 in 5th S.D. for $J_n(x) > 1.0$ exact to 5 D.P. for $J_n(x) < 1.0$ (limit of table) 3 in 5th S.D. 2 in 5th S.D.
BESK K Bessel function	$K_n(x)$	$e^x K_n(x)$ $e^x K_n(x)$ $K_n(x)$	n= 0(1)2 x=0.2(0.2)20 n=3(1)9 x=0.2(0.2)10 =10(1.0) 20 n=0(1)20 x=1,2,5,10 20(10)50 50,100 100	1 in 6th S.D. 1 in 5th S.D. 2 in 5th S.D.
BESY Y Bessel function	$Y_n(x)$	$Y_n(x)$	n=0(1)2 x=0.2(0.2)17.4 n=3(1)9 x=0.2(0.2)20 n=10 x=10(0.2)20 n=0(1)20 20(10)50 100 x=1,2,5,10,50,100	3 in 5th S.D. 1 in 5th S.D. for $Y_n(x) > 1.0$ exact to 5 D.P. for $Y_n(x) < 1.0$ (limit of table) 3 in 5th S.D. 1 in 5th S.D.
CDTR	$y = P_g(x)$ where P is the χ^2 distribution function with parameter g.	$y = P_g(x)$	x=.001(.001).01; .01(.01)1.0; 1.0(.1)2.0; 2.0(.2)10.0; 10.0(.5)20.0; 20 (1) 40 40 (2) 76 for g=1(1)30	1 in the 5th D.P.

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
CEL1 Complete elliptic 1st integral	K(k)	K(m); $k = \sqrt{m}$	m = .01 (.01) .99	2 in 7th S. D.
		K(α); k = sin α	α = 1(1)73	2 in 7th S. D.
		(α in degrees)	α = 74(1)86	3 in 7th S. D.
DCEL1 Complete elliptic 1st integral	K(k)	K(m); $k = \sqrt{m}$	m = .01(.01).86	1 in 16th S. D.
			m = .87(.01) .96	4 in 16th S. D.
			m = .97(.01).99	11 in 16th S. D.
		K(α); k = sin α	α = 1(1)75	1 in 16th S. D.
		(α in degrees)	α = 76(1)80	2 in 16th S. D.
			α = 81(1)86	11 in 16th S. D.
CEL2 Generalized complete elliptic 2nd integral	K(k) with A = B = 1	K(m); $k = \sqrt{m}$	m = .01(.01) .99	2 in 7th S. D.
		K(α); k = sin α	α = 1(1) 73	2 in 7th S. D.
		(α in degrees)	α = 74(1)86	3 in 7th S. D.
	E(k) with A = 1 B = $1 - k^2$	E(m); $k = \sqrt{m}$	m = .01(01)	2 in 7th S. D.
		E(α); k = sin α	α = 1(1) 86	2 in 7th S. D.
		K'E + E'K - KK'	m = .01(.01) .99	7 in 7th S. D.
		(Legendre's relation)	α = 1(1) 89	1 in 6th S. D.
DCEL2 Generalized complete elliptic 2nd integral	K(k) with A = B = 1	K(m); $k = \sqrt{m}$	m = .01(.01).99	2 in 16th S. D.
		K(α); k = sin α (α in degrees)	α = 1(1)80	2 in 16th S. D.
			α = 81(1)86	11 in 16th S. D.
	E(k) with A = 1 B = $1 - k^2$	E(α); k = sin α	α = 1(1)89	2 in 16th S. D.
		K'E + E'K - KK'	m = .01(.01).99	9 in 16th S. D.
		(Legendre's relation)	α = 1(1)89	9 in 16th S. D.
CNP Chebyshev polynomials	T_n (x)	T_n (x) n = 0(1)12	x = .0(.2)1.0	1 in 5th D. P.
		T_n (x) n = 0(1)50	x = .00(.05)1.00	6 in 5th D. P.+

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
DCNP Chebyshev polynomials	$T_n(x)$	$T_n(x)$ $n = 0(1)12$	$x = .0(.2)1.0$	No error in 10th D. P.
CS	$C_2(u)$	$C_2(u)$	$x = .02(.02).78$ $0 < u < 1$	1 in 7th D. P.
Fresnel integrals	$S_2(u)$		$x = .80(.02)1.58$ $1 < u < 4$	6 in 7th S. D.
	with $u = \frac{\pi}{2} x^2$		$x = 1.60(.02)5.00$ $4 < u < 40$	2 in 7th S. D.
		$S_2(u)$	$x = .02(.02).78$ $0 < u < 1$	1 in 7th D. P.
			$x = .80(.02)1.58$ $1 < u < 4$	8 in 7th S. D.
			$x = 1.60(.02)5.00$ $4 < u < 40$	2 in 7th S. D.
CSP Shifted Chebyshev polynomials	$T_n^*(x)$	$T_n^*(x) = T_n(2x - 1)$ $n = 0(1)12$	$x = .5(.1)1.0$	2 in 5th D. P.
		$T_n^*(x)$ $n = 0(1)50$	$x = .00(.05)1.00$	7 in 5th D. P.+
DCSP Shifted Chebyshev polynomials	$T_n^*(x)$	$T_n^*(x) = T_n(2x - 1)$ $n = 0(1)12$	$x = .5(.1)1.0$	No error in 10th D. P.
DLGAM (log of the gamma function)	$\ln \Gamma(x)$	$\ln \Gamma(x)$	$x=1$ $x=1.005(.005)$ 1.025 $x=1.980(.005)$ 1.995 $x=1.03(.01)1.31$ $x=1.32(.01)1.67$ $x=1.68(.01)1.97$ $x=2$	6 in 9th D. P. 9 in 8th D. P. 9 in 8th S. D. 8 in 9th S. D. 8 in 10th S. D. 7 in 9th S. D. 6 in 9th S. D.
		$\log_{10} \Gamma(x)$	$x=3.0(1.0)100.0$	No error in 8 place tables

+Differences between results of single- and double-precision routines

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
ELI1 Incomplete elliptic 1st integral	$F(\zeta/\alpha)$ with $x = \tan\zeta$, $k = \sin\alpha$, $ck = \sqrt{1-k^2}$	$F(\zeta/\alpha)$ (ζ, α in degrees)	$\zeta = 0(5)10$, $\alpha = 0(2)90$	2 in 7th D. P.
			$\zeta = 15(5)35$, $\alpha = 0(2)90$	7 in 7th S. D
			$\zeta = 40(5)50$, $\alpha = 0(2)90$	11 in 7th D. P.
			$\zeta = 55(5)85$, $\alpha = 0(2)90$	3 in 7th S. D.
DELI1 Incomplete elliptic 1st integral	$F(\varphi/\alpha)$ with $x = \tan\varphi$, $k = \sin\alpha$, $ck = \sqrt{1-k^2}$	$F(\varphi/\alpha)$ (φ, α in degrees)	$\varphi = 0(5)85$, $\alpha = 0(2)90$	1 in 9th D. P. (probably due to rounding errors in table)
		$F(\varphi/\alpha) + F(\psi/\alpha) = F\left(\frac{\pi}{2}/\alpha\right)$ (φ, α, ψ in degrees)	$\varphi = 0(5)85$, $\alpha = 0(2)80$, $\psi = \text{Arctan } f$, $f = 1/(\cos\alpha \cdot \tan\varphi)$	2 in 15th D. P.
ELI2 Generalized incomplete 2nd integral	$F(\zeta/\alpha)$ with $A = B = 1$; $E(\zeta/\alpha)$ with $A = 1$ and $B = 1 - k^2$; $x = \tan\zeta$, $k = \sin\alpha$, $ck = \sqrt{1-k^2}$	$F(\zeta/\alpha)$ (ζ, α in degrees)	$\zeta = 0(5)10$, $\alpha = 0(2)90$	2 in 7th D. P.
			$\zeta = 15(5)35$, $\alpha = 0(2)90$	7 in 7th S. D.
			$\zeta = 40(5)50$, $\alpha = 0(2)90$	11 in 7th D. P.
			$\zeta = 55(5)85$, $\alpha = 0(2)90$	3 in 7th S. D.
		$E(\zeta/\alpha)$ (ζ, α in degrees)	$\zeta = 0, 5$, $\alpha = 0(2)90$	2 in 7th D. P.
			$\zeta = 10(5)35$, $\alpha = 0(2)90$	7 in 7th S. D.
			$\zeta = 40(5)55$, $\alpha = 0(2)90$	12 in 7th D. P.
			$\zeta = 60(5)85$, $\alpha = 0(2)90$	36 in 7th D. P.

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
DELI2 Generalized incomplete elliptic 2nd integral	$F(\varphi/\alpha)$ with $A = B = 1$ $E(\varphi/\alpha)$ with $A = 1$ $B = 1-k^2$ and $x = \tan\varphi$ $k = \sin\alpha$ $ck = \sqrt{1-k^2}$	$F(\varphi/\alpha)$ (φ, $/\alpha$ in degrees)	$\varphi = 0(5)85$ $\alpha = 0(2)90$	1 in 9th D. P. (probably due to rounding errors in table)
		$E(\varphi/\alpha)$ (φ, α in degrees)	$\varphi = 0(5)85$ $\alpha = 0(2)90$	1 in 9th D. P. (probably due to rounding errors in table)
		$E(\varphi/\alpha) + E(\psi/\alpha)$ $= E\left(\frac{\pi}{2}/\alpha\right) + \sin^2\alpha \sin\phi \sin\psi$ (φ, α in degrees)	$\varphi = 0(5)85$ $\alpha = 0(2)90$ $\psi = \text{Arctan } f$ $f = 1/(\cos\alpha \cdot \tan\varphi)$	2 in 15th D. P.
		$F(\varphi/\alpha) + F(\psi/\alpha)$ $= F\left(\frac{\pi}{2}/\alpha\right)$ (φ, α in degrees)	$\varphi = 0(5)85$ $\alpha = 0(2)82$ $\psi = \text{Arctan } f$ $f = 1/(\cos\alpha \cdot \tan\varphi)$	3 in 15th D. P.
EXPI Exponential integral	$-E_i(-x)$ when $x < 0$	$(E_1(x) + \ln x + \gamma)/x$	$x = .01(.01).50$	1 in 7th S. D.
	$E_1(x)$ when $x > 0$	$E_1(x)$	$x = .50(.01)1.00$	4 in 7th S. D.
		$xe^xE_1(x)$	$x = 1.00(.01)2.00$	7 in 7th S. D.
			$x = 2.0(.1)4.0$	2 in 4th S. D.
			$x = 4.0(.1)10.0$	2 in 7th S. D.
		$(-E_i(-x) + \ln\lvert x\rvert + \gamma)/x$	$x = -.01(-.01)-.50$	1 in 7th S. D.
		$-E_i(-x)$	$x = -.50(.01)-1.00$	3 in 7th S. D.
			$x = -1.00(-.01)-2.00$	4 in 7th S. D.
		$-xe^xE_i(-x)$	$x = -2.0(-.1)-4.0$	4 in 6th S. D.
GMMMA Gamma function	$\tau(x)$	$\tau(x)$	$x = 1.0(0.005)2.0$ $x = 1.0(1.0)57.0$	3 in 6th S. D. 6 in 6th S. D.
HEP Hermite polynomials	$H_n(x)$	$H_n(x)$ $n = 0(1)12$	$x = 0.5, 1, 3, 5, 10$	Maximum relative error 2.10^{-7}
		$H_n(x)$ $n = 0(1)50$	$x = 0.0(.1)1.0$ $x = 1(1)10$	Maximum relative error 3.10^{-5+}

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
DHEP Hermite polynomials	$H_n(x)$	$H_n(x)$ $n = 0(1)12$	$x = 0.5, 1, 3, 5, 10$	No error in 11th S.D.
I0	$I_0(x)$		$x = [1\ (1)\ 20]$ (500 random numbers applied per unit interval)	1.2 in 6th D.P.
INUE	$I_n(x)$ $n = 1, 2, \ldots, 10$		$x = [1\ (1)\ 20]$ (500 random arguments per unit interval)	5.5 in 6th D.P.
JELF Jacobian elliptic functions	sn u = sin φ cn u = cos φ dn u = $\sqrt{1-k^2 \sin^2\varphi}$ with φ = am u or u = $F(\varphi/\alpha)$, k = sin α sck = $1-k^2$	sn u = sin φ (φ, α in degrees) cn u = cos φ (φ, α in degrees) dn u = $\sqrt{1-k^2\sin^2\varphi}$ (φ, α in degrees) sn u cn u dn u sn u - sn(2K-u) sn u + sn(2K + u) sn u + sn(4K - u) cn u + cn(2K - u) cn u + cn(2K + u) cn u - cn(4K - u) dn u - dn(2K - u) dn u - dn(2K + u) dn u - dn(4K - u)	$\varphi = 0(1)89$ $\alpha = 0(5)85$ $\varphi = 0(1)89$ $\alpha = 0(5)85$ $\varphi = 0(1)89$ $\alpha = 0(5)85$ $k^2 = .00(.05).95$ $t = 0(1)25$ $u = t.K(k)/25$ $k^2 = .00(.05).95$ $t = 0(1)25$ $u = t.K(k)/25$ $k^2 = .00(.05).95$ $t = 0(1)25$ $u = t.K(k)/25$ $k^2 = .00(.05).90$ $t = 0(1)25$ $u = t.K(k)/25$ $k^2 = .00(.05).90$ $t = 0(1)25$ $u = t.K(k)/25$ $k^2 = .00(.05).90$ $t = 0(1)25$ $u = t.K(k)/25$	1 in 6th D.P. + 2 in 6th D.P. + 1 in 6th D.P. + 1 in 6th D.P. ++ 2 in 6th D.P. ++ 1 in 6th D.P. ++ 6 in 6th D.P. 6 in 6th D.P. 10 in 6th D.P. 4 in 6th D.P. 4 in 6th D.P. 6 in 6th D.P. 3 in 6th D.P. 3 in 6th D.P. 5 in 6th D.P.

+ Calculation of $\mu = f(\varphi/\alpha)$ with double-precision subroutine
+ Difference between result of single- and double-precision routines

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
DJELF	sn u = sin φ	sn u = sin φ (φ, α in degrees)	φ = 5(5)85 α = 0(2)90	2 in 15th D. P. +
Jacobian elliptic functions	cn u = cos φ	cn u = cos φ (φ, α in degrees)	φ = 5(5)85 α = 0(2)90	3 in 15th D. P. +
	dn u = $\sqrt{1 - k^2\alpha}$ ($\alpha = \sin^2\varphi$ with φ = am u, u = F(φ/α), k = sin α, sck = 1 - k^2)	dn u = $\sqrt{1-k^2\sin^2\varphi}$ (φ, α in degrees)	φ = 5(5)85 α = 0(2)90	2 in 15th D. P. +
		sn u - sn(2K - u)	k^2 = .00(.05).90 t = 0(1)25	5 in 15th D. P.
		sn u + sn(2K + u)	u = t.K(k)/25	5 in 15th D. P.
		sn u + sn(4K - u)		12 in 15th D. P.
		cn u + cn(2K - u)	k^2 = .00(.05).90 t = 0(1)25	3 in 15th D. P.
		cn u + cn(2K + u)	u = t.K(k)/25	3 in 15th D. P.
		cn u - cn(4K - u)		7 in 15th D. P.
		dn u - dn(2K - u)	k^2 = .00(.05).90 t = 0(1)25	3 in 15th D. P.
		dn u - dn(2K + u)	u = t.K(k)/25	2 in 15th D. P.
		dn u - dn(4K - u)		6 in 15th D. P.
LAP Laguerre polynomials	$L_n(x)$	$L_n(x)$	x = 0.5, 1, 3	2 in 6th D. P.
		n = 0(1)12	x = 5, 10	1 in 5th S. D.
		$L_n(x)$ n = 0(1)50	x = 0.0(.1)1.0 except arguments with $L_n(x) < 0.1$	Maximum relative error 4.10^{-6} ++
			x = 1(1)10 except arguments with $L_n(x) < 0.1$	Maximum relative error 1.10^{-5} ++
DLAP Laguerre polynomials	$L_n(x)$	$L_n(x)$ n = 0(1)12	x = 0.5, 1, 3, 5, 10	No error in 10th D. P.

+ Calculation of μ = F (φ/α) with double-precision routine
++ Difference between results of single- and double-precision routines

Form H20-0205-3
Page revised 2/14/69
By TNL N20-1944

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
LEP Legendre polynomials	$P_n(x)$	$P_n(x)$ n = 0,1,2,3,9,10	x = .00(.05)1.00	1 in 6th D. P.
		$P_n(x)$ n =0(1)50	x = .00(.05)1.00	5 in 6th D. P. +
DLEP Legendre Polynomials	$P_n(x)$	$P_n(x)$ n = 0,1,2,3,9,10	x = .00(.05)1.00	No error in 8th D. P.
NDTR	y = P (x) P = normal pdf.	y = P (x)	x = -6 (.01)6	7 in 7th D. P.
NDTRI	$x = P^{-1}(y)$ p = normal pdf.	$x = P^{-1}(y)$	y =.01(.01).99	5 in 4th D. P.
SICI SINE integral and cosine integral	$s_i(x)$	$s_i(x)/x$	x = .01(.01).50	1 in 7th S. D.
	$c_i(x)$	$s_i(x)$	x = .50(.01)2.00	1 in 7th S. D.
			x = 2.0(.1)10.0	1 in 7th S. D.
		$-(c_i(x) - \ln x - \gamma)/x^2$	x = .01(.01).50	1 in 7th S. D.
			x = .50(.01)1.50	2 in 7th S. D.
			x =1.50(.01)2.00	1 in 6th S. D.
		$c_i(x)$	x = 2.0(.1)4.0	2 in 6th D. P.
			x = 4.0(.1)10.0	1 in 7th D. P.

+Difference between results of single-and double-precision routines.

Name	Functions calculated	Functions checked with reference	Range checked with reference	Maximum difference
SMIRN Kolmogorov-Smirnov limiting distribution	L(x)	L (x); Tables by N. Smirnov, reprinted in Annals of Math. Stat. 19, pp. 280-281 (6- and 7- place tables). Double-precision version differences are given in parentheses in the right-hand column.	x = 0(.01) .61	1 in 6th D. P. (1 in 6th D. P.)
			x = .62	3 in 5th D. P. (see program comments) (3 in 5th D. P.)
			x = .63 (.01) 1.04	3 in 6th D. P. (2 in 6th D. P)
			x = 1.05(.01)1.15	6 in 6th D. P. (2 in 6th D. P)
			x =1.16(.01) 1.20	9 in 6th D. P. (2 in 6th D. P)
			x = 1.21 (.01) 1.45	8 in 6th D. P. (3 in 6th D. P)
			x=1.46(.01) 1.65	6 in 6th D. P. (1 in 6th D. P)
			x =1.66(.01) 1.86	2 in 6th D. P. (0 in 6th D. P)
			x =1.87	2 in 5th D. P. (2 in 5th D. P)
			x =1.88 (.01) 2.04	2 in 6th D. P. (1 in 6th D. P)
			x =2.05 (.01) 2.50	1 in 6th D. P. (1 in 6th D. P)
			x =2.51 (.01) 3.5	2 in 7th D. P. (1 in 7th D. P)

APPENDIX B: SAMPLE PROGRAM DESCRIPTIONS

DATA SCREENING

Problem Description

A set of observations is read along with information on propositions to be satisfied and limits on a selected variable. From this input a subset is obtained and a histogram of frequency over given class intervals is plotted for the selected variable. Total, average, standard deviation, minimum, and maximum are calculated for the selected variable. This procedure is repeated until all sets of input data have been processed.

Program

Description

The data screening sample program consists of a main routine, DASCR, and six subroutines; namely, SUBST, TAB1, LOC, BOOL, HIST, and MATIN. Three of these (SUBST, TAB1, LOC) are from the Scientific Subroutine Package. MATIN is a sample input routine. HIST is a program for plotting a histogram. For a description of subroutine BOOL see subroutine SUBST.

Capacity

The maximum size of matrix of observations has been set at 1000 elements, the number of observations at 200, and the number of conditions at 21. All of these can be increased as described under "Program Modification" below.

Input

Control Cards

A parameter card with the following format must precede each matrix of observations:

Columns	Contents	For Sample Problem
1 - 2	Blank	
3 - 6	Up to four-digit identification code (numeric only)	0001
7 - 10	Number of observations	0100
11 - 14	Number of variables	0004

Each matrix of observations must be followed by a card with a 9 punch in column 1. The condition matrix and bounds data are preceded by a card containing the number of conditions and the variable to be selected for analysis:

Columns	Contents	For Sample Problem
1 - 2	Number of conditions	02
3 - 4	Variable to be selected	03

A blank card after the last set of input data terminates the run.

Data Cards

1. For the observation matrix--data cards have seven fields of ten columns each. The decimal point may appear anywhere in a field. If no decimal point is included, it is assumed that the decimal point is to the right of the last digit. The number in each field may be preceded by blanks. All values for an observation are punched consecutively and may continue from card to card. However, a new observation must start in the first field of the next card.
2. For the condition matrix (see description in the subroutine SUBST)--each ten-column field contains a condition to be satisfied. The first two columns contain the variable number (right-justified), the third column the relational code, and the last seven columns of each field a floating-point number. This number may be punched anywhere in the seven-column field but must contain a decimal point. There may be as many as seven conditions per card and a total of three cards or 21 conditions.
3. For the UBO vector (see description in the subroutine TAB1)--the UBO vector is punched in three fields of ten columns each as a floating-point number with decimal point.

Deck Setup

The deck setup is shown in Figure 35.

Sample

A listing of input cards for the sample problem is presented in Figure 36. Card 002 is the parameter card for the matrix of observations. It shows that there are 100 observations and four variables in matrix number 1. Card 003 is the first observation, card 102 the last. Each observation represents an individual with four characteristics (variables). The variables are, in the order shown, (1) age in years, (2) height in inches, (3) weight in pounds, and (4) education in years.

Card 103 signals the end of the matrix of observations.

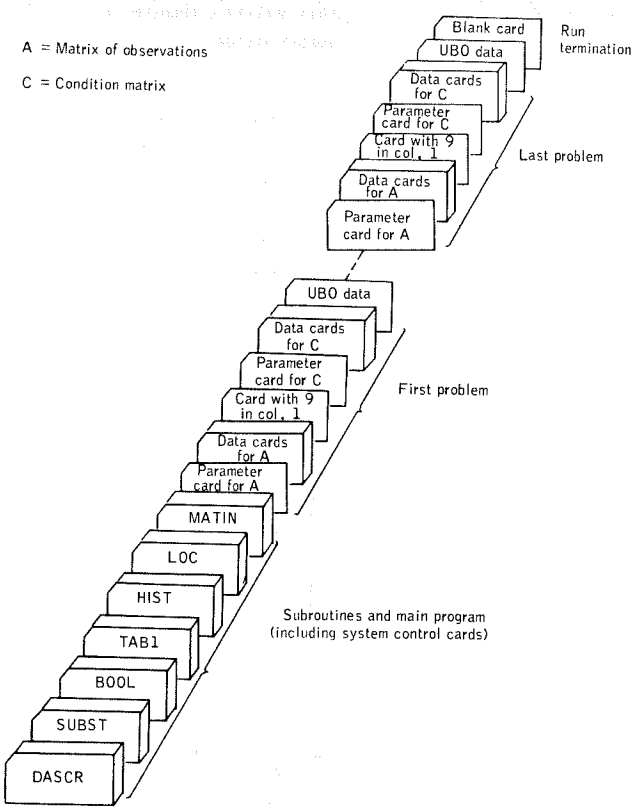


Figure 35. Deck setup (data screening)

Card 104 indicates that there are two conditions to be satisfied and that variable 3 (weight) is selected for analysis. Card 105 shows the condition matrix. The first condition states that variable 1 (age) must be less than or equal to 65, and therefore observations for individuals over 65 will be rejected. The second condition states that variable 4 (education) must be greater than eight years, and consequently observations for individuals with less than eight years of education will be rejected.

Card 106 contains the information on bounds and number of intervals for the selected variable. It shows that the lower bound is 120, the upper bound is 210, and that there are 20 intervals (including one for under lower bound and another for equal to or greater than upper bound).

Card 107 is a blank card for terminating the run.

Output

Description

The output consists of the subset vector showing which observations are rejected (zero) and accepted (nonzero), summary statistics for the selected variable, and a histogram of frequencies versus intervals for that variable.

Sample

The output listing for the sample problem is shown in Figure 37.

Program Modification

The size of the maximum problem to be solved can be increased or decreased by changing the DIMENSION statement in DASCR.

Subroutine BOOL can be replaced if the user wishes to use a different boolean expression (see description in subroutine SUBST). The boolean expression provided in the sample program is for both conditions to be satisfied:

$$T = R(1) * R(2)$$

Operating Instructions

The sample program for data screening is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and logical unit 6 is used for output. Data set 13 is used for intermediate storage.

Error Messages

The following error conditions will result in messages:

1. Reserved storage area is too small for matrix--DIMENSIONED AREA TOO SMALL FOR INPUT MATRIX. GO ON TO NEXT CASE.
2. Number of data cards does not correspond to that required by parameter card--INCORRECT NUMBER OF DATA CARDS FOR MATRIX. EXECUTION TERMINATED.

Error condition 1 allows the computer run to continue. Error condition 2, however, terminates execution and requires another run to process succeeding cases.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 42 seconds.

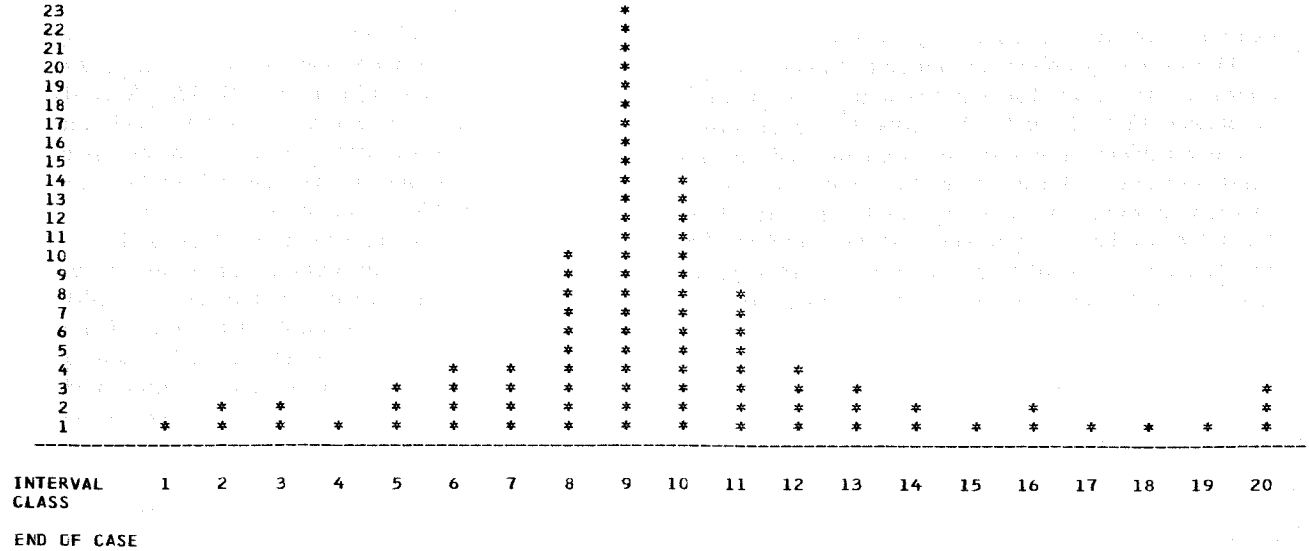
```
/DATA                                                                   10
  000101000004                                                          20
46.       64.       173.      12.                                       30
24.       72.       170.       8.                                       40
32.       71.       154.      16.                                       50
41.       68.       129.      10.                                       60
50.       65.       192.       9.                                       70
63.       75.       203.      12.                                       80
29.       70.       122.      14.                                       90
28.       64.       136.      13.                                      100
52.       77.       147.      11.                                      110
36.       67.       153.      18.                                      120
31.       68.       165.       9.                                      130
72.       70.       178.      10.                                      140
53.       71.       205.      14.                                      150
21.       65.       219.      12.                                      160
49.       63.       150.       6.                                      170
28.       62.       160.      16.                                      180
53.       72.       161.      13.                                      190
47.       73.       142.      15.                                      200
37.       67.       193.      18.                                      210
64.       58.       156.      14.                                      220
65.       60.       114.      10.                                      230
62.       64.       153.      12.                                      240
19.       68.       225.       9.                                      250
46.       67.       158.      11.                                      260
33.       72.       121.       4.                                      270
37.       65.       132.      13.                                      280
41.       76.       148.      16.                                      290
52.       71.       123.      16.                                      300
29.       68.       128.      14.                                      310
32.       65.       155.      17.                                      320
24.       72.       172.      16.                                      330
56.       73.       163.      10.                                      340
63.       65.       158.      11.                                      350
67.       69.       146.       2.                                      360
58.       66.       171.       9.                                      370
41.       65.       153.      12.                                      380
49.       66.       165.      14.                                      390
52.       72.       172.      16.                                      400
23.       78.       183.      15.                                      410
56.       71.       195.      16.                                      420
52.       68.       118.       7.                                      430
40.       66.       165.      14.                                      440
39.       68.       215.      16.                                      450
23.       71.       154.      12.                                      460
56.       65.       149.      10.                                      470
25.       65.       162.      16.                                      480
37.       68.       152.      16.                                      490
46.       70.       159.      15.                                      500
41.       69.       137.      14.                                      510
62.       71.       163.      12.                                      520
29.       72.       191.       4.                                      530
19.       68.       168.      10.                                      540
46.       63.       158.      16.                                      550
37.       64.       139.      18.                                      560
34.       68.       156.      10.                                      570
64.       67.       153.      12.                                      580
57.       67.       141.      13.                                      590
32.       68.       157.      17.                                      600
29.       70.       183.      15.                                      610
53.       72.       164.      18.                                      620
47.       72.       156.      18.                                      630
56.       73.       160.      16.                                      640
61.       74.       169.      12.                                      650
21.       68.       161.      10.                                      660
25.       76.       178.      11.                                      670
23.       72.       157.      16.                                      680
29.       68.       186.      16.                                      690
39.       70.       159.      14.                                      700
42.       70.       134.      10.                                      710
56.       62.       159.      12.                                      720
63.       70.       177.      12.                                      730
51.       71.       161.       9.                                      740
41.       66.       158.      10.                                      750
33.       69.       158.      16.                                      760
37.       68.       157.      16.                                      770
25.       70.       163.      15.                                      780
63.       68.       159.      12.                                      790
53.       71.       202.       6.                                      800
51.       72.       167.      14.                                      810
47.       73.       164.      14.                                      820
39.       75.       151.      12.                                      830
28.       68.       166.      10.                                      840
64.       59.       156.      16.                                      850
55.       67.       144.      16.                                      860
51.       66.       177.      10.                                      870
46.       65.       157.      12.                                      880
72.       66.       125.      10.                                      890
66.       65.       131.      12.                                      900
28.       74.       149.      18.                                      910
27.       71.       168.      11.                                      920
23.       72.       158.      12.                                      930
23.       72.       163.      12.                                      940
60.       68.       157.       9.                                      950
30.       66.       142.      10.                                      960
39.       67.       162.      16.                                      970
46.       74.       154.      16.                                      980
50.       68.       158.      10.                                      990
61.       66.       161.      14.                                     1000
36.       64.       157.      15.                                     1010
32.       71.       156.      16.                                     1020
9                                                                     1030
0203                                                                  1040
 12  65.    46  8.                                                    1050
  120.      20.       210.                                            1060
                                                                      1070
```

Figure 36. Input cards sample problem (data screening)

```
DATA SCREENING PROBLEM  1

SUBSET VECTOR


  1   1.
  2   0.0
  3   1.
  4   1.
  5   1.
  6   1.
  7   1.
  8   1.
  9   1.
 10   1.
 11   1.
 12   0.0
 13   1.
 14   1.
 15   0.0
 16   1.
 17   1.
 18   1.
 19   1.
 20   1.
 21   1.
 22   1.
 23   1.
 24   1.
 25   0.0
 26   1.
 27   1.
 28   1.
 29   1.
 30   1.
 31   1.
 32   1.
 33   1.
 34   0.0
 35   1.
 36   1.
 37   1.
 38   1.
 39   1.
 40   1.
 41   0.0
 42   1.
 43   1.
 44   1.
 45   1.
 46   1.
 47   1.
 48   1.
 49   1.
 50   1.
 51   0.0
 52   1.
 53   1.
 54   1.
 55   1.
 56   1.
 57   1.
 58   1.
 59   1.
 60   1.
 61   1.
 62   1.
 63   1.
 64   1.
 65   1.
 66   1.
 67   1.
 68   1.
 69   1.
 70   1.
 71   1.
 72   1.
 73   1.
 74   1.
 75   1.
 76   1.
 77   1.
 78   0.0
 79   1.
 80   1.
 81   1.
 82   1.
 83   1.
 84   1.
 85   1.
 86   1.
 87   0.0
 88   0.0
 89   1.
 90   1.
 91   1.
 92   1.
 93   1.
 94   1.
 95   1.
 96   1.
 97   1.
 98   1.
 99   1.
100   1.
```

Figure 37. Output listing (data screening)

```
SUMMARY STATISTICS FOR VARIABLE   3

TOTAL = 14492.000  AVERAGE =   161.022  STANDARD DEVIATION =    19.329  MINIMUM =   114.000  MAXIMUM =   225.000

                                                  HISTOGRAM   1

FREQUENCY    1   2   2   1   3   4   4  10  23  14   8   4   3   2   1   2   1   1   1   3
```




```
END OF CASE
```

Figure 37. Output listing (data screening) (Continued)

```
C                                                                       DASC  10
C     ...................................................................DASC  20
C                                                                       DASC  30
C        SAMPLE MAIN PROGRAM FOR DATA SCREENING - DASCR                 DASC  40
C                                                                       DASC  50
C        PURPOSE                                                        DASC  60
C           PERFORM DATA SCREENING CALCULATIONS ON A SET OF OBSERVATIONSDASC  70
C                                                                       DASC  80
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DASC  90
C           SUBST                                                       DASC 100
C           TABL                                                        DASC 110
C           LOC                                                         DASC 120
C           BOOL                                                        DASC 130
C           HIST                                                        DASC 140
C           MATIN                                                       DASC 150
C                                                                       DASC 160
C        METHOD                                                         DASC 170
C           DERIVE A SUBSET OF OBSERVATIONS SATISFYING CERTAIN          DASC 180
C           CONDITIONS ON THE VARIABLES. FOR THIS SUBSET, THE FREQUENCY DASC 190
C           OF A SELECTED VARIABLE OVER GIVEN CLASS INTERVALS IS        DASC 200
C           OBTAINED. THIS IS PLOTTED IN THE FORM OF A HISTOGRAM.       DASC 210
C           TOTAL, AVERAGE, STANDARD DEVIATION, MINIMUM, AND MAXIMUM    DASC 220
C           ARE ALSO CALCULATED.                                        DASC 230
C                                                                       DASC 240
C     ...................................................................DASC 250
C                                                                       DASC 260
      DIMENSION A(1000),C(63),UBO(3),S(200),R(21),FREQ(20),             DASC 270
     1PCT(20),STATS(5)                                                  DASC 280
      EXTERNAL BOOL                                                     DASC 290
   10 FORMAT(1H1,22HDATA SCREENING PROBLEM,I3)                          DASC 300
   11 FORMAT(1H0,44HDIMENSIONED AREA TOO SMALL FOR INPUT MATRIX ,I4)    DASC 310
   12 FORMAT(1H0,20HEXECUTION TERMINATED)                               DASC 320
   13 FORMAT(1H0,42HINCORRECT NUMBER OF DATA CARDS FOR MATRIX ,I4)      DASC 330
   14 FORMAT(1H0,18HGO ON TO NEXT CASE)                                 DASC 340
   15 FORMAT(1H0,11HEND OF CASE)                                        DASC 350
   16 FORMAT(7(F2.0,F1.0,F7.0))                                         DASC 360
   17 FORMAT(3F10.0)                                                    DASC 370
   18 FORMAT(1H0,13HSUBSET VECTOR,///)                                  DASC 380
   19 FORMAT(1H ,I3,F5.0)                                               DASC 390
   20 FORMAT(1H1,32HSUMMARY STATISTICS FOR VARIABLE ,I3)                DASC 400
   21 FORMAT(1H0,7HTOTAL =,F10.3,2X,9HAVERAGE =,F10.3,2X,20HSTANDARD DEVDASC 410
     1IATION =,F10.3,2X,9HMINIMUM =,F10.3,2X,9HMAXIMUM =,F10.3)         DASC 420
   22 FORMAT(2I2)                                                       DASC 430
C                                                                       DASC 440
      KC=0                                                              DASC 450
   24 KC=KC+1                                                           DASC 460
      CALL MATIN(ICOD,A,1000,NO,NV,MS,IER)                              DASC 470
      IF(NO) 25,50,25                                                   DASC 480
   25 IF(IER-1) 40,30,35                                                DASC 490
   30 WRITE(6,11) ICOD                                                  DASC 500
      WRITE(6,14)                                                       DASC 510
      GO TO 24                                                          DASC 520
   35 WRITE(6,13)                                                       DASC 530
      WRITE(6,12)                                                       DASC 540
      GO TO 50                                                          DASC 550
   40 READ(5,22)NC,NOVAR                                                DASC 560
      JC=NC*3                                                           DASC 570
      READ(5,16)(C(I),I=1,JC)                                           DASC 580
      READ(5,17)(UBO(I),I=1,3)                                          DASC 590
      CALL SUBST(A,C,R,BOOL,S,NO,NV,NC)                                 DASC 600
      WRITE(6,10)KC                                                     DASC 610
      WRITE(6,18)                                                       DASC 620
      WRITE(6,19) (I,S(I),I=1,NO)                                       DASC 630
      CALL TABL(A,S,NOVAR,UBO,FREQ,PCT,STATS,NO,NV)                     DASC 640
      WRITE(6,20) NOVAR                                                 DASC 650
      WRITE(6,21)(STATS(I),I=1,5)                                       DASC 660
      JZ=UBO(2)                                                         DASC 670
      CALL HIST(KC,FREQ,JZ)                                             DASC 680
      WRITE(6,15)                                                       DASC 690
      GO TO 24                                                          DASC 700
   50 RETURN                                                            DASC 710
      END                                                               DASC 720
```

MULTIPLE LINEAR REGRESSION

Problem Description

Multiple linear regression analysis is performed for a set of independent variables and a dependent variable. Selection of different sets of independent variables and designation of a dependent variable can be made as many times as desired.

The sample problem for multiple linear regression consists of 30 observations with six variables as presented in Table 4. The first five variables are independent variables (predictors), and the last variable is the dependent variable (criteria). All five independent variables are used to predict the dependent variable in the first analysis, and only the second, third, and fifth variables are used to predict the dependent variable in the second analysis.

Table 4. Sample Data for Multiple Linear Regression

Observation	Variables X_1	X_2	X_3	X_4	X_5	X_6
1	29	289	216	85	14	1
2	30	391	244	92	16	2
3	30	424	246	90	18	2
4	30	313	239	91	10	0
5	35	243	275	95	30	2
6	35	365	219	95	21	2
7	43	396	267	100	39	3
8	43	356	274	79	19	2
9	44	346	255	126	56	3
10	44	156	258	95	28	0
11	44	278	249	110	42	4
12	44	349	252	88	21	1
13	44	141	236	129	56	1
14	44	245	236	97	24	1
15	45	297	256	111	45	3
16	45	310	262	94	20	2
17	45	151	339	96	35	3
18	45	370	357	88	15	4
19	45	379	198	147	64	4
20	45	463	206	105	31	3
21	45	316	245	132	60	4
22	45	280	225	108	36	4
23	44	395	215	101	27	1
24	49	139	220	136	59	0
25	49	245	205	113	37	4
26	49	373	215	88	25	1
27	51	224	215	118	54	3
28	51	677	210	116	33	4
29	51	424	210	140	59	4
30	51	150	210	105	30	0

Program

Description

The multiple linear regression program consists of the main routine named REGRE, a special input subroutine named DATA, and four subroutines from the Scientific Subroutine Package: CORRE, ORDER, MINV, and MULTR.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to 40 variables, including both independent and dependent variables
2. Up to 99,999 observations, if observations are read into the computer one at a time by the special input subroutine named DATA. If all data are to be stored in core prior to the calculation of correlation coefficients, the limitation on the number of observations depends on the size of core storage available for input data
3. (12F6.0) format for input data cards.

Therefore, if a problem satisfies the above conditions, the sample program need not be modified. However, if there are more than 40 variables, dimension statements in the sample main program must be modified to handle this particular problem. Similarly, if input data cards are prepared using a different format, the input format in the input subroutine, DATA, must be modified. The general rules for program modification are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, REGRE. This card is prepared as follows:

Columns	Contents	For Sample Problem
1 - 6	Problem number (may be alphameric)	SAMPLE
7 - 11	Number of observations	00030
12 - 13	Number of variables	06
14 - 15	Number of selection cards (see below)	02

Leading zeros are not required to be keypunched.

Data Cards

Since input data are read into the computer one observation at a time, each row of data in Table 4 is keypunched on a separate card using the format (12F6.0). This format assumes twelve 6-column fields per card.

If there are more than twelve variables in a problem, each row of data is continued on the second and third cards until the last data point is keypunched. However, each row of data must begin on a new card.

Selection Card

The selection card is used to specify a dependent variable and a set of independent variables in a multiple linear regression analysis. Any variable in the set of original variables can be designated as a dependent variable, and any number of variables can be specified as independent variables. Selection of a dependent variable and a set of independent variables can be performed over and over again using the same set of original variables.

The selection card is prepared as follows:

Columns	Contents	For Sample Problem Selection 1	For Sample Problem Selection 2
1 - 2	Option code for table of residuals: 00 if it is not desired 01 if it is desired	01	01
3 - 4	Dependent variable designated for the forthcoming regression	06	06
5 - 6	Number of independent variables included in the forthcoming regression (the subscript numbers of individual variables are specified below)	05	03
7 - 8	1st independent variable included	01	02
9 - 10	2nd independent variable included	02	03
11 - 12	3rd independent variable included	03	05
13 - 14	4th independent variable included	04	
15 - 16	5th independent variable included	05	
etc.			

The input format of (36I2) is used for the selection card.

Deck Setup

The deck setup is shown in Figure 38.

Sample

The listing of input cards for the sample problem is presented in Figure 39.

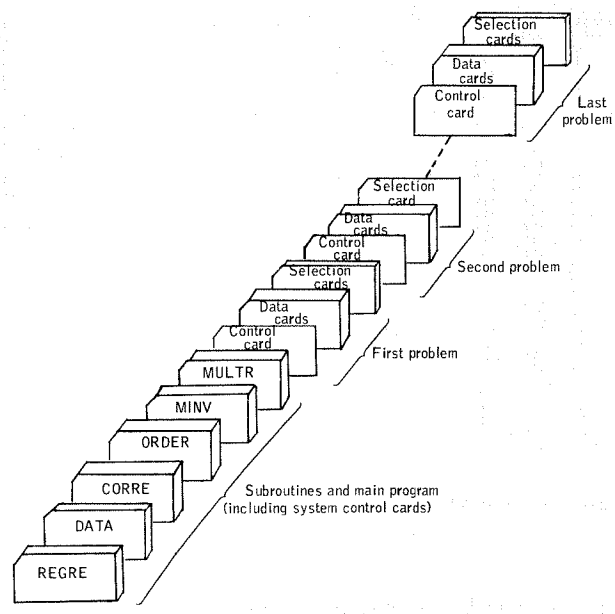


Figure 38. Deck setup (multiple linear regression)

Output

Description

The output of the sample program for multiple linear regression includes:

1. Means
2. Standard deviations
3. Correlation coefficients between the independent variables and the dependent variable
4. Regression coefficients
5. Standard errors of regression coefficients
6. Computed t-values
7. Intercept
8. Multiple correlation coefficients
9. Standard error of estimate
10. Analysis of variance for the multiple regression
11. Table of residuals (optional)

Sample

The output listing for the sample problem is shown in Figure 40.

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize

```
/DATA                                                                  10
SAMPLE000300602                                                        20
    29  289  216   95   14    1                                        30
    30  391  244   92   16    2                                        40
    30  424  246   90   18    2                                        50
    30  313  239   91   10    0                                        60
    35  243  275   95   30    2                                        70
    35  365  219   95   21    2                                        80
    43  396  267  100   39    3                                        90
    43  356  274   79   19    2                                       100
    44  346  255  126   56    3                                       110
    44  156  258   95   28    0                                       120
    44  278  249  110   42    4                                       130
    44  349  252   88   21    1                                       140
    44  141  236  129   56    1                                       150
    44  245  236   97   24    1                                       160
    45  297  256  111   45    3                                       170
    45  310  262   94   20    2                                       180
    45  151  339   96   35    3                                       190
    45  370  357   88   15    4                                       200
    45  379  198  147   64    4                                       210
    45  463  206  105   31    3                                       220
    45  316  245  132   60    4                                       230
    45  280  225  108   36    4                                       240
    44  395  215  101   27    1                                       250
    49  139  220  136   59    0                                       260
    49  245  205  113   37    4                                       270
    49  373  215   88   25    1                                       280
    51  224  215  118   54    3                                       290
    51  677  210  116   33    4                                       300
    51  424  210  140   59    4                                       310
    51  150  210  105   30    0                                       320
010605C102030405                                                       330
010603020305                                                           340
```

Figure 39. Input listing (multiple regression)

```
MULTIPLE REGRESSION.....SAMPLE

     SELECTION..... 1


VARIABLE      MEAN        STANDARD      CORRELATION    REGRESSION    STD. ERROR     COMPUTED
 NO.                      DEVIATION       X VS Y       COEFFICIENT   OF REG.COEF.    T VALUE
  1       43.13333        6.52176        0.28422        0.01242       0.03635       0.34171
  2      316.16650      114.42990        0.42189        0.00739       0.00186       3.96545
  3      241.79999       36.43074        0.11500        0.01504       0.00635       2.36881
  4      105.66666       17.85640        0.37822        0.00151       0.03679       0.04100
  5       34.13333       15.97571        0.39412        0.04919       0.04141       1.18782
DEPENDENT
  6        2.26667        1.41259


INTERCEPT                   -6.07928

MULTIPLE CORRELATION         0.73575

STD. ERROR OF ESTIMATE       1.05162


                    ANALYSIS OF VARIANCE FOR THE REGRESSION

    SOURCE OF VARIATION        DEGREES      SUM OF        MEAN          F VALUE
                             OF FREEDOM    SQUARES       SQUARES
ATTRIBUTABLE TO REGRESSION        5        31.32506      6.26501        5.66508
DEVIATION FROM REGRESSION        24        26.54161      1.10590
     TOTAL                       29        57.86667


MULTIPLE REGRESSION.....SAMPLE

     SELECTION..... 1


                TABLE OF RESIDUALS

CASE NO.     Y VALUE     Y ESTIMATE     RESIDUAL
    1        1.00000      0.48091       0.51909
    2        2.00000      1.77670       0.22330
    3        2.00000      2.14586      -0.14586
    4        0.0          0.82880      -0.82880
    5        2.00000      1.90522       0.09478
    6        2.00000      1.52125       0.47875
    7        3.00000      3.46447      -0.46447
    8        2.00000      2.25887      -0.25887
    9        3.00000      3.80259      -0.80259
   10        0.0          1.02042      -1.02042
   11        4.00000      2.49735       1.50265
   12        1.00000      2.00066      -1.00066
   13        1.00000      2.00735      -1.00735
   14        1.00000      1.15308      -0.15308
   15        3.00000      2.90446       0.09554
   16        2.00000      1.83532       0.16468
   17        3.00000      2.56004       0.43996
   18        4.00000      3.45229       0.54771
   19        4.00000      3.62661       0.37339
   20        3.00000      2.68068       0.31932
   21        4.00000      3.64885       0.35115
   22        4.00000      1.86542       2.13458
   23        1.00000      2.09863      -1.09863
   24        0.0          1.97217      -1.97217
   25        4.00000      1.41253       2.58747
   26        1.00000      1.88027      -0.88027
   27        3.00000      2.27646       0.72354
   28        4.00000      4.51080      -0.51080
   29        4.00000      3.95745       0.04255
   30        0.0          0.45456      -0.45456
```

Figure 40. Output listing (multiple regression)

the user with the program modification, the following general rules are supplied in terms of the sample problem:

1. Changes in the dimension statements of the main program, REGRE
   a. The dimension of arrays XBAR, STD, D, RY, ISAVE, B, SB, T, and W must be greater than or equal to the number of variables, m. Since there are six variables in the sample problem, the value of m is 6.
   b. The dimension of array RX must be greater than or equal to the product of m x m. For the sample problem this product is 36 = 6 x 6.
   c. The dimension of array R must be greater than or equal to (m + 1)m/2. For the sample problem this number is 21 = (6 + 1)6/2.

2. Changes in the input format statement of the special input subroutine, DATA:

   Only the format statement for input data may be changed. Since sample data are either one-, two-, or three-digit numbers, rather than using six-column fields as in the sample problem, each row of data may be keypunched in six 3-column fields, and, if so, the format is changed to (6F3.0).

   The special input subroutine, DATA, is normally written by the user to handle different formats for different problems. The user may modify this subroutine to perform testing of input data, transformation of data, and so on.

Operating Instructions

The sample program for multiple linear regression is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output. A scratch tape (data set 13) is used as intermediate storage.

Error Messages

The following error conditions will result in messages:

1. The number of selection cards is not specified on the control card--NUMBER OF SELECTIONS NOT SPECIFIED. JOB TERMINATED.
2. The matrix of correlation coefficients is singular--THE MATRIX IS SINGULAR. THIS SELECTION IS SKIPPED.

Error condition 2 allows the computer run to continue; however, error condition 1 terminates execution of the job.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 42 seconds.

```
C                                                                       REGR  10
C     ..................................................................REGR  20
C                                                                       REGR  30
C        SAMPLE MAIN PROGRAM FOR MULTIPLE REGRESSION - REGRE            REGR  40
C                                                                       REGR  50
C        PURPOSE                                                        REGR  60
C           (1) READ THE PROBLEM PARAMETER CARD FOR A MULTIPLE REGRES-  REGR  70
C           SION, (2) READ SUBSET SELECTION CARDS, (3) CALL THE SUB-    REGR  80
C           ROUTINES TO CALCULATE MEANS, STANDARD DEVIATIONS, SIMPLE    REGR  90
C           AND MULTIPLE CORRELATION COEFFICIENTS, REGRESSION COEFFI-   REGR 100
C           CIENTS, T-VALUES, AND ANALYSIS OF VARIANCE FOR MULTIPLE     REGR 110
C           REGRESSION, AND (4) PRINT THE RESULTS.                      REGR 120
C                                                                       REGR 130
C        REMARKS                                                        REGR 140
C           THE NUMBER OF OBSERVATIONS, N, MUST BE GREATER THAN M+1,    REGR 150
C           WHERE M IS THE NUMBER OF VARIABLES.  IF SUBSET SELECTION    REGR 160
C           CARDS ARE NOT PRESENT, THE PROGRAM CAN NOT PERFORM MULTIPLE REGR 170
C           REGRESSION.                                                 REGR 180
C           AFTER RETURNING FROM SUBROUTINE MINV, THE VALUE OF DETER-   REGR 190
C           MINANT (DET) IS TESTED TO CHECK WHETHER THE CORRELATION     REGR 200
C           MATRIX IS SINGULAR.  IF DET IS COMPARED AGAINST A SMALL     REGR 210
C           CONSTANT, THIS TEST MAY ALSO BE USED TO CHECK NEAR-         REGR 220
C           SINGULARITY.                                                REGR 230
C                                                                       REGR 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  REGR 250
C           CORRE  (WHICH, IN TURN, CALLS THE SUBROUTINE NAMED DATA)    REGR 260
C           ORDER                                                       REGR 270
C           MINV                                                        REGR 280
C           MULTR                                                       REGR 290
C                                                                       REGR 300
C        METHOD                                                         REGR 310
C           REFER TO B. OSTLE, 'STATISTICS IN RESEARCH', THE IOWA STATE REGR 320
C           COLLEGE PRESS', 1954, CHAPTER 8.                            REGR 330
C                                                                       REGR 340
C     ..................................................................REGR 350
C                                                                       REGR 360
C     THE FOLLOWING  DIMENSIONS MUST BE GREATER THAN OR EQUAL TO THE    REGR 370
C     NUMBER OF VARIABLES, M..                                          REGR 380
C                                                                       REGR 390
      DIMENSION XBAR(40),STD(40),D(40),RY(40),ISAVE(40),B(40),          REGR 400
     1          SB(40),T(40),W(40)                                      REGR 410
C                                                                       REGR 420
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      REGR 430
C     PRODUCT OF M*M..                                                  REGR 440
C                                                                       REGR 450
      DIMENSION RX(1600)                                                REGR 460
C                                                                       REGR 470
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO          REGR 480
C     (M+1)*M/2..                                                       REGR 490
C                                                                       REGR 500
      DIMENSION R(820)                                                  REGR 510
C                                                                       REGR 520
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO 10..     REGR 530
C                                                                       REGR 540
      DIMENSION ANS(10)                                                 REGR 550
C                                                                       REGR 560
C     ..................................................................REGR 570
C                                                                       REGR 580
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  REGR 590
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      REGR 600
C        STATEMENT WHICH FOLLOWS.                                       REGR 610
C                                                                       REGR 620
C     DOUBLE PRECISION XBAR,STD,RX,R,D,B,T,RY,DET,SB,ANS,SUM            REGR 630
C                                                                       REGR 640
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    REGR 650
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      REGR 660
C        ROUTINE.                                                       REGR 670
C                                                                       REGR 680
C        ...............................................................REGR 690
C                                                                       REGR 700
    1 FORMAT(A4,A2,I5,2I2)                                              REGR 710
    2 FORMAT(25H1MULTIPLE REGRESSION.....A4,A2//6X,14HSELECTION.....I2//REGR 720
     1)                                                                 REGR 730
    3 FORMAT(9H0VARIABLE,5X,4HMEAN,6X,8HSTANDARD,6X,11HCORRELATION,4X,10REGR 740
     1HREGRESSION,4X,10HSTD. ERROR,5X,8HCOMPUTED/6H   NO.,18X,9HDEVIATIOREGR 750
     2N,7X,6HX VS Y,7X,11HCOEFFICIENT,3X,12HOF REG.COEF.,3X,7HT VALUE)  REGR 760
    4 FORMAT(1H ,I4,6F14.5)                                             REGR 770
    5 FORMAT(10H DEPENDENT)                                             REGR 780
    6 FORMAT(1H0/10H INTERCEPT,10X,F15.5//23H MULTIPLE CORRELATION  ,F13REGR 790
     1.5//23H STD. ERROR OF ESTIMATE,F13.5//)                           REGR 800
    7 FORMAT(1H0,21X,39HANALYSIS OF VARIANCE FOR THE REGRESSION//5X,19HSREGR 810
     1OURCE OF VARIATION,7X,7HDEGREES,7X,6HSUM OF,10X,4HMEAN,12X,7HF VALREGR 820
     2UE/30X,10HOF FREEDOM,4X,7HSQUARES,9X,7HSQUARES)                   REGR 830
    8 FORMAT(30H ATTRIBUTABLE TO REGRESSION   ,I6,3F16.5/30H DEVIATION FREGR 840
     1ROM REGRESSION     ,I6,2F16.5)                                    REGR 850
    9 FORMAT(1H ,5X,5HTOTAL,19X,I6,F16.5)                               REGR 860
   10 FORMAT(36I2)                                                      REGR 870
   11 FORMAT(1H ,15X,18HTABLE OF RESIDUALS//9H CASE NO.,5X,7HY VALUE,5X,REGR 880
     110HY ESTIMATE,6X,8HRESIDUAL)                                      REGR 890
   12 FORMAT(1H ,I6,F15.5,2F14.5)                                       REGR 900
   13 FORMAT(53H1NUMBER OF SELECTIONS NOT SPECIFIED.  JOB TERMINATED.)  REGR 910
   14 FORMAT(52H0THE MATRIX IS SINGULAR.  THIS SELECTION IS SKIPPED.)   REGR 920
C                                                                       REGR 930
C     ..................................................................REGR 940
C                                                                       REGR 950
C     READ PROBLEM PARAMETER CARD                                       REGR 960
C                                                                       REGR 970
  100 READ (5,1) PR,PR1,N,M,NS                                          REGR 980
C        PR........PROBLEM NUMBER (MAY BE ALPHAMERIC)                   REGR 990
C        PR1.......PROBLEM NUMBER (CONTINUED)                           REGR1000
C        N.........NUMBER OF OBSERVATIONS                               REGR1010
C        M.........NUMBER OF VARIABLES                                  REGR1020
C        NS........NUMBER OF SELECTIONS                                 REGR1030
C                                                                       REGR1040
C     LOGICAL TAPE 13 IS USED AS INTERMEDIATE STORAGE TO HOLD INPUT     REGR1050
C     DATA.  THE INPUT DATA ARE WRITTEN ON LOGICAL TAPE 13 BY THE       REGR1060
C     SPECIAL INPUT SUBROUTINE NAMED DATA.  THE STORED DATA MAY BE USED REGR1070
C     FOR RESIDUAL ANALYSIS.                                            REGR1080
C                                                                       REGR1090
      REWIND 13                                                         REGR1100
C                                                                       REGR1110
      IO=0                                                              REGR1120
      X=0.0                                                             REGR1130
C                                                                       REGR1140
      CALL CORRE (N,M,IO,X,XBAR,STD,RX,R,D,B,T)                         REGR1150
C                                                                       REGR1160
      REWIND 13                                                         REGR1170
C                                                                       REGR1180
C     TEST NUMBER OF SELECTIONS                                         REGR1190
C                                                                       REGR1200
      IF(NS) 108, 108, 109                                              REGR1210
  108 WRITE (6,13)                                                      REGR1220
      GO TO 300                                                         REGR1230
C                                                                       REGR1240
  109 DO 200 I=1,NS                                                     REGR1250
      WRITE (6,2) PR,PR1,I                                              REGR1260
C                                                                       REGR1270
C     READ SUBSET SELECTION CARD                                        REGR1280
C                                                                       REGR1290
      READ (5,10) NRESI,NDEP,K,(ISAVE(J),J=1,K)                         REGR1300
C        NRESI.....OPTION CODE FOR TABLE OF RESIDUALS                   REGR1310
C                   0  IF IT IS NOT DESIRED.                            REGR1320
C                   1  IF IT IS DESIRED.                                REGR1330
C        NDEP......DEPENDENT VARIABLE                                   REGR1340
C        K.........NUMBER OF INDEPENDENT VARIABLES INCLUDED             REGR1350
C        ISAVE.....A VECTOR CONTAINING THE INDEPENDENT VARIABLES        REGR1360
C                  INCLUDED                                             REGR1370
C                                                                       REGR1380
      CALL ORDER (M,R,NDEP,K,ISAVE,RX,RY)                               REGR1390
C                                                                       REGR1400
      CALL MINV (RX,K,DET,B,T)                                          REGR1410
C                                                                       REGR1420
C     TEST SINGULARITY OF THE MATRIX INVERTED                           REGR1430
C                                                                       REGR1440
      IF(DET) 112, 110, 112                                             REGR1450
  110 WRITE (6,14)                                                      REGR1460
      GO TO 200                                                         REGR1470
C                                                                       REGR1480
  112 CALL MULTR (N,K,XBAR,STD,D,RX,RY,ISAVE,B,SB,T,ANS)                REGR1490
C                                                                       REGR1500
C     PRINT MEANS, STANDARD DEVIATIONS, INTERCORRELATIONS BETWEEN       REGR1510
C     X AND Y, REGRESSION COEFFICIENTS, STANDARD DEVIATIONS OF          REGR1520
C     REGRESSION COEFFICIENTS, AND COMPUTED T-VALUES                    REGR1530
C                                                                       REGR1540
      MM=K+1                                                            REGR1550
      WRITE (6,3)                                                       REGR1560
      DO 115 J=1,K                                                      REGR1570
      L=ISAVE(J)                                                        REGR1580
  115 WRITE (6,4) L,XBAR(L),STD(L),RY(J),B(J),SB(J),T(J)                REGR1590
      WRITE (6,5)                                                       REGR1600
      L=ISAVE(MM)                                                       REGR1610
      WRITE (6,4) L,XBAR(L),STD(L)                                      REGR1620
C                                                                       REGR1630
C     PRINT INTERCEPT, MULTIPLE CORRELATION COEFFICIENT, AND STANDARD   REGR1640
C     ERROR OF ESTIMATE                                                 REGR1650
C                                                                       REGR1660
      WRITE (6,6) ANS(1),ANS(2),ANS(3)                                  REGR1670
C                                                                       REGR1680
C     PRINT ANALYSIS OF VARIANCE FOR THE REGRESSION                     REGR1690
C                                                                       REGR1700
      WRITE (6,7)                                                       REGR1710
      L=ANS(9)                                                          REGR1720
      WRITE (6,8) K,ANS(4),ANS(6),ANS(10),L,ANS(7),ANS(9)               REGR1730
      L=N-1                                                             REGR1740
      SUM=ANS(4)+ANS(7)                                                 REGR1750
      WRITE (6,9) L,SUM                                                 REGR1760
      IF(NRESI) 200, 200, 120                                           REGR1770
C                                                                       REGR1780
C     PRINT TABLE OF RESIDUALS                                          REGR1790
C                                                                       REGR1800
  120 WRITE (6,2) PR,PR1,I                                              REGR1810
      WRITE (6,11)                                                      REGR1820
      MM=ISAVE(K+1)                                                     REGR1830
      DO 140 II=1,N                                                     REGR1840
      READ (13) (W(J),J=1,M)                                            REGR1850
      SUM=ANS(1)                                                        REGR1860
      DO 130 J=1,K                                                      REGR1870
      L=ISAVE(J)                                                        REGR1880
  130 SUM=SUM+W(L)*B(J)                                                 REGR1890
      RESI=W(MM)-SUM                                                    REGR1900
  140 WRITE (6,12) II,W(MM),SUM,RESI                                    REGR1910
      REWIND 13                                                         REGR1920
  200 CONTINUE                                                          REGR1930
      GO TO 100                                                         REGR1940
  300 CONTINUE                                                          REGR1950
      END                                                               REGR1960
```

POLYNOMIAL REGRESSION

Problem Description

Powers of an independent variable are generated to calculate polynomials of successively increasing degrees. If there is no reduction in the residual sum of squares between two successive degrees of polynomials, the program terminates the problem before completing the analysis for the highest degree polynomial specified.

The sample problem for polynomial regression consists of 15 observations, as presented in Table 5. The highest degree polynomial specified for this problem is 4.

Program

Description

The polynomial regression program consists of the main routine named POLRG, a special plot subroutine named PLOT, and four subroutines from the Scientific Subroutine Package: GDATA, ORDER, MINV, and MULTR.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to 100 observations
2. Up to 10th degree polynomial
3. (2F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions it is not necessary to modify the sample program. However, if there are more than 100 observations or if greater than 10th degree polynomial is desired, dimension statements in the sample main program must be modified to handle this particular problem. Similarly, if input data cards are prepared using a different format, the input format in the sample main program must be modified. The general rules for program modification are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, POLRG. This card is prepared as follows:

Columns	Contents	For Sample Problem
1 - 6	Problem number (may be alphameric)	SAMPLE
7 - 11	Number of observations	00015
12 - 13	Highest degree polynomial to be fitted	04
14	Option code for plotting Y values and Y estimates: 0 if it is not desired, 1 if it is desired	1

Leading zeros are not required to be keypunched.

Data Cards

Since input data are read into the computer one observation at a time, each pair of X and Y data in Table 5 is keypunched in that order on a separate card using the format (2F6.0).

Table 5. Sample Data for Polynomial Regression

X	Y
1	10
2	16
3	20
4	23
5	25
6	26
7	30
8	36
9	48
10	62
11	78
12	94
13	107
14	118
15	127

Deck Setup

The deck setup is shown in Figure 41.

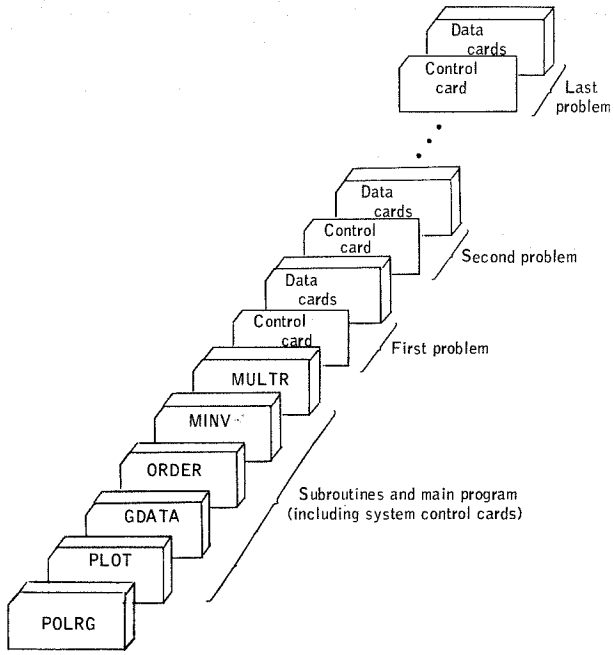


Figure 41. Deck setup (polynomial regression)

Sample

The listing of input cards for the sample problem is presented in Figure 42.

Output

Description

The output of the sample program for polynomial regression includes:

1. Regression coefficients for successive degree polynomials
2. Analysis-of-variance table for successive degree polynomials
3. Table of residuals for the final degree polynomial (optional)
4. Plot of Y values and Y estimates (optional)

```
/DATA                                                                   10
SAMPLE00015041                                                          20
     1    10                                                            30
     2    16                                                            40
     3    20                                                            50
     4    23                                                            60
     5    25                                                            70
     6    26                                                            80
     7    30                                                            90
     8    36                                                           100
     9    48                                                           110
    10    62                                                           120
    11    78                                                           130
    12    96                                                           140
    13   107                                                           150
    14   118                                                           160
    15   127                                                           170
```

Figure 42. Input card listing (polynomial regression)

Sample

The output listing for the sample problem is shown in Figure 43.

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize the user with the program modification, the following general rules are supplied in terms of the sample problem.

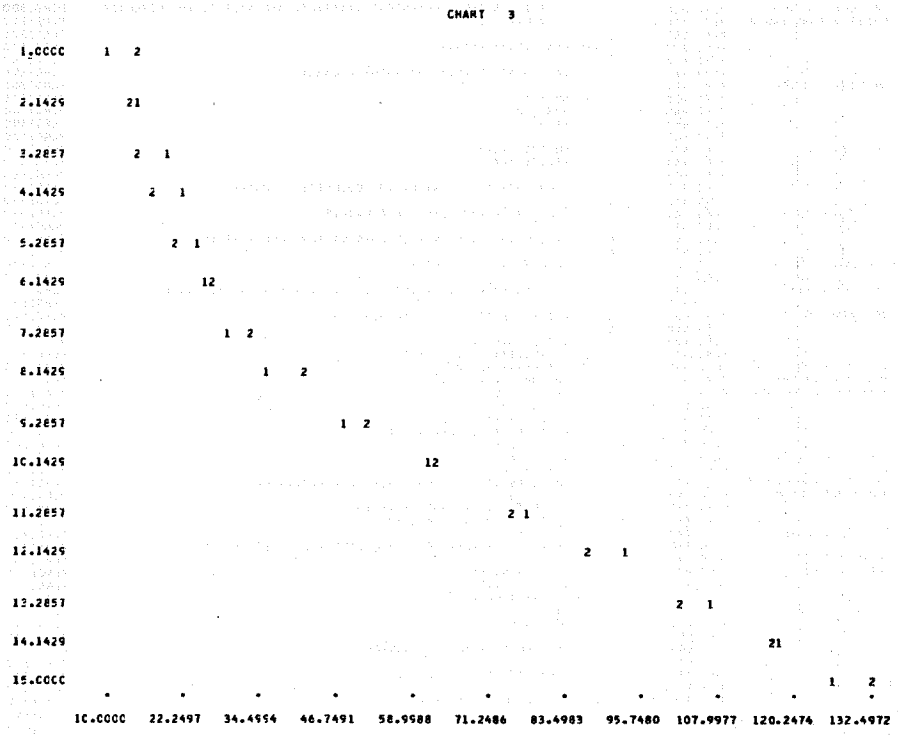
1. Changes in the dimension statements of the main program, POLRG:
   a. The dimension of array X must be greater than or equal to the product of n (m + 1), where n is the number of observations and m is the highest degree polynomial to be fitted. Since there are 15 observations and the highest degree polynomial specified is 4, the product is 75 = 15 (4 + 1).
   b. The dimension of array DI must be greater than or equal to the product of m x m. For the sample problem this product is 16 = 4 x 4.
   c. The dimension of array D must be greater than or equal to (m + 2) (m + 1)/2. For the sample problem this number is 15 = (4 + 2) (4 + 1)/2.
   d. The dimension of arrays B, E, SB, and T must be greater than or equal to the highest degree polynomial to be fitted, m. For the sample problem the value of m is 4.
   e. The dimension of arrays XBAR, STD, COE, SUMSQ, and ISAVE must be greater than or equal to (m + 1). For the sample problem this value is 5 = (4 + 1).
   f. The dimension of array P must be greater than or equal to 3(n). For the sample problem this value is 45 = 3(15). The array P is used when a plot of Y values and Y estimates is desired.

2. Changes in the input format statement of the main program, POLRG:

   Only the format statement for input data may be changed. Since sample data are either one-, two-, or three-digit numbers, rather than using six-column fields as in the sample problem, each row of data may be keypunched in two 3-column fields, and if so the format is changed to (2F3.0).

```
.YNOMIAL REGRESSION.....SAMPLE

IBER OF OBSERVATIONS    15


.YNOMIAL REGRESSION OF DEGREE  1

INTERCEPT       -13.87613

IEGRESSION COEFFICIENTS
  8.56785

                    ANALYSIS OF VARIANCE FOR   1  DEGREE POLYNOMIAL

   SOURCE OF VARIATION        DEGREE OF        SUM OF          MEAN          F          IMPROVEMENT IN TERMS
                               FREEDOM         SQUARES         SQUARE        VALUE        OF SUM OF SQUARES

DUE TO REGRESSION                 1        20554.26172     20554.26172    135.56577         20554.26172
DEVIATION ABOUT REGRESSION       13         1971.03906       151.61838
     TOTAL                       14        22525.30078


LYNOMIAL REGRESSION OF DEGREE  2

INTERCEPT       15.07547

REGRESSION COEFFICIENTS
 -1.64957      0.63856

                    ANALYSIS OF VARIANCE FOR   2  DEGREE POLYNOMIAL

   SOURCE OF VARIATION        DEGREE OF        SUM OF          MEAN          F          IMPROVEMENT IN TERMS
                               FREEDOM         SQUARES         SQUARE        VALUE        OF SUM OF SQUARES

DUE TO REGRESSION                 2        22235.39062     11117.69531    460.18506          1681.12891
DEVIATION ABOUT REGRESSION       12          289.91016        24.15918
     TOTAL                       14        22525.30078


LYNOMIAL REGRESSION OF DEGREE  3

INTERCEPT       18.49422

REGRESSION COEFFICIENTS
 -3.88107      0.97684     -0.01409

                    ANALYSIS OF VARIANCE FOR   3  DEGREE POLYNOMIAL

   SOURCE OF VARIATION        DEGREE OF        SUM OF          MEAN          F          IMPROVEMENT IN TERMS
                               FREEDOM         SQUARES         SQUARE        VALUE        OF SUM OF SQUARES

DUE TO REGRESSION                 3        22256.32812      7418.77344    303.40063            20.93750
DEVIATION ABOUT REGRESSION       11          268.97266        24.45206
     TOTAL                       14        22525.30078


OLYNOMIAL REGRESSION OF DEGREE  4

NO IMPROVEMENT


OLYNOMIAL REGRESSION.....SAMPLE

OLYNOMIAL REGRESSION OF DEGREE  3


                         TABLE OF RESIDUALS
BSERVATION NO.    X VALUE      Y VALUE      Y ESTIMATE      RESIDUAL

      1          1.00000      10.00000       15.57588       -5.57588
      2          2.00000      16.00000       14.52666        1.47334
      3          3.00000      20.00000       15.26197        4.73803
      4          4.00000      23.00000       17.69725        5.30275
      5          5.00000      25.00000       21.74792        3.25208
      6          6.00000      26.00000       27.32945       -1.32945
      7          7.00000      30.00000       34.35724       -4.35724
      8          8.00000      36.00000       42.74675       -6.74675
      9          9.00000      48.00000       52.41338       -4.41338
     10         10.00000      62.00000       63.27257       -1.27257
     11         11.00000      78.00000       75.23978        2.76022
     12         12.00000      94.00000       88.23039        5.76961
     13         13.00000     107.00000      102.15950        4.84010
     14         14.00000     118.00000      116.94366        1.05634
     15         15.00000     127.00000      132.49719       -5.49719
```

igure 43. Output listing (polynomial regression)



• Figure 43. Output listing (polynomial regression) (Continued)

Operating Instructions

The sample program for polynomial regression is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output. A scratch tape (data set 13) is used as intermediate storage when Y values and Y estimates are plotted.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 35 seconds.

```
C                                                                       PLRG  10
C     ...................................................................PLRG  20
C                                                                       PLRG  30
C        SAMPLE MAIN PROGRAM FOR POLYNOMIAL REGRESSION - POLRG          PLRG  40
C                                                                       PLRG  50
C        PURPOSE                                                        PLRG  60
C           (1) READ THE PROBLEM PARAMETER CARD FOR A POLYNOMIAL REGRES-PLRG  70
C           SION, (2) CALL SUBROUTINES TO PERFORM THE ANALYSIS, (3)     PLRG  80
C           PRINT THE REGRESSION COEFFICIENTS AND ANALYSIS OF VARIANCE  PLRG  90
C           TABLE FOR POLYNOMIALS OF SUCCESSIVELY INCREASING DEGREES,   PLRG 100
C           AND (4) OPTIONALLY PRINT THE TABLE OF RESIDUALS AND A PLOT  PLRG 110
C           OF Y VALUES AND Y ESTIMATES.                                PLRG 120
C                                                                       PLRG 130
C        REMARKS                                                        PLRG 140
C           THE NUMBER OF OBSERVATIONS, N, MUST BE GREATER THAN M+1,    PLRG 150
C           WHERE M IS THE HIGHEST DEGREE POLYNOMIAL SPECIFIED.         PLRG 160
C           IF THERE IS NO REDUCTION IN THE RESIDUAL SUM OF SQUARES     PLRG 170
C           BETWEEN TWO SUCCESSIVE DEGREES OF THE POLYNOMIALS, THE      PLRG 180
C           PROGRAM TERMINATES THE PROBLEM BEFORE COMPLETING THE ANALY- PLRG 190
C           SIS FOR THE HIGHEST DEGREE POLYNOMIAL SPECIFIED.            PLRG 200
C                                                                       PLRG 210
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PLRG 220
C           GDATA                                                       PLRG 230
C           ORDER                                                       PLRG 240
C           MINV                                                        PLRG 250
C           MULTR                                                       PLRG 260
C           PLOT  (A SPECIAL PLOT SUBROUTINE PROVIDED FOR THE SAMPLE    PLRG 270
C                 PROGRAM.)                                             PLRG 280
C                                                                       PLRG 290
C        METHOD                                                         PLRG 300
C           REFER TO B. OSTLE, 'STATISTICS IN RESEARCH', THE IOWA STATE PLRG 310
C           COLLEGE PRESS', 1954, CHAPTER 6.                            PLRG 320
C                                                                       PLRG 330
C     ...................................................................PLRG 340
```

```
THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE
PRODUCT OF N*(M+1), WHERE N IS THE NUMBER OF OBSERVATIONS AND M
IS THE HIGHEST DEGREE POLYNOMIAL SPECIFIED..

   DIMENSION X(1100)

THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE
PRODUCT OF M*M..

   DIMENSION DI(100)

THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO
(M+2)*(M+1)/2..

   DIMENSION D(66)

THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO M..

   DIMENSION B(10),E(10),SB(10),T(10)

THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO (M+1)..

   DIMENSION XBAR(11),STD(11),COE(11),SUMSQ(11),ISAVE(11)

THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO 10..

   DIMENSION ANS(10)

THE FOLLOWING DIMENSION WILL BE USED IF THE PLOT OF OBSERVED DATA
AND ESTIMATES IS DESIRED.  THE SIZE OF THE DIMENSION, IN THIS
CASE, MUST BE GREATER THAN OR EQUAL TO N*3.  OTHERWISE, THE SIZE
OF DIMENSION MAY BE SET TO 1.

   DIMENSION P(300)

   ...............................................................

      IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE
C     IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION
      STATEMENT WHICH FOLLOWS.

   DOUBLE PRECISION X,XBAR,STD,D,SUMSQ,DI,E,B,SB,T,ANS,DET,COE

      THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS
      APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS
      ROUTINE.

      ...............................................................

 1 FORMAT(A4,A2,I5,I2,I1)
 2 FORMAT(2F6.0)
 3 FORMAT(27H1POLYNOMIAL REGRESSION......,A4,A2/)
 4 FORMAT(23HONUMBER OF OBSERVATIONS,I6//)
 5 FORMAT(32HOPOLYNOMIAL REGRESSION OF DEGREE,I3)
 6 FORMAT(12HO  INTERCEPT,E20.7)
 7 FORMAT(26HO  REGRESSION COEFFICIENTS/(6E20.7))
 8 FORMAT(1HO/24X,24HANALYSIS OF VARIANCE FOR,I4,19H  DEGREE POLYNOMI
  1AL/)
 9 FORMAT(1HO,5X,19HSOURCE OF VARIATION,7X,9HDEGREE OF,7X,6HSUM OF,9X
  1,4HMEAN,10X,1HF,9X,20HIMPROVEMENT IN TERMS/33X,7HFREEDOM,8X,7HSQUA
  2RES,7X,6HSQUARE,7X,5HVALUE,8X,17HOF SUM OF SQUARES)
10 FORMAT(20HO  DUE TO REGRESSION,12X,I6,F17.5,F14.5,F13.5,F20.5)
11 FORMAT(32H   DEVIATION ABOUT REGRESSION   ,I6,F17.5,F14.5)
12 FORMAT(8X,5HTOTAL,19X,I6,F17.5///)
13 FORMAT(17HO  NO IMPROVEMENT)
14 FORMAT(1HO//27X,18HTABLE OF RESIDUALS//16H OBSERVATION NO.,5X,7HX
  1VALUE,7X,7HY VALUE,7X,10HY ESTIMATE,7X,8HRESIDUAL/)
15 FORMAT(1HO,3X,I6,F18.5,F14.5,F17.5,F15.5)

   ...............................................................

   READ PROBLEM PARAMETER CARD

100 READ (5,1) PR,PR1,N,M,NPLOT

      PR....PROBLEM NUMBER (MAY BE ALPHAMERIC)
      PR1...PROBLEM NUMBER (CONTINUED)
      N.....NUMBER OF OBSERVATIONS
      M.....HIGHEST DEGREE POLYNOMIAL SPECIFIED
      NPLOT.OPTION CODE FOR PLOTTING
            0  IF PLOT IS NOT DESIRED.
            1  IF PLOT IS DESIRED.

   PRINT PROBLEM NUMBER AND N.

   WRITE (6,3) PR,PR1
   WRITE (6,4) N

   READ INPUT DATA

   L=N*M
   DO 110 I=1,N
   J=L+I
C
C        X(I) IS THE INDEPENDENT VARIABLE, AND X(J) IS THE DEPENDENT
C        VARIABLE.
C
 110 READ (5,2) X(I),X(J)
C
      CALL GDATA (N,M,X,XBAR,STD,D,SUMSQ)
C
      MM=M+1
      SUM=0.0
      NT=N-1
C
      DO 200 I=1,M
      ISAVE(I)=I
C
C     FORM SUBSET OF CORRELATION COEFFICIENT MATRIX
C
      CALL ORDER (MM,D,MM,I,ISAVE,DI,E)
C
C     INVERT THE SUBMATRIX OF CORRELATION COEFFICIENTS
C
      CALL MINV (DI,I,DET,B,T)
C
      CALL MULTR (N,I,XBAR,STD,SUMSQ,DI,E,ISAVE,B,SB,T,ANS)
C
C     PRINT THE RESULT OF CALCULATION
C
      WRITE (6,5) I
      IF(ANS(7)) 140,130,130
 130  SUMIP=ANS(4)-SUM.
      IF(SUMIP) 140, 140, 150
 140  WRITE (6,13)
      GO TO 210
 150  WRITE (6,6) ANS(1)
      WRITE (6,7) (B(J),J=1,I)
      WRITE (6,8) I
      WRITE (6,9)
      SUM=ANS(4)
      WRITE (6,10) I,ANS(4),ANS(6),ANS(10),SUMIP
      NI=ANS(8)
      WRITE (6,11) NI,ANS(7),ANS(9)
      WRITE (6,12) NT,SUMSQ(MM)
C
C     SAVE COEFFICIENTS FOR CALCULATION OF Y ESTIMATES
C
      COE(1)=ANS(1)
      DO 160 J=1,I
 160  COE(J+1)=B(J)
      LA=I
 200  CONTINUE
C
C     TEST WHETHER PLOT IS DESIRED
C
 210  IF(NPLOT) 100, 100, 220
C
C        CALCULATE ESTIMATES
C
 220  NP3=N+N
      DO 230 I=1,N
      NP3=NP3+1
      P(NP3)=COE(1)
      L=I
      DO 230 J=1,LA
      P(NP3)=P(NP3)+X(L)*COE(J+1)
 230  L=L+N
C
C        COPY OBSERVED DATA
C
      N2=N
      L=N*M
      DO 240 I=1,N
      P(I)=X(I)
      N2=N2+1
      L=L+1
 240  P(N2)=X(L)
C
C     PRINT TABLE OF RESIDUALS
C
      WRITE (6,3) PR,PR1
      WRITE (6,5) LA
      WRITE (6,14)
      NP2=N
      NP3=N+N
      DO 250 I=1,N
      NP2=NP2+1
      NP3=NP3+1
      RESID=P(NP2)-P(NP3)
 250  WRITE (6,15) I,P(I),P(NP2),P(NP3),RESID
C
      CALL PLOT (LA,P,N,3,0,1)
C
      GO TO 100
      END
```

STEPWISE MULTIPLE REGRESSION

Problem Description

Stepwise multiple regression is a statistical technique for analyzing a relationship between a dependent variable (y) and a set of independent variables ($x_1, x_2, \ldots, x_m$) and for selecting the independent variables in the order of their importance. The criterion of importance is based on the reduction of sums of squares, and the independent variable most important in this reduction in a given step is entered in the regression. Any variable in the original set can be named as the dependent variable. A set of variables can be forced to enter in the regression, and another set of variables can be deleted.

The sample problem for stepwise multiple regression consists of 30 observations on six variables as presented in Table 4. The first five variables are independent variables (predictors), and the last variable is the dependent variable (criterion). All five independent variables are available for selection in the first analysis. In the second analysis, the third variable is forced to enter in the regression, the fourth variable is deleted, and the remaining three independent variables are available for selection. Note that the same sample problem has been used for multiple linear regression and that the results of these two programs may be compared.

Program

Description

The stepwise multiple regression program consists of the main routine named STEPR, a special input subroutine named DATA, a special output subroutine named STOUT, and four subroutines from the Scientific Subroutine Package: CORRE, MSTR, LOC, and STPRG.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to 35 variables, including both dependent and independent variables
2. Up to 99,999 observations, if observations are read into the computer one at a time by the special input subroutine named DATA. If all data are to be stored in core prior to the calculation of correlation coefficients, the limitation on the number of observations depends on the size of core storage available for input data.
3. (12F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions, it is not necessary to modify the sample program. However, if there are more than 35 variables, dimension statements in the sample main program must be modified to handle this particular problem. Similarly, if input data cards are prepared using a different format, the format in the input subroutine, DATA, must be modified. The general rules for program modification are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, STEPR. This card is prepared as follows:

Columns	Contents	For Sample Problem
1-6	Problem code (may be alphameric)	SAMPLE
7-11	Number of observations	00030
12-13	Number of variables	06
14-15	Number of selection cards (see below)	02
16-21	This constant is the proportion of the sum of squares which will be used to limit the entering of variables into the regression. Unless the proportion of the sum of squares attributable to an entering variable is greater than this value, the variable will not be entered. The keypunched value should have a decimal point.*	0.0
22	Option code for table of residuals 0--if it is not desired 1--if it is desired	 1

*The choice of value depends upon the purpose of analysis. A suggested trial value is 0.01 (one percent). If all variables are to be included in the regression, use 0.0.

Leading zeros are not required to be keypunched.

Data Cards

Since input data can be read into the computer one observation at a time, each row of data in Table 4 is keypunched on a separate card using the format (12F6.0). This format assumes twelve 6-column fields per card.

If there are more than twelve variables in a problem, each row of data is continued on the second and third cards until the last data point is keypunched. However, each row of data must begin on a new card.

Selection Card

The selection card is used to specify a dependent variable, independent variables to be forced to enter in the regression, variables to be deleted, and independent variables to be chosen by the program to enter in the regression. Any one variable in the set of original variables can be designated as a dependent variable, and any number of variables can be specified as forcing variables, deleting variables, and unrestricted variables to be chosen by the program. The specification of the variables can be performed over and over again using the same set of original variables.

In a selection card, each variable is specified using one of the following codes:

Code	Specification
0 or blank	Independent variable to be chosen by the program to enter in the regression
1	Independent variable forced by the user to enter in the regression
2	Variable to be deleted
3	Dependent variable

The selection card itself is prepared as follows:

Columns	Contents	For Sample Problem Selections 1	2
1	Code for 1st variable	0	0
2	Code for 2nd variable	0	0
3	Code for 3rd variable	0	1
4	Code for 4th variable	0	2
5	Code for 5th variable	0	0
6	Code for 6th variable	3	3
etc.			

The input format of (72I1) is used for the selection card.

Deck Setup

The deck setup is shown in Figure 44.

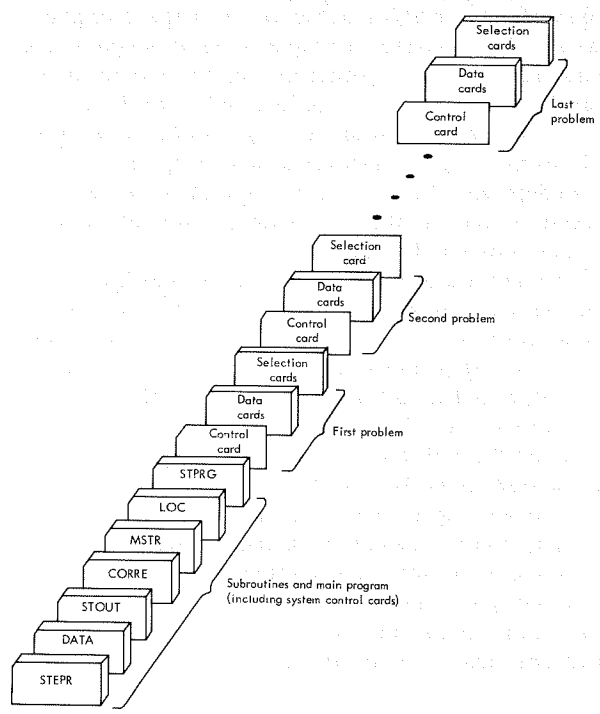


Figure 44. Deck setup (stepwise multiple regression)

Sample

The listing of input cards for the sample problem is presented in Figure 45.

```
/DATA
SAMPLE   30 6 2  0.001
    29   289   216    85    14     1
    30   391   244    92    16     2
    30   424   246    90    18     2
    30   313   239    91    10     0
    35   243   275    95    30     2
    35   365   219    95    21     2
    43   396   267   100    39     3
    43   356   274    79    19     2
    44   346   255   126    56     3
    44   156   258    95    28     0
    44   278   249   110    42     4
    44   349   252    88    21     1
    44   141   236   129    56     1
    44   245   236    97    24     1
    45   297   256   111    45     3
    45   310   262    94    20     2
    45   151   339    96    35     3
    45   370   357    88    15     4
    45   379   198   147    64     4
    45   463   206   105    31     3
    45   316   245   132    60     4
    45   280   225   108    36     4
    44   395   215   101    27     1
    49   139   220   136    59     0
    49   245   205   113    37     4
    49   373   215    88    25     1
    51   224   215   118    54     3
    51   677   210   116    33     4
    51   424   210   140    59     4
    51   150   210   105    30     0
000003
001203
```

Figure 45.

Output

Description

The output of the sample program for stepwise multiple regression includes:

1. For all variables --
   a. Means
   b. Standard deviations
   c. Correlation coefficient matrix
2. For each step of multiple regression --
   a. Sum of squares reduced
   b. Proportion reduced
   c. Cumulative sum of squares reduced
   d. Cumulative proportion reduced
   e. Multiple correlation coefficient (adjusted and unadj.)
   f. F-value for analysis of variance
   g. Standard error of estimate (adjusted and unadj.)
   h. Regression coefficients
   i. Standard errors of regression coefficients
   j. Computed t-values
3. Tables of residuals (optional)

Sample

The output listing for the sample problem is shown in Figure 46.

```
STEP-WISE MULTIPLE REGRESSION.....SAMPLE

NUMBER OF OBSERVATIONS   30
NUMBER OF VARIABLES       6
NUMBER OF SELECTIONS      2

CONSTANT TO LIMIT VARIABLES  0.0

VARIABLE      MEAN      STANDARD
  NO.                   DEVIATION
   1       43.13333      6.52176
   2      316.16650    114.42990
   3      241.79999     36.43074
   4      105.66666     17.85640
   5       34.13333     15.97571
   6        2.26667      1.41259

CORRELATION MATRIX

ROW  1
    1.00000    -0.06721    -0.13689     0.49755     0.55849     0.28422

ROW  2
   -0.06721     1.00000    -0.17857    -0.05227    -0.18381     0.42189

ROW  3
   -0.13689    -0.17857     1.00000    -0.40874    -0.26319     0.11900

ROW  4
    0.49755    -0.05227    -0.40874     1.00000     0.93552     0.37822

ROW  5
    0.55849    -0.18381    -0.26319     0.93552     1.00000     0.39412

ROW  6
    0.28422     0.42189     0.11900     0.37822     0.39412     1.00000


SELECTION..... 1

DEPENDENT VARIABLE............ 6
NUMBER OF VARIABLES FORCED.... 0
NUMBER OF VARIABLES DELETED... 0


STEP  1

VARIABLE ENTERED..... 2

SUM OF SQUARES REDUCED IN THIS STEP....        10.300
PROPORTION REDUCED IN THIS STEP........         0.178

CUMULATIVE SUM OF SQUARES REDUCED......        10.300
CUMULATIVE PROPORTION REDUCED..........         0.178  OF        57.867
```

Figure 46.

```
FOR  1 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.422
      (ADJUSTED FOR D.F.)............     0.422
  F-VALUE FOR ANALYSIS OF VARIANCE...     6.063
  STANDARD ERROR OF ESTIMATE.........     1.303
      (ADJUSTED FOR D.F.)............     1.303

  VARIABLE      REGRESSION     STD. ERROR OF     COMPUTED
   NUMBER       COEFFICIENT     REG. COEFF.      T-VALUE
     2            0.00521        0.00212         2.462
  INTERCEPT       0.62005


STEP  2

VARIABLE ENTERED..... 5

SUM OF SQUARES REDUCED IN THIS STEP....        13.324
PROPORTION REDUCED IN THIS STEP........         0.230

CUMULATIVE SUM OF SQUARES REDUCED......        23.624
CUMULATIVE PROPORTION REDUCED..........         0.408  OF        57.867

FOR  2 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.639
      (ADJUSTED FOR D.F.)............     0.622
  F-VALUE FOR ANALYSIS OF VARIANCE...     9.314
  STANDARD ERROR OF ESTIMATE.........     1.126
      (ADJUSTED FOR D.F.)............     1.146

  VARIABLE      REGRESSION     STD. ERROR OF     COMPUTED
   NUMBER       COEFFICIENT     REG. COEFF.      T-VALUE
     2            0.00632        0.00186         3.397
     5            0.04316        0.01332         3.241
  INTERCEPT      -1.20349


STEP  3

VARIABLE ENTERED..... 3

SUM OF SQUARES REDUCED IN THIS STEP....         7.572
PROPORTION REDUCED IN THIS STEP........         0.131

CUMULATIVE SUM OF SQUARES REDUCED......        31.196
CUMULATIVE PROPORTION REDUCED..........         0.539  OF        57.867

FOR  3 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.734
      (ADJUSTED FOR D.F.)............     0.711
  F-VALUE FOR ANALYSIS OF VARIANCE...    10.137
  STANDARD ERROR OF ESTIMATE.........     1.013
      (ADJUSTED FOR D.F.)............     1.050

  VARIABLE      REGRESSION     STD. ERROR OF     COMPUTED
   NUMBER       COEFFICIENT     REG. COEFF.      T-VALUE
     2            0.00744        0.00172         4.318
     5            0.05363        0.01258         4.263
     3            0.01497        0.00551         2.717
  INTERCEPT      -5.53529


STEP  4

VARIABLE ENTERED..... 1

SUM OF SQUARES REDUCED IN THIS STEP....         0.127
PROPORTION REDUCED IN THIS STEP........         0.002

CUMULATIVE SUM OF SQUARES REDUCED......        31.323
CUMULATIVE PROPORTION REDUCED..........         0.541  OF        57.867

FOR  4 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.736
      (ADJUSTED FOR D.F.)............     0.699
  F-VALUE FOR ANALYSIS OF VARIANCE...     7.375
  STANDARD ERROR OF ESTIMATE.........     1.030
      (ADJUSTED FOR D.F.)............     1.088

  VARIABLE      REGRESSION     STD. ERROR OF     COMPUTED
   NUMBER       COEFFICIENT     REG. COEFF.      T-VALUE
     2            0.00741        0.00175         4.222
     5            0.05076        0.01524         3.332
     3            0.01493        0.00561         2.662
     1            0.01226        0.03541         0.346
  INTERCEPT      -5.94617


STEP  5

VARIABLE ENTERED..... 4

SUM OF SQUARES REDUCED IN THIS STEP....         0.002
PROPORTION REDUCED IN THIS STEP........         0.000

CUMULATIVE SUM OF SQUARES REDUCED......        31.325
CUMULATIVE PROPORTION REDUCED..........         0.541  OF        57.867

FOR  5 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.736
      (ADJUSTED FOR D.F.)............     0.684
  F-VALUE FOR ANALYSIS OF VARIANCE...     5.665
  STANDARD ERROR OF ESTIMATE.........     1.052
      (ADJUSTED FOR D.F.)............     1.133

  VARIABLE      REGRESSION     STD. ERROR OF     COMPUTED
   NUMBER       COEFFICIENT     REG. COEFF.      T-VALUE
     2            0.00739        0.00186         3.965
     5            0.04919        0.04141         1.188
     3            0.01504        0.00635         2.369
     1            0.01242        0.03635         0.342
     4            0.00151        0.03679         0.041
  INTERCEPT      -6.07929
```

Figure 46. (Continued)

```
SELECTION..... 1

                TABLE OF RESIDUALS

CASE NO.      Y VALUE      Y ESTIMATE      RESIDUAL
     1        1.00000        0.48090        0.51910
     2        2.00000        1.77670        0.22330
     3        2.00000        2.14586       -0.14586
     4        0.0            0.82880       -0.82880
     5        2.00000        1.90522        0.09478
     6        2.00000        1.52125        0.47875
     7        3.00000        3.46447       -0.46447
     8        2.00000        2.25887       -0.25887
     9        3.00000        3.80259       -0.80259
    10        0.0            1.02042       -1.02042
    11        4.00000        2.49735        1.50265
    12        1.00000        2.00065       -1.00065
    13        1.00000        2.00736       -1.00736
    14        1.00000        1.15308       -0.15308
    15        3.00000        2.90446        0.09554
    16        2.00000        1.83531        0.16469
    17        3.00000        2.56004        0.43996
    18        4.00000        3.45228        0.54772
    19        4.00000        3.62661        0.37339
    20        3.00000        2.68068        0.31932
    21        4.00000        3.64886        0.35114
    22        4.00000        1.86541        2.13459
    23        1.00000        2.09863       -1.09863
    24        0.0            1.97217       -1.97217
    25        4.00000        1.41254        2.58746
    26        1.00000        1.88027       -0.88027
    27        3.00000        2.27646        0.72354
    28        4.00000        4.51080       -0.51080
    29        4.00000        3.95746        0.04254
    30        0.0            0.45458       -0.45458


SELECTION..... 2

DEPENDENT VARIABLE............ 6
NUMBER OF VARIABLES FORCED.... 1
NUMBER OF VARIABLES DELETED... 1


STEP  1

VARIABLE ENTERED..... 3
  (FORCED VARIABLE)

SUM OF SQUARES REDUCED IN THIS STEP....          0.819
PROPORTION REDUCED IN THIS STEP........          0.014

CUMULATIVE SUM OF SQUARES REDUCED......          0.819
CUMULATIVE PROPORTION REDUCED..........          0.014  OF        57.867

FOR  1 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.119
      (ADJUSTED FOR D.F.)............     0.119
  F-VALUE FOR ANALYSIS OF VARIANCE...     0.402
  STANDARD ERROR OF ESTIMATE.........     1.427
      (ADJUSTED FOR D.F.)............     1.427

  VARIABLE      REGRESSION      STD. ERROR OF      COMPUTED
   NUMBER      COEFFICIENT       REG. COEFF.        T-VALUE
     3           0.00461          0.00728           0.634
  INTERCEPT      1.15093


STEP  2

VARIABLE ENTERED..... 2

SUM OF SQUARES REDUCED IN THIS STEP....         11.738
PROPORTION REDUCED IN THIS STEP........          0.203

CUMULATIVE SUM OF SQUARES REDUCED......         12.557
CUMULATIVE PROPORTION REDUCED..........          0.217  OF        57.867

FOR  2 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.466
      (ADJUSTED FOR D.F.)............     0.435
  F-VALUE FOR ANALYSIS OF VARIANCE...     3.741
  STANDARD ERROR OF ESTIMATE.........     1.295
      (ADJUSTED FOR D.F.)............     1.318

  VARIABLE      REGRESSION      STD. ERROR OF      COMPUTED
   NUMBER      COEFFICIENT       REG. COEFF.        T-VALUE
     3           0.00778          0.00671           1.160
     2           0.00565          0.00214           2.645
  INTERCEPT     -1.40194


STEP  3

VARIABLE ENTERED..... 5

SUM OF SQUARES REDUCED IN THIS STEP....         18.639
PROPORTION REDUCED IN THIS STEP........          0.322

CUMULATIVE SUM OF SQUARES REDUCED......         31.196
CUMULATIVE PROPORTION REDUCED..........          0.539  OF        57.867

FOR  3 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.734
      (ADJUSTED FOR D.F.)............     0.711
  F-VALUE FOR ANALYSIS OF VARIANCE...    10.137
  STANDARD ERROR OF ESTIMATE.........     1.013
      (ADJUSTED FOR D.F.)............     1.050

  VARIABLE      REGRESSION      STD. ERROR OF      COMPUTED
   NUMBER      COEFFICIENT       REG. COEFF.        T-VALUE
     3           0.01497          0.00551           2.717
     2           0.00744          0.00172           4.318
     5           0.05363          0.01258           4.263
  INTERCEPT     -5.53528
```

Figure 46. (Continued)

```
STEP  4

VARIABLE ENTERED..... 1

SUM OF SQUARES REDUCED IN THIS STEP....          0.127
PROPORTION REDUCED IN THIS STEP........          0.002

CUMULATIVE SUM OF SQUARES REDUCED......         31.323
CUMULATIVE PROPORTION REDUCED..........          0.541  OF        57.867

FOR  4 VARIABLES ENTERED
  MULTIPLE CORRELATION COEFFICIENT...     0.736
      (ADJUSTED FOR D.F.)............     0.699
  F-VALUE FOR ANALYSIS OF VARIANCE...     7.375
  STANDARD ERROR OF ESTIMATE.........     1.030
      (ADJUSTED FOR D.F.)............     1.088

  VARIABLE      REGRESSION      STD. ERROR OF      COMPUTED
   NUMBER      COEFFICIENT       REG. COEFF.        T-VALUE
     3           0.01493          0.00561           2.662
     2           0.00741          0.00175           4.222
     5           0.05076          0.01524           3.332
     1           0.01226          0.03541           0.346
  INTERCEPT     -5.94617


SELECTION..... 2

                TABLE OF RESIDUALS

CASE NO.      Y VALUE      Y ESTIMATE      RESIDUAL
     1        1.00000        0.48498        0.51502
     2        2.00000        1.77220        0.22780
     3        2.00000        2.14799       -0.14799
     4        0.0            0.81528       -0.81528
     5        2.00000        1.91082        0.08918
     6        2.00000        1.52156        0.47844
     7        3.00000        3.47954       -0.47954
     8        2.00000        2.27251       -0.27251
     9        3.00000        3.80533       -0.80533
    10        0.0            1.02152       -1.02152
    11        4.00000        2.50144        1.49856
    12        1.00000        2.00604       -1.00604
    13        1.00000        2.00340       -1.00340
    14        1.00000        1.14922       -0.14922
    15        3.00000        2.91121        0.08879
    16        2.00000        1.82797        0.17203
    17        3.00000        2.56125        0.43875
    18        4.00000        3.43668        0.56332
    19        4.00000        3.61723        0.38277
    20        3.00000        2.68358        0.31642
    21        4.00000        3.64918        0.35082
    22        4.00000        1.86566        2.13434
    23        1.00000        2.09898       -1.09898
    24        0.0            1.96334       -1.96334
    25        4.00000        1.40769        2.59231
    26        1.00000        1.89582       -0.89582
    27        3.00000        2.28895        0.71105
    28        4.00000        4.50336       -0.50336
    29        4.00000        3.94940        0.05060
    30        0.0            0.44790       -0.44790
```

Figure 46. (Continued)

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a different format statement. If the user encounters problems with large or small numbers in his output, he may wish to change output format statements from F to E notation. In order to familiarize the user with the program modification, the following general rules are supplied in terms of the sample problem.

1. Changes in the dimension statements of the main program, STEPR:

a. The dimension of arrays XBAR, STD, D, B, T, IDX, and L must each be greater than or equal to the number of variables, m. Since there are six variables in the sample problem, the value of m is 6.

b. The dimension of array RX must be greater than or equal to the product of m x m. For the sample problem this product is 36 = 6 x 6.

c. The dimension of array R must be greater than or equal to (m+1)m/2. For the sample problem this number is 21 = (6+1)6/2.

2. Changes in the input format statement of the special input subroutine, DATA:

Only the format statement for input data may be changed. If sample data are either one-, two-, or three-digit numbers, each row of data may be keypunched in six 3-column fields rather than using six-column fields as in the sample problem; in such case the format is changed to (6F3.0).

The special input subroutine, DATA, is normally written by the user to handle different formats for different problems. The user may modify this subroutine to perform testing of input data, transformation of data, and so on.

Operating Instructions

The sample program for stepwise multiple regression is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output. A scratch tape (data set 13) written in binary mode is used as intermediate storage.

Error Messages

The following error conditions will result in messages:

1. The number of selection cards is not specified on the control card--NUMBER OF SELECTIONS NOT SPECIFIED. JOB TERMINATED. This error terminates execution of the job.

2. More than one dependent variable is specified on a selection card--SELECTION CARD (N) DOES NOT NAME ONE AND ONLY ONE DEPENDENT VARIABLE. SELECTION IGNORED.

3. A control digit is not 0, 1, 2, or 3 on a selection card--COLUMN (K) OF SELECTION CARD (N) IS IN ERROR. IT IS POSSIBLE THAT COLUMNS SUCCEEDING THAT COLUMN ARE ALSO INCORRECT. THE SELECTION IS IGNORED.

4. There are too few observations for estimation to take place--(N) OBSERVATIONS ARE TOO FEW TO ALLOW PARAMETER ESTIMATION FOR (M) VARIABLES. JOB TERMINATED.

5. The correlation matrix is singular or badly conditioned--EITHER THE MATRIX IS SINGULAR, OR THE RESIDUAL SUM OF SQUARES IS NEGATIVE IMPLYING EXTREME ILL-CONDITION. SELECTION IGNORED.

Two other error types are not detected:

1. If the number of observations is incorrect, the job deck will be such that results on that selection and on succeeding jobs will be incorrect. The user should take care in preparing his job deck according to instructions given above.

2. If some variable is constant over all observations, a zero standard deviation will occur, and results for that selection will be incorrect or not forthcoming.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 45 seconds.

```
C                                                                       STEP  10
C     ..................................................................STEP  20
C                                                                       STEP  30
C        SAMPLE MAIN PROGRAM FOR STEP-WISE MULTIPLE REGRESSION - STEPR  STEP  40
C                                                                       STEP  50
C        PURPOSE                                                        STEP  60
C           (1) READ THE PROBLEM PARAMETER CARD FOR A STEP-WISE MULTIPLESTEP  70
C           REGRESSION. (2) READ SUBSET SELECTION CARDS, (3) CALL THE   STEP  80
C           SUBROUTINE TO CALCULATE MEANS, STANDARD DEVIATIONS, SIMPLE  STEP  90
C           CORRELATION COEFFICIENTS, AND (4) CALL THE SUBROUTINE TO    STEP 100
C           PERFORM EACH STEP OF REGRESSION ANALYSIS.                   STEP 110
C                                                                       STEP 120
C                                                                       STEP 130
C        REMARKS                                                        STEP 140
C           THE NUMBER OF OBSERVATIONS, N, MUST BE GREATER THAN M+2,    STEP 150
C           WHERE M IS THE NUMBER OF VARIABLES.  IF SELECTION CARDS ARE STEP 160
C           NOT PRESENT, THIS PROGRAM CAN NOT PERFORM STEP-WISE MULTIPLESTEP 170
C           REGRESSION.                                                 STEP 180
C                                                                       STEP 190
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  STEP 200
C           CORRE (WHICH, IN TURN, CALLS THE SUBROUTINE DATA)           STEP 210
C           MSTR  (WHICH, IN TURN, CALLS THE SUBROUTINE LOC)            STEP 220
C           STPRG (WHICH, IN TURN, CALLS THE SUBROUTINE STOUT)          STEP 230
C                                                                       STEP 240
C        METHOD                                                         STEP 250
C           REFER TO C. A. BENNETT AND N. L. FRANKLIN, 'STATISTICAL     STEP 260
C           ANALYSIS IN CHEMISTRY AND THE CHEMICAL INDUSTRY', JOHN WILEYSTEP 270
C           AND SONS, 1954, APPENDIX 6A.                                STEP 280
C     ..................................................................STEP 290
C                                                                       STEP 300
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO THE     STEP 310
C     NUMBER OF VARIABLES, M..                                          STEP 320
C                                                                       STEP 330
      DIMENSION XBAR(35),STD(35),D(35),B(35),T(35),IDX(35),L(35)        STEP 340
C                                                                       STEP 350
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      STEP 360
C     PRODUCT OF M*M..                                                  STEP 370
C                                                                       STEP 380
      DIMENSION RX(1225)                                                STEP 390
C                                                                       STEP 400
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      STEP 410
C     (M+1)*M/2..                                                       STEP 420
C                                                                       STEP 430
      DIMENSION R(630)                                                  STEP 440
C                                                                       STEP 450
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO 5..      STEP 460
C                                                                       STEP 470
      DIMENSION NSTEP(5)                                                STEP 480
C                                                                       STEP 490
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO 11..     STEP 500
C                                                                       STEP 510
      DIMENSION ANS(11)                                                 STEP 520
C                                                                       STEP 530
C     ..................................................................STEP 540
C                                                                       STEP 550
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  STEP 560
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      STEP 570
C        STATEMENT WHICH FOLLOWS.                                       STEP 580
C                                                                       STEP 590
C     DOUBLE PRECISION XBAR,STD,RX,R,B,T,ANS,YEST                       STEP 600
C                                                                       STEP 610
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    STEP 620
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      STEP 630
C        ROUTINE.                                                       STEP 640
C                                                                       STEP 650
C     ..................................................................STEP 660
C                                                                       STEP 670
    1 FORMAT(A4,A2,I5,2I2,F6.0,I1)                                      STEP 680
    2 FORMAT(53HONUMBER OF SELECTIONS NOT SPECIFIED.  JOB TERMINATED.)  STEP 690
    3 FORMAT(35H1STEP-WISE MULTIPLE REGRESSION.....A4,A2)               STEP 700
    4 FORMAT(31H0VARIABLE      MEAN      STANDARD/4X,3HN0.16X,9HDEVIATION)STEP 710
    5 FORMAT(4X,I2,F14.5,F12.5)                                         STEP 720
    6 FORMAT(19H1CORRELATION MATRIX)                                    STEP 730
    7 FORMAT(4H0ROW I3/(10F12.5))                                       STEP 740
    8 FORMAT(72I1)                                                      STEP 750
    9 FORMAT(23H0NUMBER OF OBSERVATIONSI5)                              STEP 760
   10 FORMAT(20H NUMBER OF VARIABLES3X,I5)                              STEP 770
   11 FORMAT(21H NUMBER OF SELECTIONS2X,I5)                             STEP 780
   12 FORMAT(28H0CONSTANT TO LIMIT VARIABLESF9.5)                       STEP 790
   13 FORMAT(/15H1SELECTION.....I2)                                     STEP 800
   14 FORMAT(16X,18HTABLE OF RESIDUALS//9H CASE NO.5X,7HY VALUE5X,10HY ESTEP 810
     1STIMATE6X,8HRESIDUAL)                                             STEP 820
   15 FORMAT(I7,F15.5,2F14.5)                                           STEP 830
   16 FORMAT(1H )                                                       STEP 840
   17 FORMAT(1H1)                                                       STEP 850
   18 FORMAT(1H0,'****COLUMN',I4,' OF SELECTION CARD',I5,' IS IN ERROR. STEP 860
     1 IT IS POSSIBLE THAT COLUMNS SUCCEEDING THAT COLUMN ARE ALSO'     STEP 870
     2/' INCORRECT.  THE SELECTION IS IGNORED.****')                    STEP 880
   19 FORMAT(1H0,'****SELECTION CARD',I5,' DOES NOT NAME ONE AND ONLY ONSTEP 890
     1E DEPENDENT VARIABLE.  SELECTION IGNORED.****')                   STEP 900
   20 FORMAT(1H0,'****EITHER THE MATRIX IS SINGULAR, OR THE RESIDUAL SUMSTEP 910
     1 OF SQUARES IS NEGATIVE IMPLYING EXTREME ILL CONDITION.',/,' SELECSTEP 920
     2TION IGNORED.****')                                               STEP 930
   21 FORMAT(1H0,'****',I6,' OBSERVATIONS ARE TOO FEW TO ALLOW PARAMETERSTEP 940
     1 ESTIMATION FOR',I5,' VARIABLES.  JOB TERMINATED.****')           STEP 950
C                                                                       STEP 960
C        READ PROBLEM PARAMETER CARD                                    STEP 970
C                                                                       STEP 980
  100 READ (5,1) PR1,PR2,N,M,NS,PCT,NR                                  STEP 990
C        PR1.....PROBLEM CODE (MAY BE ALPHAMERIC)                       STEP1000
C        PR2.....PROBLEM CODE (CONTINUED)                               STEP1010
C        N .......NUMBER OF OBSERVATIONS                                STEP1020
C        M .......NUMBER OF VARIABLES                                   STEP1030
C        NS......NUMBER OF SELECTIONS                                   STEP1040
C        PCT.....A CONSTANT VALUE OF PROPORTION OF SUM OF SQUARES THAT  STEP1050
C                WILL BE USED TO LIMIT VARIABLES ENTERING IN THE REGRES-STEP1060
C                SION                                                   STEP1070
C        NR......OPTION CODE FOR TABLE OF RESIDUALS                     STEP1080
```

```
C                 0 - IF IT IS NOT DESIRED
C                 1 - IF IT IS DESIRED
C
      WRITE (6,3) PR1,PR2
      WRITE (6,9) N
      WRITE (6,10) M
      IF(N-M-2) 101,101,102
  101 WRITE(6,21) N,M
      STOP
  102 WRITE (6,11) NS
      WRITE (6,12) PCT
C
C     LOGICAL TAPE 13 IS USED AS INTERMEDIATE STORAGE TO HOLD INPUT
C     DATA.  THE INPUT DATA ARE WRITTEN ON LOGICAL TAPE 13 BY THE
C     SPECIAL INPUT SUBROUTINE NAMED DATA.  THE STORED DATA MAY BE USED
C     FOR RESIDUAL ANALYSIS.
C
      REWIND 13
C
      IO=0
      X=0.0
C
      CALL CORRE (N,M,IO,X,XBAR,STD,RX,R,B,D,T)
C
      REWIND 13
C
C        PRINT MEANS AND STANDARD DEVIATION
C
      WRITE (6,4)
      DO 105 I=1,M
  105 WRITE (6,5) I,XBAR(I),STD(I)
C
C        PRINT CORRELATION MATRIX
C
      WRITE (6,6)
      DO 130 I=1,M
      DO 125 J=1,M
      IF(I-J) 110, 120, 120
  110 K=I+(J*J-J)/2
      GO TO 125
  120 K=J+(I*I-I)/2
  125 T(J)=R(K)
  130 WRITE (6,7) I,(T(J),J=1,M)
C
C        TEST NUMBER OF SELECTIONS
C
      IF(NS) 135, 135, 140
  135 WRITE (6,2)
      GO TO 200
C
C        SAVE THE MATRIX OF SUMS OF CROSS-PRODUCTS OF DEVIATIONS
C
  140 CALL MSTR (RX,R,M,0,1)
C
      NSEL=1
      GO TO 150
C
C        COPY THE MATRIX OF SUMS OF CROSS-PRODUCTS OF DEVIATIONS
C
  145 CALL MSTR (R,RX,M,1,0)
C
C        READ A SELECTION CARD
C
  150 WRITE (6,13) NSEL
      READ (5,8) (IDX(J),J=1,M)
C
C        IN EACH POSITION OF IDX, ONE OF THE FOLLOWING CODES MUST BE
C        SPECIFIED..
C          0 OR BLANK - INDEPENDENT VARIABLE AVAILABLE FOR SELECTION
C          1          - INDEPENDENT VARIABLE TO BE FORCED IN REGRESSION
C          2          - VARIABLE TO BE DELETED
C          3          - DEPENDENT VARIABLE
C
      N35=0
      DO 155 K=1,M
      IF (IDX(K)) 152,153,153
  152 WRITE (6,18) K,NSEL
      GO TO 185
  153 IF (IDX(K)-3) 155,154,152
  154 N35=N35+1
  155 CONTINUE
      IF (N35-1) 156,157,156
  156 WRITE (6,19) NSEL
      GO TO 185
C        CALL THE SUBROUTINE TO PERFORM A STEP-WISE REGRESSION ANALYSIS
C
  157 CALL STPRG (M,N,RX,XBAR,IDX,PCT,NSTEP,ANS,L,B,STD,T,D,IER)
      IF (IER) 158,159,158
  158 WRITE (6,20)
      GO TO 185
C
C        FIND WHETHER TO PRINT THE TABLE OF RESIDUALS
C
  159 IF(NR) 185, 185, 160
C
C        PRINT THE TABLE OF RESIDUALS
C
C
  160 WRITE (6,13) NSEL
      WRITE (6,16)
      WRITE (6,14)
      MM=NSTEP(1)
      DO 180 I=1,N
      READ (13) (D(J),J=1,M)
      YEST=ANS(9)
      K=NSTEP(4)
      DO 170 J=1,K
      KK=L(J)
  170 YEST=YEST+B(J)*D(KK)
      RESI=D(MM)-YEST
  180 WRITE (6,15) I,D(MM),YEST,RESI
      REWIND 13
C
C        TEST TO SEE WHETHER ALL SELECTIONS ARE COMPLETED
C
  185 IF(NSEL-NS) 190, 100, 100
  190 NSEL=NSEL+1
      WRITE (6,17)
      GO TO 145
C
  200 CONTINUE
      END
```

CANONICAL CORRELATION

Problem Description

An analysis of the interrelations between two sets of variables measured on the same subjects is performed by this program. These variables are predictors in one set and criteria in the other set, but it is irrelevant whether the variables in the first set or in the second set are considered as the prediction variables. The canonical correlation, which gives the maximum correlation between linear functions of the two sets of variables, is calculated. χ^2 is also computed to test the significance of canonical correlation.

The sample problem for canonical correlation consists of four variables in the first set (left-hand side) and three variables in the second set (right-hand side) as presented in Table 6. These two sets of measurements have been made on 23 subjects.

Table 6. Sample Data for Canonical Correlation

	First set				Second set		
Observation	X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3
1	191	155	65	19	179	145	70
2	195	149	70	20	201	152	69
3	181	148	71	19	185	149	75
4	183	153	82	18	188	149	86
5	176	144	67	18	171	142	71
6	208	157	81	22	192	152	77
7	189	150	75	21	190	149	72
8	197	159	90	20	189	152	82
9	188	152	76	19	197	159	84
10	192	150	78	20	187	151	72
11	179	158	99	18	186	148	89
12	183	147	65	18	174	147	70
13	174	150	71	19	185	152	65
14	190	159	91	19	195	157	99
15	188	151	98	20	187	158	87
16	163	137	59	18	161	130	63
17	195	155	85	20	183	158	81
18	196	153	80	21	173	148	74
19	181	145	77	20	182	146	70
20	175	140	70	19	165	137	81
21	192	154	69	20	185	152	63
22	174	143	79	20	178	147	73
23	176	139	70	20	176	143	69

Program

Description

The canonical correlation program consists of the main routine named MCANO, a special input subroutine named DATA, and five subroutines from the Scientific Subroutine Package: CORRE, CANOR, MINV, NROOT, and EIGEN.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to 20 variables, including both the first set of variables (that is, left-hand variables) and the second set of variables (that is, right-hand variables) The number of variables in the first set must be greater than or equal to the number of variables in the second set.
2. Up to 99,999 observations
3. (12F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions it is not necessary to modify the sample program. However, if there are more than 20 variables, dimension statements in the sample main program must be modified to handle the particular problem. Similarly, if input data cards are prepared using a different format, the input format in the input subroutine, DATA, must be modified. The general rules for program modification are desc ibed later.

Input

Control Card

One control card is required for each problem and is read by the main program, MCANO. This card is prepared as follows:

Columns	Contents	For Sample Problem
1 - 6	Problem number (may be alphameric)	SAMPLE
7 - 11	Number of observations	00023
12 - 13	Number of variables in the first set (that is, left-hand variables)*	04
14 - 15	Number of variables in the second set (that is, right-hand variables)	03

*The number of variables in the first set must be greater than or equal to the number of variables in the second set.

Leading zeros are not required to be keypunched.

Data Cards

Since input data are read into the computer one observation at a time, each row of data in Table 6 is keypunched on a separate card using the format (12F6.0). This format assumes twelve 6-column fields per card.

If there are more than twelve variables in a problem, each row of data is continued on the second card until the last data point is keypunched. However, each row of data must begin on a new card.

Deck Setup

The deck setup is shown in Figure 47.

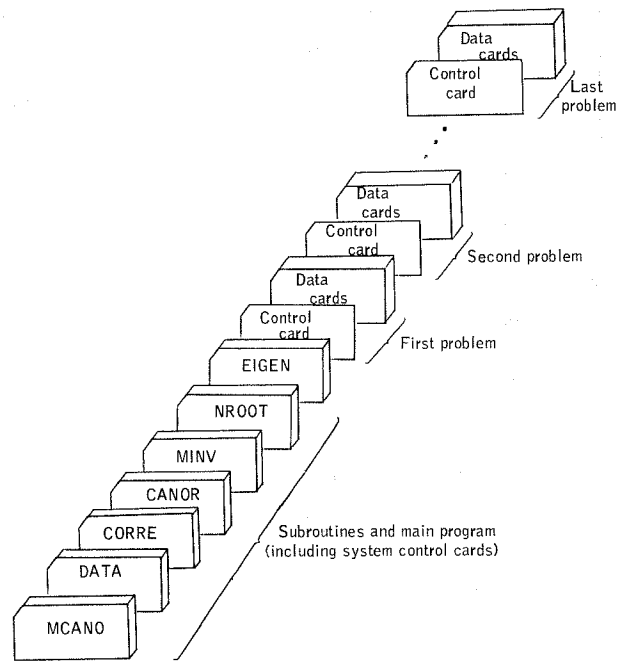


Figure 47. Deck setup (canonical correlation)

Sample

The listing of input cards for the sample problem is presented in Figure 48.

Output

Description

The output of the sample program for canonical correlation includes:

1. Means
2. Standard deviations
3. Correlation coefficients
4. Eigenvalues and corresponding canonical correlation
5. Lambda
6. Chi-square and degrees of freedom
7. Coefficients for left- and right-hand variables

Sample

The output listing for the sample problem is shown in Figure 49 of this sample problem.

```
/DATA
SAMPLE000230403
  191  155   65   19  179  145   70   10
  195  149   70   20  201  152   69   20
  181  148   71   19  185  149   75   30
  183  153   82   18  188  149   86   40
  176  144   67   18  171  142   71   50
  208  157   81   22  192  152   77   60
  189  150   75   21  190  149   72   70
  197  159   90   20  189  152   82   80
  188  152   76   19  197  159   84   90
  192  150   78   20  187  151   72  100
  179  158   99   18  186  148   89  110
  183  147   65   18  174  147   70  120
  174  150   71   19  185  152   65  130
  190  159   91   19  195  157   99  140
  188  151   98   20  187  158   87  150
  163  137   59   18  161  130   63  160
  195  155   85   20  183  158   81  170
  196  153   80   21  173  148   74  180
  181  145   77   20  182  146   70  190
  175  140   70   19  165  137   81  200
  192  154   69   20  185  152   63  210
  174  143   79   20  178  147   73  220
  176  139   70   20  176  143   69  230
                                     240
                                     250
```

Figure 48. Input card listing (canonical correlation)

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize the user with the program modification, the following general rules are supplied in terms of the sample problem.

1. Changes in the dimension statements of the main program, MCANO:
   a. The dimension of arrays XBAR, STD, CANR, CHISQ, and NDF must be greater than or equal to the total number of variables m ($m = p + q$, where p is the number of left-hand variables and q is the number of right-hand variables). Since there are seven variables, four on left and three on right, the value of m is 7.

```
CANONICAL CORRELATION.....SAMPLE

  NO. OF OBSERVATIONS          23
  NO. OF LEFT HAND VARIABLES    4
  NO. OF RIGHT HAND VARIABLES   3

MEANS
     185.47826     149.91304      76.86955      19.47826     183.00000     148.82609      75.73912

STANDARD DEVIATIONS
      10.10342       6.31673      10.46338       1.08165       9.84424       6.73965       9.05647

CORRELATION COEFFICIENTS

ROW  1
    1.00000     0.74852     0.37082     0.66441     0.62291     0.66080     0.24683

ROW  2
    0.74852     1.00000     0.63252     0.22590     0.66811     0.72780     0.53194

ROW  3
    0.37082     0.63252     1.00000     0.20657     0.47394     0.60169     0.79684

ROW  4
    0.66441     0.22590     0.20657     1.00000     0.32870     0.34863    -0.10733

ROW  5
    0.62291     0.66811     0.47394     0.32870     1.00000     0.82555     0.39258

ROW  6
    0.66080     0.72780     0.60169     0.34863     0.82555     1.00000     0.47657

ROW  7
    0.24683     0.53194     0.79684    -0.10733     0.39258     0.47657     1.00000


  NUMBER OF      LARGEST      CORRESPONDING                                  DEGREES
 EIGENVALUES    EIGENVALUE     CANONICAL        LAMBDA      CHI-SQUARE        OF
   REMOVED      REMAINING     CORRELATION                                   FREEDOM

      0           0.79880        0.89376       0.11598      40.93277         12
      1           0.41910        0.64738       0.57644      10.46676          6
      2           0.00767        0.08760       0.99233       0.14636          2


CANONICAL CORRELATION     0.89376

  COEFFICIENTS FOR LEFT HAND VARIABLES
      0.66310       -0.16059        1.05822       -0.56651

  COEFFICIENTS FOR RIGHT HAND VARIABLES
     -0.02133        0.44090        0.89730


CANONICAL CORRELATION     0.64738

  COEFFICIENTS FOR LEFT HAND VARIABLES
      0.09454       -0.83915        0.66309       -0.64892

  COEFFICIENTS FOR RIGHT HAND VARIABLES
     -0.43841       -0.55503        0.70692


CANONICAL CORRELATION     0.08760

  COEFFICIENTS FOR LEFT HAND VARIABLES
      0.02681        0.36055       -0.28827       -0.32496

  COEFFICIENTS FOR RIGHT HAND VARIABLES
      0.70325       -0.70384        0.10028

1E0031217
```

Figure 49. Output listing (canonical correlation)

b. The dimension of array RX must be greater than or equal to the product of m x m. For the sample problem this product is 49 = 7 x 7.
c. The dimension of array R must be greater than or equal to (m + 1) m/2. For the sample problem this number is 28 = (7 + 1)7/2.
d. The dimension of array COEFL must be greater than or equal to the product of p x q. For the sample problem this product is 12 = 4 x 3.
e. The dimension of array COEFR must be greater than or equal to the product of q x q. For the sample problem this product is 9 = 3 x 3.

2. Changes in the input format statement of the special input subroutine, DATA:

Only the format statement for input data may be changed. Since sample data are either two- or three-digit numbers, rather than using six-column fields as in the sample problem, each row of data may be keypunched in seven 3-column fields, and, if so, the format is changed to (7F3.0).

The special input subroutine, DATA, is normally written by the user to handle different formats for different problems. The user may modify this subroutine to perform testing of input data, transformation of data, and so on.

Operating Instructions

The sample program for canonical correlation is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 19 seconds.

```
C                                                                       MCAN  10
C     ...................................................................MCAN  20
C                                                                       MCAN  30
C        SAMPLE MAIN PROGRAM FOR CANONICAL CORRELATION - MCANO          MCAN  40
C                                                                       MCAN  50
C        PURPOSE                                                        MCAN  60
C           (1) READ THE PROBLEM PARAMETER CARD FOR A CANONICAL         MCAN  70
C           CORRELATION, (2) CALL TWO SUBROUTINES TO CALCULATE SIMPLE   MCAN  80
C           CORRELATIONS, CANONICAL CORRELATIONS, CHI-SQUARES, DEGREES  MCAN  90
C           OF FREEDOM FOR CHI-SQUARES, AND COEFFICIENTS FOR LEFT AND   MCAN 100
C           RIGHT HAND VARIABLES, NAMELY CANONICAL VARIATES, AND (3)    MCAN 110
C           PRINT THE RESULTS.                                          MCAN 120
C                                                                       MCAN 130
C        REMARKS                                                        MCAN 140
C           THE NUMBER OF LEFT HAND VARIABLES MUST BE GREATER THAN      MCAN 150
C           OR EQUAL TO THE NUMBER OF RIGHT HAND VARIABLES.             MCAN 160
C                                                                       MCAN 170
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MCAN 180
C           CORRE  (WHICH, IN TURN, CALLS THE INPUT SUBROUTINE NAMED    MCAN 190
C                   DATA.)                                              MCAN 200
C           CANOR  (WHICH, IN TURN, CALLS THE SUBROUTINES MINV AND      MCAN 210
C                   NROOT.  NROOT, IN TURN, CALLS THE SUBROUTINE EIGEN.) MCAN 220
C                                                                       MCAN 230
C        METHOD                                                         MCAN 240
C           REFER TO W. W. COOLEY AND P. R. LOHNES, "MULTIVARIATE PRO-  MCAN 250
C           CEDURES FOR THE BEHAVIORAL SCIENCES", JOHN WILEY AND SONS,  MCAN 260
C           1962, CHAPTER 3.                                            MCAN 270
C                                                                       MCAN 280
C     ...................................................................MCAN 290
C                                                                       MCAN 300
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO THE     MCAN 310
C     TOTAL NUMBER OF VARIABLES M (M=MP+MQ, WHERE MP IS THE NUMBER OF   MCAN 320
C     LEFT-HAND VARIABLES, AND MQ IS THE NUMBER OF RIGHT HAND VARI-     MCAN 330
C     ABLES)..                                                          MCAN 340
C                                                                       MCAN 350
      DIMENSION XBAR(20),STD(20),CANR(20),CHISQ(20),NDF(20)             MCAN 360
C                                                                       MCAN 370
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MCAN 380
C     PRODUCT OF M*M..                                                  MCAN 390
C                                                                       MCAN 400
      DIMENSION RX(400)                                                 MCAN 410
C                                                                       MCAN 420
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO          MCAN 430
C     (M+1)*M/2..                                                       MCAN 440
C                                                                       MCAN 450
      DIMENSION R(210)                                                  MCAN 460
C                                                                       MCAN 470
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MCAN 480
C     PRODUCT OF MP*MQ..                                                MCAN 490
C                                                                       MCAN 500
      DIMENSION COEFL(400)                                              MCAN 510
C                                                                       MCAN 520
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MCAN 530
C     PRODUCT OF MQ*MQ..                                                MCAN 540
C                                                                       MCAN 550
      DIMENSION COEFR(400)                                              MCAN 560
C                                                                       MCAN 570
C     ...................................................................MCAN 580
C                                                                       MCAN 590
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  MCAN 600
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      MCAN 610
C        STATEMENT WHICH FOLLOWS.                                       MCAN 620
C                                                                       MCAN 630
C     DOUBLE PRECISION XBAR,STD,RX,R,CANR,CHISQ,COEFL,COEFR             MCAN 640
C                                                                       MCAN 650
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    MCAN 660
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      MCAN 670
C        ROUTINE.                                                       MCAN 680
C                                                                       MCAN 690
C        ...............................................................MCAN 700
C                                                                       MCAN 710
    1 FORMAT(A4,A2,I5,2I2)                                              MCAN 720
    2 FORMAT(27H1CANONICAL CORRELATION.....,A4,A2//22H   NO. OF OBSERVAT MCAN 730
     1IONS,8X,I4/29H   NO. OF LEFT HAND VARIABLES,I5/30H   NO. OF RIGHT MCAN 740
     3HAND VARIABLES,I4/)                                               MCAN 750
    3 FORMAT(6H0MEANS/(8F15.5))                                         MCAN 760
    4 FORMAT(20H0STANDARD DEVIATIONS/(8F15.5))                          MCAN 770
    5 FORMAT(25H0CORRELATION COEFFICIENTS)                              MCAN 780
    6 FORMAT(4HOROW,I3/(10F12.5))                                       MCAN 790
    7 FORMAT(1H0//12H   NUMBER OF,7X,7HLARGEST,7X,13HCORRESPONDING,31X,7MCAN 800
     1HDEGREES/13H   EIGENVALUES,5X,10HEIGENVALUE,7X,9HCANONICAL,7X,6HLAMMCAN 810
     2BDA,5X,10HCHI-SQUARE,7X,2HDF/4X,7HREMOVED,7X,9HREMAINING,7X,11HCORMCAN 820
     3RELATION,32X,7HFREEDOM/)                                          MCAN 830
    8 FORMAT(1H ,I7,F19.5,F16.5,2F14.5,5X,I5)                           MCAN 840
    9 FORMAT(1H0/22H CANONICAL CORRELATION,F12.5)                       MCAN 850
   10 FORMAT(39H0  COEFFICIENTS FOR LEFT HAND VARIABLES/(8F15.5))       MCAN 860
   11 FORMAT(40H0  COEFFICIENTS FOR RIGHT HAND VARIABLES/(8F15.5))      MCAN 870
C                                                                       MCAN 880
C     ...................................................................MCAN 890
C                                                                       MCAN 900
C     READ PROBLEM PARAMETER CARD                                       MCAN 910
C                                                                       MCAN 920
  100 READ (5,1) PR,PR1,N,MP,MQ                                         MCAN 930
C        PR.......PROBLEM NUMBER (MAY BE ALPHAMERIC)                    MCAN 940
C        PR1......PROBLEM NUMBER (CONTINUED)                            MCAN 950
C        N........NUMBER OF OBSERVATIONS                                MCAN 960
C        MP.......NUMBER OF LEFT HAND VARIABLES                         MCAN 970
C        MQ.......NUMBER OF RIGHT HAND VARIABLES                        MCAN 980
C                                                                       MCAN 990
      WRITE (6,2) PR,PR1,N,MP,MQ                                        MCAN1000
C                                                                       MCAN1010
      M=MP+MQ                                                           MCAN1020
      ID=0                                                              MCAN1030
      X=0.0                                                             MCAN1040
C                                                                       MCAN1050
      CALL CORRE (N,M,ID,X,XBAR,STD,RX,R,CANR,CHISQ,COEFL)              MCAN1060
C                                                                       MCAN1070
C     PRINT MEANS, STANDARD DEVIATIONS, AND CORRELATION                 MCAN1080
C     COEFFICIENTS OF ALL VARIABLES                                     MCAN1090
C                                                                       MCAN1100
      WRITE (6,3) (XBAR(I),I=1,M)                                       MCAN1110
      WRITE (6,4) (STD(I),I=1,M)                                        MCAN1120
      WRITE (6,5)                                                       MCAN1130
      DO 160 I=1,M                                                      MCAN1140
      DO 150 J=1,M                                                      MCAN1150
      IF(I-J) 120, 130, 130                                             MCAN1160
  120 L=I+(J*J-J)/2                                                     MCAN1170
      GO TO 140                                                         MCAN1180
  130 L=J+(I*I-I)/2                                                     MCAN1190
  140 CANR(J)=R(L)                                                      MCAN1200
  150 CONTINUE                                                          MCAN1210
  160 WRITE (6,6) I,(CANR(J),J=1,M)                                     MCAN1220
C                                                                       MCAN1230
      CALL CANOR (N,MP,MQ,R,XBAR,STD,CANR,CHISQ,NDF,COEFR,COEFL,RX)     MCAN1240
C                                                                       MCAN1250
C     PRINT EIGENVALUES, CANONICAL CORRELATIONS, LAMBDA, CHI-SQUARES,   MCAN1260
C     DEGREES OF FREEDOMS                                               MCAN1270
C                                                                       MCAN1280
      WRITE (6,7)                                                       MCAN1290
      DO 170 I=1,MQ                                                     MCAN1300
      N1=I-1                                                            MCAN1310
C                                                                       MCAN1320
C        TEST WHETHER EIGENVALUE IS GREATER THAN ZERO                   MCAN1330
C                                                                       MCAN1340
      IF(XBAR(I)) 165, 165, 170                                         MCAN1350
  165 MM=N1                                                             MCAN1360
      GO TO 175                                                         MCAN1370
  170 WRITE (6,8) N1,XBAR(I),CANR(I),STD(I),CHISQ(I),NDF(I)             MCAN1380
      MM=MQ                                                             MCAN1390
C                                                                       MCAN1400
C     PRINT CANONICAL COEFFICIENTS                                      MCAN1410
C                                                                       MCAN1420
  175 N1=0                                                              MCAN1430
      N2=0                                                              MCAN1440
      DO 200 I=1,MM                                                     MCAN1450
      WRITE (6,9) CANR(I)                                               MCAN1460
      DO 180 J=1,MP                                                     MCAN1470
      N1=N1+1                                                           MCAN1480
  180 XBAR(J)=COEFL(N1)                                                 MCAN1490
      WRITE (6,10) (XBAR(J),J=1,MP)                                     MCAN1500
      DO 190 J=1,MQ                                                     MCAN1510
      N2=N2+1                                                           MCAN1520
  190 XBAR(J)=COEFR(N2)                                                 MCAN1530
      WRITE (6,11) (XBAR(J),J=1,MQ)                                     MCAN1540
  200 CONTINUE                                                          MCAN1550
      GO TO 100                                                         MCAN1560
      END                                                               MCAN1570
```

ANALYSIS OF VARIANCE

Problem Description

An analysis of variance is performed for a factorial design by use of three special operators suggested by H. O. Hartley.* The analysis of many other designs can be derived by reducing them first to factorial designs, and then pooling certain components of the analysis-of-variance table.

Consider a three-factor factorial experiment in a randomized complete block design as presented in Table 7. In this experiment, factor A has four levels, factors B and C have three levels, and the entire experiment is replicated twice. The replicates are completely unrelated and do not constitute a factor. Nevertheless, for the purpose of this program a four-factor experiment (with factors A, B, C, and R) is assumed. Thus, each element of the data in Table 5 may be represented in the form:

$$x_{abcr} \quad \text{where} \quad \begin{aligned} a &= 1,2,3,4 \\ b &= 1,2,3 \\ c &= 1,2,3 \\ r &= 1,2 \end{aligned}$$

The general principle of the analysis-of-variance procedure used in the program is first to perform a formal factorial analysis and then to pool certain components in accordance with summary instructions that specifically apply to the particular design. The summary instructions for four different designs are presented in the output section.

Program

Description

The analysis-of-variance program consists of the main routine named ANOVA, and three subroutines from the Scientific Subroutine Package: AVDAT, AVCAL, and MEANQ.

*H. O. Hartley, "Analysis of Variance", Mathematical Methods for Digital Computers, edited by A. Ralston and H. Wilf, John Wiley and Sons, 1962, Chapter 20.

Table 7. Sample Data for Analysis of Variance

Replicate (Block)		b_1				b_2				b_3			
		a_1	a_2	a_3	a_4	a_1	a_2	a_3	a_4	a_1	a_2	a_3	a_4
$r_1 \cdots$	c_1	3	10	9	8	24	8	9	3	2	8	9	8
	c_2	4	12	3	9	22	7	16	2	2	2	7	2
	c_3	5	10	5	8	23	9	17	3	2	8	6	3
$r_2 \cdots$	c_1	2	14	9	13	29	16	11	3	2	7	5	3
	c_2	7	11	5	8	28	18	10	6	6	6	5	9
	c_3	9	10	27	8	28	16	11	7	8	9	8	15

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to six-factor factorial experiment
2. Up to a total of 3000 data points. The total number of data points in a problem is calculated by the formula:

$$T = \prod_{i=1}^{k} (\text{LEVEL}_i + 1)$$

where LEVEL_i = number of levels of ith factor
k = number of factors
Π = notation for cumulative multiplication

3. (12F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions it is not necessary to modify the sample program. However, if there are more than six factors or if the total number of data points is more than 3000, dimension statements in the sample main program must be modified. Similarly, if input data cards are prepared using a different format, the input format statement in the sample main program must be modified. The general rules for program modifications are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, ANOVA. This card is prepared as follows:

Columns	Contents	For Sample Problem
1- 6	Problem number (may be alphameric)	SAMPLE
7- 8	Number of factors	04
9-15	Blank	
16	Label for the first factor	A
17-20	Number of levels of the first factor	0004
21	Label for the second factor	B
22-25	Number of levels of the second factor	0003
26	Label for the third factor	C
27-30	Number of levels of the third factor	0003
31	Label for the fourth factor	R
32-35	Number of levels of the fourth factor	0002
. . .		
66	Label for the eleventh factor (if present)	
67-70	Number of levels of the eleventh factor	

If there are more than eleven factors, continue to the second card in the same manner.

Columns	Contents
1	Label for the twelfth factor
2 - 5	Number of levels of the twelfth factor
. . .	
etc.	

Leading zeros are not required to be keypunched.

Data Cards

Data are keypunched in the following order: X_{1111}, X_{2111}, X_{3111}, X_{4111}, X_{1211}, X_{2211}, X_{3211}, ..., X_{4332}. In other words, the innermost subscript is changed first -- namely, the first factor -- and then the second, third, and fourth subscripts. In the sample problem, the first subscript corresponds to factor A, and the second, third, and fourth subscripts to factors B, C, and R. Since the number of data fields per card is twelve, implied by the format (12F6.0), each row in Table 7 is keypunched on a separate card.

Deck Setup

The deck setup is shown in Figure 50.

Sample

The listing of input cards for the sample problem is presented in Figure 51.

Output

Description

The output of the sample analysis-of-variance program includes the numbers of levels of factors as input, the mean of all data, and the table of analysis of variance. In order to complete the analysis of variance properly, however, certain components in the table may need to be pooled. This is accomplished by means of summary instructions that specifically apply to the particular experiment. These are presented in Table 8.

As mentioned earlier, the sample problem is a randomized complete block design with three factors replicated twice. Therefore, it is necessary to pool certain components in the table of analysis of variance shown in Figure 52. Specifically, the components AR, BR, ABR, CR, ACR, BCR, and ABCR are combined into one value called the error term. The result is indicated in Figure 52. Since these data are purely hypothetical, interpretations of the various effects are not made.

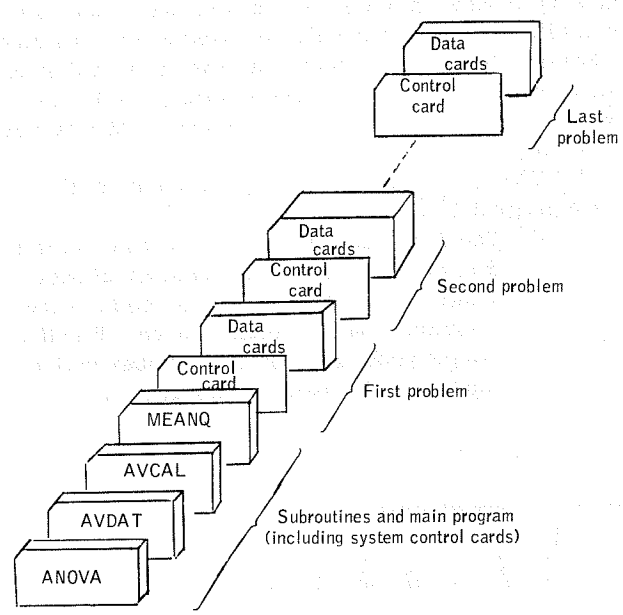


Figure 50. Deck setup (analysis of variance)

Table 8. Instructions to Summarize Components of Analysis of Variance

	Single Classification with Replicates	Two-way Classification with Cell Replicates	Randomized Complete Block with Two Factors	Split Plot
(Input) Factor No. 1 2 3	Groups = A Replicates = R	Rows = A Columns = B Replicates = R	Factor 1 = A Factor 2 = B Blocks = R	Main treatment = A Subtreatment = B Blocks = R
(Output) Sums of squares	A R AR	A B AB R AR BR ABR	A B AB R AR BR ABR	A B AB R AR BR ABR
Summary instruction	Error = R + (AR)	Error = R + (AR) +(BR)+(ABR)	Error = (AR)+(BR) +(ABR)	Error = (BR)+(ABR) (b)
Analysis of variance	Groups A Error	Rows A Columns B Interaction AB Error	Factor 1 A Factor 2 B Interaction AB Blocks R Error	Main treatment A Blocks R Error (a) AR Subtreatment B Interaction AB Error (b)

```
ANALYSIS OF VARIANCE-----SAMPLE

LEVELS OF FACTORS
  A        4
  B        3
  C        3
  R        2

GRAND MEAN              9.40278

SOURCE OF                    SUMS OF        DEGREES OF        MEAN
VARIATION                    SQUARES        FREEDOM           SQUARES

A                          229.04166            3             76.34721
B                          722.69434            2            361.34717
AB                        1382.08325            6            230.34720
C                           55.11110            2             27.55554
AC                          42.00000            6              7.00000
BC                          13.13889            4              3.28472
ABC                        140.75000           12             11.72917
R            (Block)       141.68054            1            141.68054
AR                          18.81944            3              6.27315
BR                           6.02778            2              3.01389
ABR                        176.97221            6             29.49536
CR           (Error)*       40.77777 (506.81940)*  2  (35)*   20.38889  (14.48054)*
ACR                         50.55554            6              8.42592
BCR                         62.63889            4             15.65972
ABCR                       151.02777           12             12.58565
TOTAL                     3233.31763           71
```

*summarized after the computer output is obtained.

Figure 52. Output listing (analysis of variance)

Sample

The output listing for the sample problem is shown in Figure 52.

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize the user with the program modification, the following general rules are supplied in terms of the sample problem.

1. Changes in the dimension statements of the main program, ANOVA:
   a. The dimension of array X must be greater than or equal to the total number of data points as calculated by the formula in the program capacity section above. For the sample problem, the total number of data points is 240 = (4+1)(3+1)(3+1)(2+1).
   b. The dimension of arrays HEAD, LEVEL, ISTEP, KOUNT, and LASTS must be greater than or equal to the number of factors, k. Since there are four factors in the sample problem (4 = 3 original factors + 1 pseudo factor), the value of k is 4.
   c. The dimension of arrays SUMSQ, NDF, and SMEAN must be greater than or equal to $n = 2^k-1$, where k is the number of factors. For the sample problem, the value of n is $15 = 2^4-1$.
2. Change in the input format statement of the main program, ANOVA:
   Only the format statement for input data may be changed. Since sample data are either one- or two-digit numbers, rather than using six-column fields as in the sample problem, each data may be keypunched in a two-column field, and, if so, the format is changed to (12F2.0). This format assumes twelve 2-column fields per card, beginning in column 1.

```
/DATA
SAMPLE04          A0004B0003C0003R0002                                        10
   3   10    9    8   24    8    9    3    2    8    9    8                 20
   4   12    3    9   22    7   16    2    2    2    7    2                 30
   5   10    5    8   23    9   17    3    2    8    6    3                 40
   2   14    9   13   29   16   11    3    2    7    5    3                 50
   7   11    5    8   28   18   10    6    6    6    5    9                 60
   9   10   27    8   28   16   11    7    8    9    8   15                 70
                                                                            80
```

Figure 51. Input listing (analysis of variance)

Operating Instructions

The sample analysis-of-variance program is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 11 seconds.

```
C                                                                       ANOV  10
C     ...................................................................ANOV  20
C                                                                       ANOV  30
C        SAMPLE MAIN PROGRAM FOR ANALYSIS OF VARIANCE - ANOVA           ANOV  40
C                                                                       ANOV  50
C        PURPOSE                                                        ANOV  60
C           (1) READ THE PROBLEM PARAMETER CARD FOR ANALYSIS OF VARI-   ANOV  70
C           ANCE, (2) CALL THE SUBROUTINES FOR THE CALCULATION OF SUMS  ANOV  80
C           OF SQUARES, DEGREES OF FREEDOM AND MEAN SQUARE, AND         ANOV  90
C           (3) PRINT FACTOR LEVELS, GRAND MEAN AND ANALYSIS OF VARI-   ANOV 100
C           ANCE TABLE.                                                 ANOV 110
C                                                                       ANOV 120
C        REMARKS                                                        ANOV 130
C           THE PROGRAM HANDLES ONLY COMPLETE FACTORIAL DESIGNS.  THERE-ANOV 140
C           FORE, OTHER EXPERIMENTAL DESIGN MUST BE REDUCED TO THIS FORMANOV 150
C           PRIOR TO THE USE OF THE PROGRAM.                            ANOV 160
C                                                                       ANOV 170
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ANOV 180
C           AVOAT                                                       ANOV 190
C           AVCAL                                                       ANOV 200
C           MEANQ                                                       ANOV 210
C                                                                       ANOV 220
C        METHOD                                                         ANOV 230
C           THE METHOD IS BASED ON THE TECHNIQUE DISCUSSED BY H. O.     ANOV 240
C           HARTLEY IN 'MATHEMATICAL METHODS FOR DIGITAL COMPUTERS',    ANOV 250
C           EDITED BY A. RALSTON AND H. WILF, JOHN WILEY AND SONS,      ANOV 260
C           1962, CHAPTER 20.                                           ANOV 270
C                                                                       ANOV 280
C     ...................................................................ANOV 290
C                                                                       ANOV 300
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      ANOV 310
C     CUMULATIVE PRODUCT OF EACH FACTOR LEVEL PLUS ONE (LEVEL(I)+1)     ANOV 320
C     FOR I=1 TO K, WHERE K IS THE NUMBER OF FACTORS..                  ANOV 330
C                                                                       ANOV 340
      DIMENSION X(3000)                                                 ANOV 350
C                                                                       ANOV 360
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO THE     ANOV 370
C     NUMBER OF FACTORS..                                               ANOV 380
C                                                                       ANOV 390
      DIMENSION HEAD(6),LEVEL(6),ISTEP(6),KOUNT(6),LASTS(6)             ANOV 400
C                                                                       ANOV 410
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO 2 TO    ANOV 420
C     THE K-TH POWER MINUS 1, ((2**K)-1)..                              ANOV 430
C                                                                       ANOV 440
      DIMENSION SUMSQ(63),NDF(63),SMEAN(63)                             ANOV 450
C                                                                       ANOV 460
C     THE FOLLOWING DIMENSION IS USED TO PRINT FACTOR LABELS IN ANALYSISANOV 470
C     OF VARIANCE TABLE AND IS FIXED..                                  ANOV 480
C                                                                       ANOV 490
      DIMENSION FMT(15)                                                 ANOV 500
C     ...................................................................ANOV 510
C                                                                       ANOV 520
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  ANOV 530
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      ANOV 540
C        STATEMENT WHICH FOLLOWS.                                       ANOV 550
C                                                                       ANOV 560
C     DOUBLE PRECISION X,GMEAN,SUMSQ,SMEAN,SUM                          ANOV 570
C                                                                       ANOV 580
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    ANOV 590
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      ANOV 600
C        ROUTINE.                                                       ANOV 610
C                                                                       ANOV 620
C        ...............................................................ANOV 630
C                                                                       ANOV 640
    1 FORMAT(A4,A2,I2,A4,3X,11(A1,I4)/(A1,I4,A1,I4,A1,I4,A1,I4,A1,I4))  ANOV 650
    2 FORMAT(26H1ANALYSIS OF VARIANCE.....A4,A2//)                      ANOV 660
    3 FORMAT(18HOLEVELS OF FACTORS/(3X,A1,7X,I4))                       ANOV 670
    4 FORMAT(1H0//11H GRAND MEANF20.5///)                               ANOV 680
    5 FORMAT(10HOSOURCE DF18X,7HSUMS DF10X,10HDEGREES OF9X,4HMEAN/10H VAANOV 690
   1RIATION18X,7HSQUARES11X,7HFREEDOM10X,7HSQUARES/)                    ANOV 700
    6 FORMAT(1H 15A1,F20.5,10X,I6,F20.5)                                ANOV 710
    7 FORMAT(6H TOTAL10X,F20.5,10X,I6)                                  ANOV 720
    8 FORMAT(12F6.0)                                                    ANOV 730
C                                                                       ANOV 740
C     ...................................................................ANOV 750
C                                                                       ANOV 760
C     READ PROBLEM PARAMETER CARD                                       ANOV 770
C                                                                       ANOV 780
  100 READ (5,1) PR,PR1,K,BLANK,(HEAD(I),LEVEL(I),I=1,K)                ANOV 790
C        PR.....PROBLEM NUMBER (MAY BE ALPHAMERIC)                      ANOV 800
C        PR1....PROBLEM NUMBER (CONTINUED)                              ANOV 810
C        K......NUMBER OF FACTORS                                       ANOV 820
C        BLANK..BLANK FIELD                                             ANOV 830
C        HEAD...FACTOR LABELS                                           ANOV 840
C        LEVEL..LEVELS OF FACTORS                                       ANOV 850
C                                                                       ANOV 860
C     PRINT PROBLEM NUMBER AND LEVELS OF FACTORS                        ANOV 870
C                                                                       ANOV 880
      WRITE (6,2) PR,PR1                                                ANOV 890
      WRITE (6,3) (HEAD(I),LEVEL(I),I=1,K)                              ANOV 900
C                                                                       ANOV 910
C     CALCULATE TOTAL NUMBER OF DATA                                    ANOV 920
C                                                                       ANOV 930
      N=LEVEL(1)                                                        ANOV 940
      DO 102 I=2,K                                                      ANOV 950
  102 N=N*LEVEL(I)                                                      ANOV 960
C                                                                       ANOV 970
C     READ ALL INPUT DATA                                               ANOV 980
C                                                                       ANOV 990
      READ (5,8) (X(I),I=1,N)                                           ANOV1000
C                                                                       ANOV1010
      CALL AVOAT (K,LEVEL,N,X,L,ISTEP,KOUNT)                            ANOV1020
      CALL AVCAL (K,LEVEL,X,L,ISTEP,LASTS)                              ANOV1030
      CALL MEANQ (K,LEVEL,X,GMEAN,SUMSQ,NDF,SMEAN,ISTEP,KOUNT,LASTS)    ANOV1040
C                                                                       ANOV1050
C     PRINT GRAND MEAN                                                  ANOV1060
C                                                                       ANOV1070
      WRITE (6,4) GMEAN                                                 ANOV1080
C                                                                       ANOV1090
C     PRINT ANALYSIS OF VARIANCE TABLE                                  ANOV1100
C                                                                       ANOV1110
      WRITE (6,5)                                                       ANOV1120
      LL=(2**K)-1                                                       ANOV1130
      ISTEP(1)=1                                                        ANOV1140
      DO 105 I=2,K                                                      ANOV1150
  105 ISTEP(I)=0                                                        ANOV1160
      DO 110 I=1,15                                                     ANOV1170
  110 FMT(I)=BLANK                                                      ANOV1180
      NN=0                                                              ANOV1190
      SUM=0.0                                                           ANOV1200
  120 NN=NN+1                                                           ANOV1210
      L=0                                                               ANOV1220
      DO 140 I=1,K                                                      ANOV1230
      FMT(I)=BLANK                                                      ANOV1240
      IF(ISTEP(I)) 130, 140, 130                                        ANOV1250
  130 L=L+1                                                             ANOV1260
      FMT(L)=HEAD(I)                                                    ANOV1270
  140 CONTINUE                                                          ANOV1280
      WRITE (6,6) (FMT(I),I=1,15),SUMSQ(NN),NDF(NN),SMEAN(NN)           ANOV1290
      SUM=SUM+SUMSQ(NN)                                                 ANOV1300
      IF(NN-LL) 145, 170, 170                                           ANOV1310
  145 DO 160 I=1,K                                                      ANOV1320
      IF(ISTEP(I)) 147, 150, 147                                        ANOV1330
  147 ISTEP(I)=0                                                        ANOV1340
      GO TO 160                                                         ANOV1350
  150 ISTEP(I)=1                                                        ANOV1360
      GO TO 120                                                         ANOV1370
  160 CONTINUE                                                          ANOV1380
  170 N=N-1                                                             ANOV1390
      WRITE (6,7) SUM,N                                                 ANOV1400
      GO TO 100                                                         ANOV1410
      END                                                               ANOV1420
```

DISCRIMINANT ANALYSIS

Problem Description

A set of linear functions is calculated from data on many groups for the purpose of classifying new individuals into one of several groups. The classification of an individual into a group is performed by evaluating each of the calculated linear functions, then finding the group for which the value is the largest.

The sample problem for discriminant analysis consists of four groups of observations as presented in Table 9. The number of observations in the first group is eight; the second group, seven; the third group, seven; and the fourth group, eight. The number of variables is six in all groups.

Program

Description

The discriminant analysis program consists of the main routine named MDISC, and three subroutines from the Scientific Subroutine Package: DMATX, MINV, and DISCR.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to five groups
2. Up to ten variables
3. Up to a total number of 250 observations in all groups combined
4. (12F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions it is not necessary to modify the sample program. However, if there are more than five groups, more than ten variables, or more than 250 observations, dimension statements in the sample

Table 9. Sample Data for Discriminant Analysis

	Observation	X_1	X_2	X_3	X_4	X_5	X_6
Group 1	1	3	10	9	8	24	8
	2	4	12	3	8	22	7
	3	9	3	2	8	9	8
	4	16	2	2	2	7	2
	5	5	10	5	8	23	9
	6	17	3	2	8	6	3
	7	2	10	9	8	29	16
	8	7	10	5	8	28	18
Group 2	1	9	10	27	8	28	16
	2	11	7	8	9	8	15
	3	8	10	2	8	27	16
	4	1	6	8	14	14	13
	5	7	8	9	6	18	2
	6	7	9	8	2	19	9
	7	7	10	5	8	27	17
Group 3	1	3	11	9	15	20	10
	2	9	4	10	7	9	9
	3	4	13	10	7	21	15
	4	8	5	16	16	16	7
	5	6	9	10	5	23	11
	6	8	10	5	8	27	16
	7	17	3	2	7	6	3
Group 4	1	3	10	8	8	23	8
	2	4	12	3	8	23	7
	3	9	3	2	8	21	7
	4	15	2	2	2	7	2
	5	9	10	26	8	27	16
	6	8	9	2	8	26	16
	7	7	8	6	9	18	2
	8	7	10	5	8	26	16

main program must be modified to handle this particular problem. Similarly, if input data cards are prepared using a different format, the input format statement in the sample main program must be modified. The general rules for program modification are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, MDISC. This card is prepared as follows:

Columns	Contents	For Sample Problem
1- 6	Problem number (may be alphameric)	SAMPLE
7- 8	Number of groups	04
9-10	Number of variables	06
11-15	Number of observations in first group	00008
16-20	Number of observations in second group	00007
21-25	Number of observations in third group	00007
26-30	Number of observations in fourth group	00008
⋮		
65-70	Number of observations in twelfth group (if present)	

If there are more than twelve groups in the problem, continue to the second card in the same manner.

Columns	Contents
1- 5	Number of observations in thirteenth group
6-10	Number of observations in fourteenth group

Leading zeros are not required to be keypunched.

Data Cards

Since input data are read into the computer one observation at a time, each row of data in Table 9 is keypunched on a separate card using the format (12F6.0). This format assumes twelve 6-column fields per card.

If there are more than twelve variables in a problem, each row of data is continued on the second and third cards until the last data point is keypunched. However, each row of data must begin on a new card.

Deck Setup

The deck setup is shown in Figure 53.

Sample

The listing of input cards for the sample problem is presented in Figure 54.

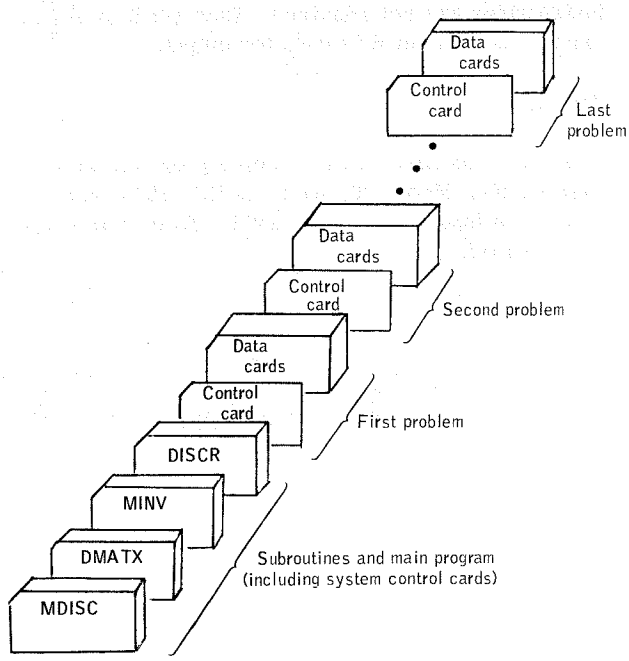


Figure 53. Deck setup (discriminant analysis)

Output

Description

The output of the sample program for discriminant analysis includes:

1. Means of variables in each group
2. Pooled dispersion matrix
3. Common means
4. Generalized Mahalanobis D-square
5. Constant and coefficients of each discriminant function
6. Probability associated with the largest discriminant function evaluated for each observation.

```
/DATA                                                                  10
SAMPLE040600008000070000700008                                         20
     3    10     9     8    24     8                                  30
     4    12     3     8    22     7                                  40
     9     3     2     8     9     8                                  50
    16     2     2     2     7     2                                  60
     5    10     5     8    23     9                                  70
    17     3     2     8     6     3                                  80
     2    10     9     8    29    16                                  90
     7    10     5     8    28    18                                 100
     9    10    27     8    28    16                                 110
    11     7     8     9     8    15                                 120
     8    10     2     8    27    16                                 130
     1     6     8    14    14    13                                 140
     7     8     9     6    18     2                                 150
     7     9     8     2    19     9                                 160
     7    10     5     8    27    17                                 170
     3    11     9    15    20    10                                 180
     9     4    10     7     9     9                                 190
     4    13    10     7    21    15                                 200
     8     5    16    16    16     7                                 210
     6     9    10     5    23    11                                 220
     8    10     5     8    27    16                                 230
    17     3     2     7     6     3                                 240
     3    10     8     8    23     8                                 250
     4    12     3     8    23     7                                 260
     9     3     2     8    21     7                                 270
    15     2     2     2     7     2                                 280
     9    10    26     8    27    16                                 290
     8     9     2     8    26    16                                 300
     7     8     6     9    18     2                                 310
     7    10     5     8    26    16                                 320
```

Figure 54. Input listing (discriminant analysis)

Sample

The output listing for the sample problem is shown in Figure 55.

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize the user with the program modification, the following general rules are supplied in terms of the sample problem.

1. Changes in the dimension statements of the main program, MDISC:
   a. The dimension of array N must be greater than or equal to the number of groups, k. Since there are four groups in the sample problem, the value of k is 4.
   b. The dimension of array CMEAN must be greater than or equal to the number of variables, m. Since there are six variables in the sample problem, the value of m is 6.
   c. The dimension of array XBAR must be greater than or equal to the product of m times k. For the sample problem this product is 24 = 6 x 4.
   d. The dimension of array C must be greater than or equal to the product of (m+1)k. For the sample problem this product is 28 = (6+1)4.
   e. The dimension of array D must be greater than or equal to the product of m times m. For the sample problem this product is 36 = 6 x 6.
   f. The dimension of arrays P and LG must be greater than or equal to the total number of observations in all groups combined, t. For the sample problem this total is 30 = 8 + 7 + 7 + 8.

```
DISCRIMINANT ANALYSIS.....SAMPLE

  NUMBER OF GROUPS          4
  NUMBER OF VARIABLES       6
  SAMPLE SIZES..
          GROUP
            1          8
            2          7
            3          7
            4          8

GROUP  1  MEANS
       7.87500      7.50000      4.62500      7.25000     18.50000      8.87500

GROUP  2  MEANS
       7.14286      8.57143      9.57143      7.85714     20.14285     12.57143

GROUP  3  MEANS
       7.85714      7.85714      8.85714      9.28571     17.42856     10.14286

GROUP  4  MEANS
       7.75000      8.00000      6.75000      7.37500     21.37500      9.25000
```

Figure 55. Output listing (discriminant analysis)

```
POOLED DISPERSION MATRIX

ROW  1
     19.61876     -11.16208     -5.21497     -6.09890    -22.74855     -9.54052

ROW  2
    -11.16208      11.94505      5.61813      1.91758     22.60982     10.66757

ROW  3
     -5.21497       5.61813     39.45938      3.93681     16.23486      9.34546

ROW  4
     -6.09890       1.91758      3.93681      9.83310      4.62156      3.83791

ROW  5
    -22.74855      22.60982     16.23486      4.62156     62.78633     30.18262

ROW  6
     -9.54052      10.66757      9.34546      3.83791     30.18262     29.57480


COMMON MEANS
      7.66667       7.96667       7.33333      7.90000     19.39998     10.13332

GENERALIZED MAHALANOBIS D-SQUARE      12.78063


DISCRIMINANT FUNCTION  1
     CONSTANT    *   COEFFICIENTS
    -28.49431    *        2.63870      2.12205     -0.17167      1.91198      0.58476     -0.40477

DISCRIMINANT FUNCTION  2
     CONSTANT    *   COEFFICIENTS
    -29.21017    *        2.61930      2.25230     -0.04816      1.88319      0.43732     -0.21784

DISCRIMINANT FUNCTION  3
     CONSTANT    *   COEFFICIENTS
    -31.86435    *        2.74450      2.39588     -0.06457      2.13260      0.42619     -0.32718

DISCRIMINANT FUNCTION  4
     CONSTANT    *   COEFFICIENTS
    -30.82028    *        2.71860      2.63937     -0.13352      1.94539      0.71677     -0.48760


EVALUATION OF CLASSIFICATION FUNCTIONS FOR EACH OBSERVATION

GROUP  1
                 PROBABILITY ASSOCIATED WITH          LARGEST
 OBSERVATION     LARGEST DISCRIMINANT FUNCTION      FUNCTION NO.
      1                      0.38065                    4
      2                      0.37045                    1
      3                      0.36261                    1
      4                      0.44190                    1
      5                      0.34454                    1
      6                      0.44215                    3
      7                      0.31787                    2
      8                      0.29274                    2

GROUP  2
                 PROBABILITY ASSOCIATED WITH          LARGEST
 OBSERVATION     LARGEST DISCRIMINANT FUNCTION      FUNCTION NO.
      1                      0.51029                    2
      2                      0.50060                    3
      3                      0.34760                    4
      4                      0.43130                    3
      5                      0.44282                    4
      6                      0.36407                    2
      7                      0.28515                    2

GROUP  3
                 PROBABILITY ASSOCIATED WITH          LARGEST
 OBSERVATION     LARGEST DISCRIMINANT FUNCTION      FUNCTION NO.
      1                      0.67611                    3
      2                      0.46629                    2
      3                      0.54636                    2
      4                      0.66688                    3
      5                      0.30600                    2
      6                      0.33043                    4
      7                      0.39005                    3

GROUP  4
                 PROBABILITY ASSOCIATED WITH          LARGEST
 OBSERVATION     LARGEST DISCRIMINANT FUNCTION      FUNCTION NO.
      1                      0.33727                    4
      2                      0.37475                    1
      3                      0.62340                    4
      4                      0.45697                    1
      5                      0.52175                    2
      6                      0.34061                    4
      7                      0.43135                    4
      8                      0.27849                    1
```

Figure 55. Output listing (discriminant analysis) (Continued)

g. The dimension of array X must be greater than or equal to the total number of data points that is equal to the product of t times m. For the sample this product is 180 = 30 x 6.

2. Changes in the input format statement of the main program, MDISC:

Only the format statement for input data may be changed. Since sample data are either one- or two-digit numbers, rather than using six-column fields as in the sample problem, each row of data may be keypunched in two-column fields, and, if so, the format is changed to (6F2.0). This format assumes six 2-column fields per card, beginning in column 1.

Operating Instructions

The sample program for discriminant analysis is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 30 seconds.

```
C                                                                       MDIS  10
C     ..................................................................MDIS  20
C                                                                       MDIS  30
C        SAMPLE MAIN PROGRAM FOR DISCRIMINANT ANALYSIS - MDISC          MDIS  40
C                                                                       MDIS  50
C        PURPOSE                                                        MDIS  60
C           (1) READ THE PROBLEM PARAMETER CARD AND DATA FOR DISCRIMI-  MDIS  70
C           NANT ANALYSIS, (2) CALL THREE SUBROUTINES TO CALCULATE VARI-MDIS  80
C           ABLE MEANS IN EACH GROUP, POOLED DISPERSION MATRIX, COMMON  MDIS  90
C           MEANS OF VARIABLES, GENERALIZED MAHALANOBIS D SQUARE,       MDIS 100
C           COEFFICIENTS OF DISCRIMINANT FUNCTIONS, AND PROBABILITY     MDIS 110
C           ASSOCIATED WITH LARGEST DISCRIMINANT FUNCTION OF EACH       MDIS 120
C           CASE IN EACH GROUP, AND (3) PRINT THE RESULTS.              MDIS 130
C                                                                       MDIS 140
C        REMARKS                                                        MDIS 150
C           THE NUMBER OF VARIABLES MUST BE GREATER THAN OR EQUAL TO    MDIS 160
C           THE NUMBER OF GROUPS.                                       MDIS 170
C                                                                       MDIS 180
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MDIS 190
C           DMATX                                                       MDIS 200
C           MINV                                                        MDIS 210
C           DISCR                                                       MDIS 220
C                                                                       MDIS 230
C        METHOD                                                         MDIS 240
C           REFER TO '360 COMPUTER PROGRAMS MANUAL', EDITED BY W. J.    MDIS 250
C           DIXON, UCLA, 1964, AND T. W. ANDERSON, 'INTRODUCTION TO     MDIS 260
C           MULTIVARIATE STATISTICAL ANALYSIS', JOHN WILEY AND SONS,    MDIS 270
C           1958, SECTION 6.6-6.8.                                      MDIS 280
C                                                                       MDIS 290
C     ..................................................................MDIS 300
C                                                                       MDIS 310
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MDIS 320
C     NUMBER OF GROUPS, K..                                             MDIS 330
C                                                                       MDIS 340
      DIMENSION N(5)                                                    MDIS 350
C                                                                       MDIS 360
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MDIS 370
C     NUMBER OF VARIABLES, M..                                          MDIS 380
C                                                                       MDIS 390
      DIMENSION CMEAN(10)                                               MDIS 400
C                                                                       MDIS 410
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MDIS 420
C     PRODUCT OF M*K..                                                  MDIS 430
C                                                                       MDIS 440
      DIMENSION XBAR(50)                                                MDIS 450
C                                                                       MDIS 460
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MDIS 470
C     PRODUCT OF (M+1)*K..                                              MDIS 480
C                                                                       MDIS 490
      DIMENSION C(55)                                                   MDIS 500
C                                                                       MDIS 510
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MDIS 520
C     PRODUCT OF M*M..                                                  MDIS 530
C                                                                       MDIS 540
      DIMENSION D(100)                                                  MDIS 550
C                                                                       MDIS 560
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO THE     MDIS 570
C     TOTAL OF SAMPLE SIZES OF K GROUPS COMBINED, T (T = N(1)+N(2)+...  MDIS 580
C     +N(K))..                                                          MDIS 590
C                                                                       MDIS 600
      DIMENSION P(250),LG(250)                                          MDIS 610
C                                                                       MDIS 620
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      MDIS 630
C     TOTAL DATA POINTS WHICH IS EQUAL TO THE PRODUCT OF T*M..          MDIS 640
C                                                                       MDIS 650
      DIMENSION X(2500)                                                 MDIS 660
C                                                                       MDIS 670
C     ..................................................................MDIS 680
C                                                                       MDIS 690
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  MDIS 700
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      MDIS 710
C        STATEMENT WHICH FOLLOWS.                                       MDIS 720
C                                                                       MDIS 730
C     DOUBLE PRECISION CMEAN,XBAR,D,DET,C,V,P                           MDIS 740
C                                                                       MDIS 750
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    MDIS 760
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      MDIS 770
C        ROUTINE.                                                       MDIS 780
C                                                                       MDIS 790
C        ...............................................................MDIS 800
C                                                                       MDIS 810
    1 FORMAT(A4,A2,2I2,12I5/(14I5))                                     MDIS 820
    2 FORMAT(27H1DISCRIMINANT ANALYSIS.....,A4,A2/19H0  NUMBER OF GROUPSMDIS 830
     1,7X,I3/22H   NUMBER OF VARIABLES,I7/17H   SAMPLE SIZES../12X,5HGROMDIS 840
     2UP)                                                               MDIS 850
    3 FORMAT(12X,I3,8X,I4)                                              MDIS 860
    4 FORMAT(1H0)                                                       MDIS 870
    5 FORMAT(12F6.0)                                                    MDIS 880
    6 FORMAT(6H0GROUP,I3,7H  MEANS/(8F15.5))                            MDIS 890
    7 FORMAT(1H0//25H POOLED DISPERSION MATRIX)                         MDIS 900
    8 FORMAT(4H0ROW,I3/(8F15.5))                                        MDIS 910
    9 FORMAT(1H0//13H COMMON MEANS/(8F15.5))                            MDIS 920
   10 FORMAT(1H///33H GENERALIZED MAHALANOBIS D-SQUARE,F15.5//)         MDIS 930
   11 FORMAT(22H0DISCRIMINANT FUNCTION,I3/1H ,6X,27HCONSTANT    *   COEFFMDIS 940
     1ICIENTS/1H F14.5,7H    *    ,7F14.5/(22X,7F14.5))                 MDIS 950
   12 FORMAT(1H0//60H EVALUATION OF CLASSIFICATION FUNCTIONS FOR EACH OBMDIS 960
     1SERVATION)                                                        MDIS 970
   13 FORMAT(6H0GROUP,I3/19X,27HPROBABILITY ASSOCIATED WITH,11X,7HLARGESMDIS 980
     1T/13H  OBSERVATION,5X,29HLARGEST DISCRIMINANT FUNCTION,8X,12HFUNCTMDIS 990
     2ION NO.)                                                          MDIS1000
   14 FORMAT(1H ,I7,20X,F8.5,20X,I6)                                    MDIS1010
C                                                                       MDIS1020
C     ..................................................................MDIS1030
C                                                                       MDIS1040
C     READ PROBLEM PARAMETER CARD                                       MDIS1050
C                                                                       MDIS1060
  100 READ (5,1) PR,PRI,K,M,(N(I),I=1,K)                                MDIS1070
C        PR.......PROBLEM NUMBER (MAY BE ALPHAMERIC)                    MDIS1080
C        PRI......PROBLEM NUMBER (CONTINUED)                            MDIS1090
C        K........NUMBER OF GROUPS                                      MDIS1100
```

```
C        M........NUMBER OF VARIABLES                                   MDIS1110
C        N........VECTOR OF LENGTH K CONTAINING SAMPLE SIZES            MDIS1120
C                                                                       MDIS1130
      WRITE (6,2) PR,PR1,K,M                                            MDIS1140
      DO 110 I=1,K                                                      MDIS1150
  110 WRITE (6,3) I,N(I)                                                MDIS1160
      WRITE (6,4)                                                       MDIS1170
C                                                                       MDIS1180
C     READ DATA                                                         MDIS1190
C                                                                       MDIS1200
      L=0                                                               MDIS1210
      DO 130 I=1,K                                                      MDIS1220
      N1=N(I)                                                           MDIS1230
      DO 120 J=1,N1                                                     MDIS1240
      READ (5,5) (CMEAN(IJ),IJ=1,M)                                     MDIS1250
      L=L+1                                                             MDIS1260
      N2=L-N1                                                           MDIS1270
      DO 120 IJ=1,M                                                     MDIS1280
      N2=N2+N1                                                          MDIS1290
  120 X(N2)=CMEAN(IJ)                                                   MDIS1300
  130 L=N2                                                              MDIS1310
C                                                                       MDIS1320
      CALL DMATX (K,M,N,X,XBAR,D,CMEAN)                                 MDIS1330
C                                                                       MDIS1340
C     PRINT MEANS AND POOLED DISPERSION MATRIX                          MDIS1350
C                                                                       MDIS1360
      L=0                                                               MDIS1370
      DO 150 I=1,K                                                      MDIS1380
      DO 140 J=1,M                                                      MDIS1390
      L=L+1                                                             MDIS1400
  140 CMEAN(J)=XBAR(L)                                                  MDIS1410
  150 WRITE (6,6) I,(CMEAN(J),J=1,M)                                    MDIS1420
      WRITE (6,7)                                                       MDIS1430
      DO 170 I=1,M                                                      MDIS1440
      L=I-M                                                             MDIS1450
      DO 160 J=1,M                                                      MDIS1460
      L=L+M                                                             MDIS1470
  160 CMEAN(J)=D(L)                                                     MDIS1480
  170 WRITE (6,8) I,(CMEAN(J),J=1,M)                                    MDIS1490
C                                                                       MDIS1500
      CALL MINV (D,M,DET,CMEAN,C)                                       MDIS1510
C                                                                       MDIS1520
      CALL DISCR (K,M,N,X,XBAR,D,CMEAN,V,C,P,LG)                        MDIS1530
C                                                                       MDIS1540
C     PRINT COMMON MEANS                                                MDIS1550
C                                                                       MDIS1560
      WRITE (6,9) (CMEAN(I),I=1,M)                                      MDIS1570
C                                                                       MDIS1580
C     PRINT GENERALIZED MAHALANOBIS D-SQUARE                            MDIS1590
C                                                                       MDIS1600
      WRITE (6,10) V                                                    MDIS1610
C                                                                       MDIS1620
C     PRINT CONSTANTS AND COEFFICIENTS OF DISCRIMINANT FUNCTIONS        MDIS1630
C                                                                       MDIS1640
      N1=1                                                              MDIS1650
      N2=M+1                                                            MDIS1660
      DO 180 I=1,K                                                      MDIS1670
      WRITE (6,11) I,(C(J),J=N1,N2)                                     MDIS1680
      N1=N1+(M+1)                                                       MDIS1690
  180 N2=N2+(M+1)                                                       MDIS1700
C                                                                       MDIS1710
C     PRINT EVALUATION OF CALSSIFICATION FUNCTIONS FOR EACH OBSERVATION MDIS1720
C                                                                       MDIS1730
      WRITE (6,12)                                                      MDIS1740
      N1=1                                                              MDIS1750
      N2=N(1)                                                           MDIS1760
      DO 210 I=1,K                                                      MDIS1770
      WRITE (6,13) I                                                    MDIS1780
      L=0                                                               MDIS1790
      DO 190 J=N1,N2                                                    MDIS1800
      L=L+1                                                             MDIS1810
  190 WRITE (6,14) L,P(J),LG(J)                                         MDIS1820
      IF(I-K) 200, 100, 100                                             MDIS1830
  200 N1=N1+N(I)                                                        MDIS1840
      N2=N2+N(I+1)                                                      MDIS1850
  210 CONTINUE                                                          MDIS1860
      END                                                               MDIS1870
```

FACTOR ANALYSIS

Problem Description

A principal component solution and the varimax rotation of the factor matrix are performed. Principal component analysis is used to determine the minimum number of independent dimensions needed to account for most of the variance in the original set of variables. The varimax rotation is used to simplify columns (factors) rather than rows (variables) of the factor matrix.

The sample problem for factor analysis consists of 23 observations with nine variables as presented in Table 10. In order to keep the number of independent dimensions as small as possible, only those eigenvalues (of correlation coefficients) greater than or equal to 1.0 are retained in the analysis.

Table 10. Sample Data for Factor Analysis

Observation	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9
1	7	7	9	7	15	36	60	15	24
2	13	18	25	15	13	35	61	18	30
3	9	18	24	23	12	43	62	14	31
4	7	13	25	36	11	12	63	26	32
5	6	8	20	7	15	46	18	28	15
6	10	12	30	11	10	42	27	12	17
7	7	6	11	7	15	35	60	20	25
8	16	19	25	16	13	30	64	20	30
9	9	22	26	24	13	40	66	15	32
10	8	15	26	30	13	10	66	25	34
11	8	10	20	8	17	40	20	30	18
12	9	12	28	11	8	45	30	15	19
13	11	17	21	30	10	45	60	17	30
14	9	16	26	27	14	31	59	19	17
15	10	15	24	18	12	29	48	18	26
16	11	11	30	19	19	26	57	20	30
17	16	9	16	20	18	31	60	21	17
18	9	8	19	14	16	33	67	9	19
19	7	18	22	9	15	37	62	11	20
20	8	11	23	18	9	36	61	22	24
21	6	6	27	23	7	40	55	24	31
22	10	9	26	26	10	37	57	27	29
23	8	10	26	15	11	42	59	20	28

Program

Description

The factor analysis program consists of the main routine named FACTO, a special input subroutine named DATA, and five subroutines from the Scientific Subroutine Package: CORRE, EIGEN, TRACE, LOAD, and VARMX.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to 35 variables
2. Up to 99,999 observations
3. (12F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions it is not necessary to modify the sample program. However, if there are more than 35 variables, dimension statements in the sample main program must be modified to handle this particular problem. Similarly, if input data cards are prepared using a different format, the input format statement in the input subroutine, DATA, must be modified. The general rules for program modification are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, FACTO. This card is prepared as follows:

Columns	Contents	For Sample Problem
1- 6	Problem number (may be alphameric)	SAMPLE
7-11	Number of observations	00023
12-13	Number of variables	09
14-19	Value used to limit the number of eigenvalues of correlation coefficients. Only those eigenvalues greater than or equal to this value are retained in the analysis. (A decimal point must be specified.)	0001.0

Leading zeros are not required to be keypunched.

Data Cards

Since input data are read into the computer one observation at a time, each row of data in Table 10 is keypunched on a separate card using the format (12F6.0). This format assumes twelve 6-column fields per card.

If there are more than twelve variables in a problem, each row of data is continued on the second and third cards until the last data point is keypunched. However, each row of data must begin on a new card.

Deck Setup

The deck setup is shown in Figure 56.

Sample

The listing of input cards for the sample problem is presented in Figure 57.

Output

Description

The output of the sample program for factor analysis includes:

1. Means
2. Standard deviations
3. Correlation coefficients
4. Eigenvalues
5. Cumulative percentage of eigenvalues
6. Eigenvectors
7. Factor matrix
8. Variance of the factor matrix for each iteration cycle

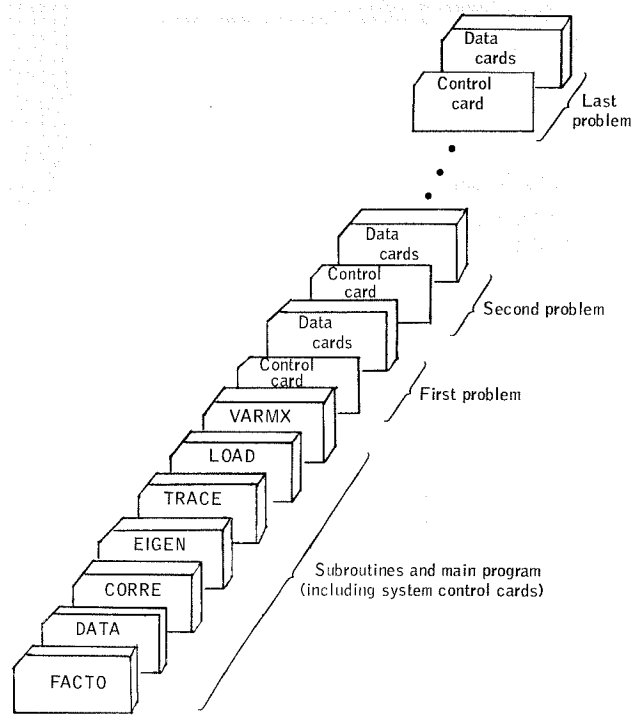


Figure 56. Deck setup (factor analysis)

9. Rotated factor matrix
10. Check on communalities

Sample

The output listing for the sample problem is shown in Figure 58.

Program Modification

Program capacity can be increased or decreased by making changes in dimension statements. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize the user with the program modification, the

```
/DATA                                                                   10
SAMPLE00023090001.0                                                     20
     7     7     9     7    15    36    60    15    24                30
    13    18    25    15    13    35    61    18    30                40
     9    18    24    23    12    43    62    14    31                50
     7    13    25    36    11    12    63    26    32                60
     6     8    20     7    15    46    18    28    15                70
    10    12    30    11    10    42    27    12    17                80
     7     6    11     7    15    35    60    20    25                90
    16    19    25    16    13    30    64    20    30               100
     9    22    26    24    13    40    66    15    32               110
     8    15    26    30    13    10    66    25    34               120
     8    10    20     8    17    40    20    30    18               130
     9    12    28    11     8    45    30    15    19               140
    11    17    21    30    10    45    60    17    30               150
     9    16    26    27    14    31    59    19    17               160
    10    15    24    18    12    29    48    18    26               170
    11    11    30    19    19    26    57    20    30               180
    16     9    16    20    18    31    60    21    17               190
     9     8    19    14    16    33    67     9    19               200
     7    18    22     9    15    37    62    11    20               210
     8    11    23    18     9    36    61    22    24               220
     6     6    27    23     7    40    55    24    31               230
    10     9    26    26    10    37    57    27    29               240
     8    10    26    15    11    42    59    20    28               250
```

Figure 57. Input listing (factor analysis)

```
FACTOR ANALYSIS.....SAMPLE

  NO. OF CASES          23
  NO. OF VARIABLES       9

MEANS
      9.30435      12.60870      73.00000      14.00000      12.86957      34.82608      [illegible]      [illegible]
     25.13043

STANDARD DEVIATIONS
      2.70417       4.59978       5.33427       8.33393       3.13741       9.29149      14.87624      [illegible]
      6.09249

CORRELATION COEFFICIENTS

ROW  1
    1.00000       0.34987       0.11975       0.12102       0.21917      -0.09549       0.20901      -0.12908       0.05818

ROW  2
    0.34987       1.00000       0.41311       0.35572      -0.08243      -0.09100       0.29622      -0.32044       0.35387

ROW  3
    0.11975       0.41311       1.00000       0.41512      -0.43179      -0.08346      -0.10252       0.03215       0.27833

ROW  4
    0.12102       0.35572       0.41512       1.00000      -0.31288      -0.50365       0.49856       0.22539       0.59890

ROW  5
    0.21917      -0.08243      -0.43179      -0.31288       1.00000      -0.23000       0.03310      -0.00475      -0.30341

ROW  6
   -0.09549      -0.09100      -0.08346      -0.50365      -0.23000       1.00000      -0.44520      -0.25441      -0.37456

ROW  7
    0.20901       0.29622      -0.10252       0.49856       0.03310      -0.44520       1.00000      -0.28050       0.60124

ROW  8
   -0.12908      -0.32044       0.03215       0.22539      -0.00475      -0.25441      -0.28050       1.00000       0.13516

ROW  9
    0.05818       0.35387       0.27833       0.59890      -0.30341      -0.37456       0.60124       0.13516       1.00000

EIGENVALUES
    2.94988       1.64368       1.55514       1.06579

CUMULATIVE PERCENTAGE OF EIGENVALUES
    0.32776       0.51040       0.68319       0.80161

EIGENVECTORS

VECTOR  1
    0.16437       0.34836       0.28797       0.49661      -0.16806      -0.32922       0.39935       0.01287       0.47518

VECTOR  2
    0.34837       0.06552      -0.44647      -0.11893       0.61210      -0.26428       0.38860      -0.24845      -0.06014

VECTOR  3
   -0.29899      -0.46825      -0.23534       0.17377       0.14468      -0.43545       0.01891       0.61587       0.12476

VECTOR  4
    0.54441       0.16909       0.38288       0.04163       0.30537      -0.16163      -0.43410       0.40283      -0.23789

FACTOR MATRIX (  4 FACTORS)

VARIABLE  1
    0.28232       0.44663      -0.37286       0.56203

VARIABLE  2
    0.59831       0.08400      -0.58394       0.17457

VARIABLE  3
    0.49460      -0.57240      -0.29348       0.39528

VARIABLE  4
    0.85293      -0.15248       0.21671       0.04297

VARIABLE  5
   -0.28865       0.78475       0.18043       0.31525

VARIABLE  6
   -0.56544      -0.33882      -0.54303      -0.16686

VARIABLE  7
    0.68590       0.49821       0.02345      -0.44816

VARIABLE  8
    0.02711      -0.31853       0.76802       0.41587

VARIABLE  9
    0.81614      -0.07710       0.15551      -0.24559

ITERATION        VARIANCES
  CYCLE
    0            0.211288
    1            0.336137
    2            0.397022
    3            0.403005
    4            0.405177
    5            0.405528
    6            0.405581
    7            0.405587
    8            0.405588
    9            0.405587
   10            0.405587
   11            0.405587
   12            0.405587

ROTATED FACTOR MATRIX (  4 FACTORS)

VARIABLE  1
    0.05498       0.07183      -0.05578       0.85017

VARIABLE  2
    0.29329      -0.39653      -0.35581       0.60549

VARIABLE  3
    0.05114      -0.82493       0.15068       0.32984

VARIABLE  4
    0.74040      -0.41401       0.24579       0.13972

VARIABLE  5
   -0.09091       0.80662       0.13525       0.39229
```

Figure 58. Output listing (factor analysis)

```
VARIABLE  6
   -0.68286    -0.21579    -0.44983    -0.20503

VARIABLE  7
    0.86997     0.18299    -0.34918     0.08830

VARIABLE  8
    0.03602    -0.05500     0.91375    -0.15962

VARIABLE  9
    0.80531    -0.32759     0.00994    -0.02380

CHECK ON COMMUNALITIES

VARIABLE        ORIGINAL          FINAL        DIFFERENCE
    1            0.73409         0.73408         0.00001
    2            0.73649         0.73647         0.00001
    3            0.81464         0.81463         0.00001
    4            0.79955         0.79954         0.00001
    5            0.83109         0.83108         0.00001
    6            0.75725         0.75724         0.00001
    7            0.92006         0.92005         0.00001
    8            0.86476         0.86475         0.00001
    9            0.75652         0.75651         0.00001

1F0031217
```

Figure 58. Output listing (factor analysis) (Continued)

following general rules are supplied in terms of the sample problem.

1. Changes in the dimension statements of the main program, FACTO:
   a. The dimension of arrays B, D, S, T, and XBAR must be greater than or equal to the number of variables, m. Since there are nine variables in the sample problem, the value of m is 9.
   b. The dimension of array V must be greater than or equal to the product of m times m. For the sample problem this product is $81 = 9 \times 9$.
   c. The dimension of array R must be greater than $\frac{(m+1)m}{2}$. For the sample problem, this number is $45 = \frac{(9+1)9}{2}$.
2. Changes in the input format statement of the special input subroutine, DATA:

   Only the format statement for input data may be changed. Since sample data are either one- or two-digit numbers, rather than using six-column fields as in the sample problem, each row of data may be keypunched in two-column fields, and, if so, the format is changed to (9F2.0). This format assumes nine 2-column fields per card, beginning in column 1.

   The special input subroutine, DATA, is normally written by the user to handle different formats for different problems. The user may modify this subroutine to perform testing of input data, transformation of data, and so on.

Operating Instructions

The sample program for factor analysis is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Error Messages

If the number of factors to be rotated is one or zero, the following message will be printed:

ONLY____FACTOR RETAINED. NO ROTATION.

The program skips rotation and goes to the next problem if it is present.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 50 seconds.

```
C                                                                       FCTO  10
C     ..................................................................FCTO  20
C                                                                       FCTO  30
C        SAMPLE MAIN PROGRAM FOR FACTOR ANALYSIS - FACTO                FCTO  40
C                                                                       FCTO  50
C        PURPOSE                                                        FCTO  60
C           (1) READ THE PROBLEM PARAMETER CARD, (2) CALL FIVE SUBROU-  FCTO  70
C           TINES TO PERFORM A PRINCIPAL COMPONENT SOLUTION AND THE     FCTO  80
C           VARIMAX ROTATION OF A FACTOR MATRIX, AND (3) PRINT THE      FCTO  90
C           RESULTS.                                                    FCTO 100
C                                                                       FCTO 110
C        REMARKS                                                        FCTO 120
C           NONE                                                        FCTO 130
C                                                                       FCTO 140
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  FCTO 150
C           CORRE  (WHICH, IN TURN, CALLS THE SUBROUTINE NAMED DATA.)   FCTO 160
C           EIGEN                                                       FCTO 170
C           TRACE                                                       FCTO 180
C           LOAD                                                        FCTO 190
C           VARMX                                                       FCTO 200
C                                                                       FCTO 210
C        METHOD                                                         FCTO 220
C           REFER TO 'BMD COMPUTER PROGRAMS MANUAL', EDITED BY W. J.    FCTO 230
C           DIXON, UCLA, 1964.                                          FCTO 240
C                                                                       FCTO 250
C     ..................................................................FCTO 260
C                                                                       FCTO 270
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN OR EQUAL TO THE     FCTO 280
C     NUMBER OF VARIABLES, M..                                          FCTO 290
C                                                                       FCTO 300
      DIMENSION B(35),D(35),S(35),T(35),XBAR(35)                        FCTO 310
C                                                                       FCTO 320
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      FCTO 330
C     PRODUCT OF M*M..                                                  FCTO 340
C                                                                       FCTO 350
      DIMENSION V(1225)                                                 FCTO 360
C                                                                       FCTO 370
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO          FCTO 380
C     (M+1)*M/2..                                                       FCTO 390
C                                                                       FCTO 400
      DIMENSION R(630)                                                  FCTO 410
C                                                                       FCTO 420
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO 51..     FCTO 430
C                                                                       FCTO 440
      DIMENSION TV(51)                                                  FCTO 450
C                                                                       FCTO 460
C     ..................................................................FCTO 470
C                                                                       FCTO 480
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  FCTO 490
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      FCTO 500
C        STATEMENT WHICH FOLLOWS.                                       FCTO 510
C                                                                       FCTO 520
C     DOUBLE PRECISION XBAR,S,V,R,D,B,T,TV                              FCTO 530
C                                                                       FCTO 540
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    FCTO 550
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      FCTO 560
C        ROUTINE.                                                       FCTO 570
C                                                                       FCTO 580
C        ..............................................................FCTO 590
C                                                                       FCTO 600
    1 FORMAT(21H1FACTOR ANALYSIS.....A4,A2//3X,12HNO. OF CASES,4X,I6/3X,FCTO 610
     116HNO. OF VARIABLES,I6/)                                          FCTO 620
    2 FORMAT(6HOMEANS/(8F15.5))                                         FCTO 630
    3 FORMAT(2CHOSTANDARD DEVIATIONS/(8F15.5))                          FCTO 640
    4 FORMAT(25HOCORRELATION COEFFICIENTS)                              FCTO 650
    5 FORMAT(4HOROW I3/(10F12.5))                                       FCTO 660
    6 FORMAT(1H0/12H EIGENVALUES/(10F12.5))                             FCTO 670
    7 FORMAT(37H0CUMULATIVE PERCENTAGE OF EIGENVALUES/(10F12.5))        FCTO 680
    8 FORMAT(1H0/13H EIGENVECTORS)                                      FCTO 690
    9 FORMAT(7HOVECTOR I3/(10F12.5))                                    FCTO 700
   10 FORMAT(1H0/16H FACTOR MATRIX (,I3,9H FACTORS))                    FCTO 710
   11 FORMAT(9HOVARIABLE I3/(10F12.5))                                  FCTO 720
   12 FORMAT(1H0/1CH ITERATION,7X,9HVARIANCES/8H   CYCLE)               FCTO 730
   13 FORMAT(I6,F2C.6)                                                  FCTO 740
   14 FORMAT(1H0/24H ROTATED FACTOR MATRIX (I3,9H FACTORS))             FCTO 750
   15 FORMAT(9HOVARIABLE I3/(10F12.5))                                  FCTO 760
   16 FORMAT(1H0/23H CHECK ON COMMUNALITIES//9H VARIABLE,7X,8HORIGINAL,1FCTO 770
     12X,5HFINAL,10X,10HDIFFERENCE)                                     FCTO 780
   17 FORMAT(I6,3F18.5)                                                 FCTO 790
   18 FORMAT(A4,A2,I5,I2,F6.C)                                          FCTO 800
   19 FORMAT(5HOONLY,I2,30H FACTOR RETAINED.  NO ROTATION)              FCTO 810
C                                                                       FCTO 820
C     ..................................................................FCTO 830
C                                                                       FCTO 840
C     READ PROBLEM PARAMETER CARD                                       FCTO 850
C                                                                       FCTO 860
  100 READ (5,18) PR,PR1,N,M,CON                                        FCTO 870
C        PR.........PROBLEM NUMBER (MAY BE ALPHAMERIC)                  FCTO 880
C        PR1........PROBLEM NUMBER (CONTINUED)                          FCTO 890
C        N..........NUMBER OF CASES                                     FCTO 900
C        M..........NUMBER OF VARIABLES                                 FCTO 910
C        CON........CONSTANT USED TO DECIDE HOW MANY EIGENVALUES        FCTO 920
C                   TO RETAIN                                           FCTO 930
C                                                                       FCTO 940
      WRITE (6,1) PR,PR1,N,M                                            FCTO 950
C                                                                       FCTO 960
```

```
      IO=0
      X=0.0
C
      CALL CORRE (N,M,IO,X,XBAR,S,V,R,D,B,T)
C
C     PRINT MEANS
C
      WRITE (6,2) (XBAR(J),J=1,M)
C
C     PRINT STANDARD DEVIATIONS
C
      WRITE (6,3) (S(J),J=1,M)
C
C     PRINT CORRELATION COEFFICIENTS
C
      WRITE (6,4)
      DO 120 I=1,M
      DO 110 J=1,M
      IF(I-J) 102, 104, 104
  102 L=I+(J*J-J)/2
      GO TO 110
  104 L=J+(I*I-I)/2
  110 D(J)=R(L)
  120 WRITE (6,5) I,(D(J),J=1,M)
C
      MV=0
      CALL EIGEN (R,V,M,MV)
C
      CALL TRACE (M,R,CON,K,D)
C
C     PRINT EIGENVALUES
C
      DO 130 I=1,K
      L=I+(I*I-I)/2
  130 S(I)=R(L)
      WRITE (6,6) (S(J),J=1,K)
C
C     PRINT CUMULATIVE PERCENTAGE OF EIGENVALUES
C
      WRITE (6,7) (D(J),J=1,K)
C
C     PRINT EIGENVECTORS
C
      WRITE (6,8)
      L=0
      DO 150 J=1,K
      DO 140 I=1,M
      L=L+1
  140 D(I)=V(L)
  150 WRITE (6,9) J,(D(I),I=1,M)
C
      CALL LOAD (M,K,R,V)
C
C     PRINT FACTOR MATRIX
C
      WRITE (6,10) K
      DO 180 I=1,M
      DO 170 J=1,K
      L=M*(J-1)+I
  170 D(J)=V(L)
  180 WRITE (6,11) I,(D(J),J=1,K)
C
      IF(K-1) 185, 185, 188
  185 WRITE (6,19) K
      GO TO 100
C
  188 CALL VARMX (M,K,V,NC,TV,B,T,D)
C
C     PRINT VARIANCES
C
      NV=NC+1
      WRITE (6,12)
      DO 190 I=1,NV
      NC=I-1
  190 WRITE (6,13) NC,TV(I)
C
C     PRINT ROTATED FACTOR MATRIX
C
      WRITE (6,14) K
      DO 220 I=1,M
      DO 210 J=1,K
      L=M*(J-1)+I
  210 S(J)=V(L)
  220 WRITE (6,15) I,(S(J),J=1,K)
C
C     PRINT COMMUNALITIES
C
      WRITE (6,16)
      DO 230 I=1,M
  230 WRITE (6,17) I,B(I),T(I),D(I)
      GO TO 100
      END
```

KOLMOGOROV - SMIRNOV TEST

Problem Description

This program is concerned with the problem of determining (a) from what probability density function a particular sample is drawn, or (b) whether two different samples were drawn from the same population. In other words, in the one sample case, the actual distribution function of the sample is compared with one or more theoretical distribution functions, which may be normal, and/or exponential, and/or Cauchy, and/or uniform, and/or a user-specified distribution. In the two-sample case, the pair of sample (actual) distribution functions are compared with one another.

From the above comparisons, a statistic is derived. In the one sample case, this statistic evaluates the probability that the statistic will be as great as or greater than its current value, if the hypothesis that the actual (sample) and the theoretical distribution functions are equal is correct. In other words, if the probability is determined to be 0.40, for example, rejecting the hypothesis of equality of the distribution functions will be an incorrect action 40 times out of 100. Similarly, in the two-sample case, the hypothesis being tested is the equality of the two actual (sample) distribution functions.

This probability is calculated using asymptotic formulas. This means that the sample sizes involved should be large. Sizes greater than 100 are suggested by the literature. In this connection, the remarks given under the topic "Subroutine SMIRN" should be considered.

Note also that added problems arise when, in the one sample case, the parameters of the continuous distribution in question are estimated from the sample. Lilliefors discusses these problems (see reference given in KOLMO description).

Program

Description

This program consists of the main routine KOLM, and four subroutines from the Scientific Subroutine Package: KOLMO, KOLM2, SMIRN, and NDTR.

Capacity

This program allows up to two samples each with 500 or fewer observations to be examined. If the user desires to modify this program to handle more observations, the instructions given below under "Program Modification" should be followed.

Input

After two initializing cards (1, 2 below) each job consists of two control cards and the data cards (3-5 below).

1. Initializing card (minus signs in cc 1-4)
2. Names of distribution functions (as supplied, these are the normal, exponential, Cauchy, uniform, and user's pdf). These names are punched in the order given, centrally located in 16 digit fields on one card; that is, "normal" is punched in cc 1-16, "exponential" in cc 17-32, and so on.
3. Job control card (minus signs in cc 1-4)
4. Program control card

Each job requires one program control card, defined below:

Columns	Contents	For Sample Problems
1 - 20	Title (alphameric)	Uniform test (Job 1) Uniform-Gauss Test (job 2)
21	1 -- one-sample test 2 -- two-sample test	1 (job 1) 2 (job 2)
22	Leave blank for one-sample test. 0 -- Read both samples (two-sample tests). 1 -- Read only one sample and compare it with the first sample read on preceding job.	0 (job 1) 1 (job 2)
23	0 -- Do not print the sample(s). 1 -- Print the sorted sample(s). (F10.3, ten per line)	0 (job 1) 1 (job 2)

(The rest of this control card pertains to a one-sample test.)

Columns	Contents	For Sample Problems
24	0 -- Do not compare with normal pdf. 1 -- Compare with normal pdf.	1 (job 1)
25 - 29	u -- mean of the normal pdf	0.5 (job 1)
30 - 34	s -- standard deviation of the normal pdf	0.5 (job 1)
35	0 -- Do not compare with exponential pdf. 1 -- Compare with exponential pdf.	1 (job 1)
36 - 40	u -- mean of the exponential pdf	0.5 (job 1)
41 - 45	s -- standard deviation of the exponential pdf	1.0 (job 1)
46	0 -- Do not compare with Cauchy pdf. 1 -- Compare with Cauchy pdf.	1 (job 1)
47 - 51	u -- median of the Cauchy pdf	0.5 (job 1)
52 - 56	s -- u-s is the first quartile of the Cauchy pdf	1.0 (job 1)
57	0 -- Do not compare with uniform pdf. 1 -- Compare with uniform pdf.	1 (job 1)
58 - 62	u -- left endpoint of the uniform pdf	0 (job 1)
63 - 67	s -- right endpoint of uniform pdf	1.0 (job 1)
68	0 -- Do not compare with user's pdf. 1 -- Compare with user-specified pdf.	0 (job 1)
69 - 73	u } -- Parameters for	0 (job 1)
74 - 78	s } -- user-specified pdf	0 (job 1)

u and s are described fully in "Description of Parameters" under subroutine KOLMO, and are read using Format F 5.0; decimal points will override the format specification.

5. Data is read into the computer one sample at a time. The reading of a sample is terminated by a data element of 999999. New samples must begin on a new card. Data elements are punched on cards using format F6.0, which assumes twelve 6-column fields per card; decimal points on the card override the format specification. Since the routines KOLMO and KOLM2 sort the samples, no particular order within a sample is necessary.

Deck Setup

See figure 59.

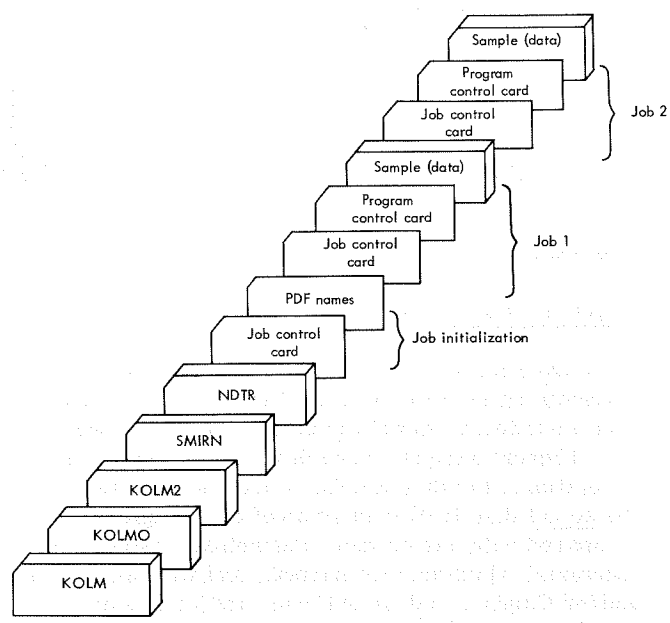


Figure 59. Deck setup (Kolmogorov - Smirnov test)

Sample Data

See Figure 60.

```
/DATA                                                                   10
----                                                                    20
    NORMAL        EXPONENTIAL        CAUCHY        UNIFORM        USER'  30
----                                                                    40
UNIFORM TEST        1001000.5000.51000.5000011000000001                 50
 0.377 0.260 0.172 0.688 0.581 0.290 0.514 0.472 0.204 0.976 0.018 0.326  60
 0.795 0.837 0.870 0.686 0.288 0.555 0.737 0.427 0.931 0.745 0.092 0.843  70
 0.231 0.806 0.753 0.263 0.804 0.458 0.508 0.928 0.994 0.608 0.702 0.743  80
 0.005 0.951 0.664 0.425 0.570 0.596 0.444 0.302 0.817 0.183 0.746 0.833  90
 0.282 0.201 0.662 0.167 0.043 0.750 0.117 0.953 0.665 0.411 0.477 0.164 100
 0.692 0.683 0.867 0.054 0.518 0.624 0.083 0.882 0.540 0.301 0.953 0.006 110
 0.458 0.694 0.041 0.995 0.604 0.666 0.561 0.367 0.156 0.630 0.377 0.589 120
 0.139 0.536 0.963 0.956 0.068 0.801 0.199 0.985 0.113 0.816 0.880 0.931 130
 0.670 0.640 0.805 0.073 0.196 0.516 0.336 0.371 0.197 0.843 0.288 0.139 140
 0.242 0.200 0.025 0.349 0.870 0.080 0.652 0.190 0.275 0.939 0.161 0.514 150
 0.636 0.190 0.416 0.786 0.973 0.767 0.845 0.168 0.400 0.888 0.726 0.365 160
 0.652 0.632 0.923 0.844 0.761 0.969 0.965 0.073 0.751 0.851 0.340 0.383 170
 0.243 0.008 0.860 0.093 0.816 0.058 0.006 0.515 0.033 0.565 0.093 0.470 180
 0.982 0.666 0.154 0.933 0.215 0.890 0.409 0.441 0.963 0.810 0.195 0.876 190
 0.501 0.123 0.228 0.264 0.531 0.810 0.083 0.202 0.469 0.996 0.752 0.545 200
 0.503 0.117 0.170 0.972 0.298 0.042 0.574 0.065 0.225 0.766 0.570 0.520 210
 0.996 0.292 0.790 0.111 0.556 0.337 0.012 0.042 0.143 0.482 0.607 0.302 220
 0.353 0.397 0.206 0.662 0.119 0.754 0.450 0.918 0.453 0.463 0.699 0.022 230
 0.842 0.859 0.577 0.725 0.163 0.450 0.232 0.349 0.000 0.864 0.181 0.311 240
 0.236 0.622 0.607 0.042 0.787 0.348 0.006 0.904 0.365 0.053 0.037 0.745 250
 0.136 0.113 0.455 0.708 0.158 0.572 0.012 0.928 0.455 0.381 0.193 0.728 260
 0.629 0.220 0.657 0.962 0.860 0.501 0.268 0.098 0.181 0.203 0.588 0.701 270
 0.909 0.148 0.708 0.909 0.088 0.345 0.277 0.556 0.840 0.033 0.639 0.539 280
 0.482 0.041 0.907 0.077 0.299 0.097 0.892 0.478 0.835 0.707 0.733 0.029 290
 0.581 0.224 0.112 0.659 0.945 0.741 0.940 0.969 0.360 0.434 0.365 0.285 300
 0.422 0.967 0.005 0.328 0.924 0.595 0.253 0.157 0.668 0.594 0.554 0.984 310
 0.913 0.622 0.516 0.502 0.364 0.667 0.724 0.344 0.546 0.178 0.151 0.302 320
 0.457 0.021 0.019 0.923 0.365 0.882 0.010 0.121 0.637 0.734 0.671 0.416 330
 0.459 0.009 0.919 0.434 0.331 0.079 0.500 0.284 0.209 0.694 0.283 0.454 340
 0.178 0.978 0.272 0.827 0.512 0.634 0.195 0.462 0.019 0.956 0.560 0.761 350
 0.524 0.294 0.047 0.634 0.382 0.591 0.103 0.303 0.889 0.607 0.638 0.367 360
 0.463 0.471 0.664 0.742 0.476 0.178 0.785 0.113 0.610 0.646 0.390 0.520 370
 0.611 0.988 0.431 0.699 0.312 0.580 0.672 0.810 0.814 0.597 0.256 0.164 380
 0.679 0.603 0.504 0.595 0.033 0.846 0.783 0.079 0.430 0.868 0.343 0.244 390
 0.376 0.062 0.990 0.381 0.371 0.801 0.467 0.592 0.348 0.759 0.422 0.697 400
 0.388 0.055 0.836 0.518 0.585 0.842 0.793 0.177 0.926 0.964 0.450 0.022 410
 0.085 0.311 0.102 0.816 0.973 0.494 0.208 0.803 0.948 0.462 0.242 0.287 420
 0.546 0.698 0.269 0.339 0.607 0.594 0.102 0.266 0.677 0.668 0.913 0.462 430
 0.562 0.207 0.188 0.264 0.895 0.991 0.893 0.442 0.615 0.709 0.722 0.950 440
 0.208 0.696 0.304 0.557 0.605 0.617 0.256 0.984 0.595 0.715 0.936 0.178 450
 0.141 0.153 0.654 0.544 0.376 0.363 0.793 0.492 0.812 0.447 0.376 0.231 460
 0.644 0.263 0.785 0.341 0.982 0.829999999                              470
----                                                                    480
UNIFORM-GAUSS TEST  211                                                 490
-0.283 0.916 0.776 0.690 0.910 0.506 0.816 0.348 0.659-0.301 0.630-0.397 500
 0.515 0.227 1.253 0.421 0.499 0.288 1.189-0.264 0.099 0.051 0.411 0.951 510
 0.273 0.154 0.861 0.937 0.446 0.702 1.451 0.035 0.515 0.770 0.559 1.053 520
-1.157 0.902 0.533 1.270 0.761 1.110 1.190 0.433 0.573 0.374 1.317 1.255 530
 0.547 1.145 0.667-0.077 0.422-0.159-0.037 0.088 0.406 0.849 0.898 0.372 540
-0.324 0.025 0.632 0.365 0.375 0.694-0.206 0.126-0.381 1.149 0.983 1.184 550
-0.011 0.653 0.266 1.035 0.536 0.936 1.177 1.644 0.782 0.198 0.222 0.445 560
 0.714 0.607 0.374 0.341 0.790 0.302 1.075 0.204 0.436 0.887 0.234 0.874 570
 0.048 0.938 0.733-0.340-0.012 0.497 0.418 0.840 0.091 0.578 0.606-0.340 580
 0.656 0.660 0.584 0.837 0.454 0.695 0.606 0.053-0.276 1.600 1.394-0.038 590
 1.173 0.762 0.642 0.185-0.023 0.037 0.508 0.313-0.718-0.249 0.124-0.096 600
-0.299 0.196 1.086 0.487 0.317 0.635 0.462 0.559 0.181 1.799 0.287 0.583 610
 0.313 0.387 1.067 0.996 0.702 0.068-0.227 0.198 0.305-0.021 0.849 1.063 620
 1.049 0.226-0.297 0.930 0.828 0.884 1.217-0.391 0.007-0.004 1.238 0.376 630
 0.418 1.075 0.083-0.020 0.362 0.601 0.037 0.634 0.109 0.524 1.356 1.024 640
-0.602 0.763 1.261 0.302-0.063 0.704 0.446-0.410 0.401 0.704 1.070-0.023 650
-0.064 1.087-0.737-0.476 1.156 0.648 0.624 0.257 0.643 0.147 0.719 0.174 660
-0.552-0.183 0.583 0.740 0.592-0.144 0.222 0.388 0.563 0.933 1.108 1.022 670
 0.818 0.686 0.683 0.514 0.284-0.280 0.358 1.203-0.643 0.110 0.012 0.399 680
 0.899 0.400 0.994 0.880 0.743 0.102 1.120 0.391 0.191 0.196 1.176 0.149 690
 0.512 1.132 0.916 0.838 0.445 1.330 0.563 0.610 0.699 0.675 0.310 0.586 700
 0.448-0.475 0.317 0.858 0.839-0.297 0.214 0.985 0.484 1.004 1.598 0.494 710
 0.467 1.188 0.536 0.381 1.339-0.011 0.064 0.113 0.619 0.604 0.687 0.622 720
 0.297 0.203 0.378 1.313 0.829 0.422 0.078 0.057 0.143 0.868-0.302 0.693 730
 0.633 1.116 0.118-0.469 0.663 0.708 0.685 0.850 0.566 0.657 1.217 0.394 740
 0.643-0.055-0.000 0.881 1.163 0.520 0.787 1.493 1.366 0.801 0.301 1.384 750
-0.541 1.176 0.236 0.675 1.119 1.000 0.250 0.457-0.010 0.098 0.975 0.288 760
 0.686 0.764 0.007 0.697 0.789 0.259 0.414 0.680 0.852 0.315 0.231 0.203 770
 1.394 0.131 0.963 0.699 0.404-0.124 0.583 0.071 1.838-0.313-0.467 0.191 780
 0.125 1.670 0.224 0.400 0.658 0.900 1.034 0.005 0.801 0.920-0.188 0.786 790
 0.387 1.243 0.875 0.989 0.718-0.152 0.009 1.329 0.562 0.687 0.968 0.490 800
 0.160 0.177-0.025 1.125 0.217 1.206 1.221-0.145-0.088 0.629-0.131 0.272 810
 0.388 0.772 1.046-0.067 0.760 0.428 0.852 0.867 0.610 0.359 1.352 0.571 820
-0.011-0.205 1.084 0.009 0.917 0.438 1.050 0.616 0.884 0.954-0.275 1.295 830
 0.346-1.018 1.049 0.417 1.230 1.127 1.435 0.823-0.069 0.464 1.261-0.179 840
 0.812 0.537 0.875 0.190 0.707 0.857 0.694 0.176 0.916 0.406 0.204 0.494 850
 1.709 0.133 0.460 0.828-0.174 0.457 0.584 0.832-0.026 1.265-0.232-0.373 860
-0.172 0.913 0.673 0.303 0.035 1.226-0.072 0.920 1.207-0.003 0.632 0.004 870
 0.951 0.307 0.798 1.479 0.196 1.058 0.873 0.060 0.524 0.501 0.373 0.954 880
-0.072 0.988 0.351 0.053 0.248 0.430-0.379 0.533 0.688 0.524 0.447 1.376 890
-0.263-0.064 0.301 1.486 0.351 0.806-0.374 0.735 0.958-0.262 0.332 0.963 900
999999                                                                  910
```

Figure 60.

Output

Description

The output from the program KOLM gives the statistics and probability statements described below, and in addition identifies the distribution functions being considered. Sorted samples are printed on option.

The following items are produced as output:

1. Z score, where $z = \sqrt{n}\, D_n$ for the one-sample test; n is the sample size, and D_n is the maximum difference between the empirical distribution function and the theoretical distribution function.

   $z = \sqrt{\frac{mn}{m+n}}\; D_{m,n}$ for the two-sample test, m is the size of the second sample; n is the size of the first sample; $D_{m,n}$ is the maximum difference between the two empirical distribution functions.

2. The probability of incorrectly rejecting the hypothesis of equality of distribution functions.

Sample

See Figure 61.

Program Modification

1. Program capacity may be increased or decreased by making changes in the dimension statements. If this is done, the statements DO 111 I = 1, 50 and DO 117 I = 1, 50 may require modification (under READ SAMPLE comments card in listing of program). This will also be true if data formats require changing. The "read sample" sections of the program use FORMAT statement 3, and the user may wish to change these sections.

2. Any modifications to the subroutine KOLMO in terms of added continuous pdf's should be reflected in the program KOLM. It may be necessary to:
   a. Modify the dimensions of DIST (5, 3), which contains the switches calling on pdf's and also contains the parameters u and s used by KOLMO
   b. Modify the pdf titling card, statement 6 (FORMAT), the READ (5, 6)TIT1 card which reads titles, and the dimension of TIT1 (now 20, to hold five 16-digit titles)
   c. Modify the section of the program from statement 121 through 132 to reflect changes a and b above. These statements call KOLMO to perform tests and output results.

3. List of names in KOLM, and their usage:

D -- Temporary vector used in reading samples
DAS2 -- Initializing card name (----)
DASH -- Job control card name (----)
DIST -- 5 by 3 matrix. The five elements in column 1 are switches that allow the (5) pdf's to be used in one-sample tests. Columns 2 and 3 contain the parameters u and s for the associated test.
IFL -- Error indicator (job deck error)
IER -- Error (in KOLMO, used for skipping the test concerned)
IES -- Error (in KOLMO , used for error message)

```
UNIFORM TEST

A 1 SAMPLE TEST WAS REQUESTED

THE SIZE OF SAMPLE  1 IS 498.

THE HYPOTHESIS THAT THE SAMPLE IS FROM A(N)       NORMAL       DISTRIBUTION
 WITH MEAN        0.5000 AND VARIANCE        0.2500
 CAN BE REJECTED WITH PROBABILITY 0.0   OF BEING INCORRECT.  THE STATISTIC Z
 IS  0.3584E 01 FOR THIS SAMPLE.

THE HYPOTHESIS THAT THE SAMPLE IS FROM A(N)   EXPONENTIAL    DISTRIBUTION
 WITH MEAN        0.5000 AND VARIANCE        1.0000
 CAN BE REJECTED WITH PROBABILITY 0.0   OF BEING INCORRECT.  THE STATISTIC Z
 IS  0.8803E 01 FOR THIS SAMPLE.

THE HYPOTHESIS THAT THE SAMPLE IS FROM A(N)       CAUCHY       DISTRIBUTION
 WITH MEDIAN      0.5000 AND FIRST QUARTILE      -0.5000
 CAN BE REJECTED WITH PROBABILITY 0.0   OF BEING INCORRECT.  THE STATISTIC Z
 IS  0.7887E 01 FOR THIS SAMPLE.

THE HYPOTHESIS THAT THE SAMPLE IS FROM A(N)      UNIFORM       DISTRIBUTION
 IN THE INTERVAL        0.0    TO       1.0000 INCLUSIVE
 CAN BE REJECTED WITH PROBABILITY 0.989 OF BEING INCORRECT.  THE STATISTIC Z
 IS  0.4444E 00 FOR THIS SAMPLE.

 THE JOB WITH TITLE UNIFORM TEST          WAS COMPLETED.


UNIFORM-GAUSS TEST

A 2 SAMPLE TEST WAS REQUESTED

THE SIZE OF SAMPLE  2 IS 492.

SORTED SAMPLE ONE FOLLOWS

   0.0     0.005   0.005   0.006   0.006   0.006   0.008   0.009   0.010   0.012
   0.012   0.018   0.019   0.019   0.021   0.022   0.022   0.025   0.029   0.033
   0.033   0.033   0.037   0.041   0.041   0.042   0.042   0.042   0.043   0.047
   0.053   0.054   0.055   0.058   0.062   0.065   0.068   0.073   0.073   0.077
   0.079   0.079   0.080   0.083   0.083   0.085   0.088   0.092   0.093   0.093
   0.097   0.098   0.102   0.102   0.103   0.111   0.112   0.113   0.113   0.113
   0.117   0.117   0.119   0.121   0.123   0.136   0.139   0.139   0.141   0.143
   0.148   0.151   0.153   0.154   0.156   0.157   0.158   0.161   0.163   0.164
   0.164   0.167   0.168   0.170   0.172   0.177   0.178   0.178   0.178   0.178
   0.181   0.181   0.183   0.188   0.190   0.190   0.193   0.195   0.195   0.196
   0.197   0.199   0.200   0.201   0.202   0.203   0.204   0.206   0.207   0.208
   0.208   0.209   0.215   0.220   0.224   0.225   0.228   0.231   0.231   0.232
   0.236   0.242   0.242   0.243   0.244   0.253   0.256   0.256   0.260   0.263
   0.263   0.264   0.264   0.266   0.268   0.269   0.272   0.275   0.277   0.282
   0.283   0.284   0.285   0.287   0.288   0.288   0.290   0.292   0.294   0.298
   0.299   0.301   0.302   0.302   0.302   0.303   0.304   0.311   0.311   0.312
   0.326   0.328   0.331   0.336   0.337   0.339   0.340   0.341   0.343   0.344
   0.345   0.348   0.348   0.349   0.349   0.353   0.360   0.363   0.364   0.365
   0.365   0.365   0.365   0.367   0.367   0.371   0.371   0.376   0.376   0.376
   0.377   0.377   0.381   0.381   0.382   0.383   0.388   0.390   0.397   0.400
   0.409   0.411   0.416   0.416   0.422   0.422   0.425   0.427   0.430   0.431
   0.434   0.434   0.441   0.442   0.444   0.447   0.450   0.450   0.450   0.453
   0.454   0.455   0.455   0.457   0.458   0.458   0.459   0.462   0.462   0.462
   0.463   0.463   0.467   0.469   0.470   0.471   0.472   0.476   0.477   0.478
   0.482   0.482   0.492   0.494   0.500   0.501   0.501   0.502   0.503   0.504
   0.508   0.512   0.514   0.514   0.515   0.516   0.516   0.518   0.518   0.520
   0.520   0.524   0.531   0.536   0.539   0.540   0.544   0.545   0.546   0.546
   0.554   0.555   0.556   0.556   0.557   0.560   0.561   0.562   0.565   0.570
   0.570   0.572   0.574   0.577   0.580   0.581   0.581   0.585   0.588   0.589
   0.591   0.592   0.594   0.594   0.595   0.595   0.595   0.596   0.597   0.603
   0.604   0.605   0.607   0.607   0.607   0.607   0.608   0.610   0.611   0.615
   0.617   0.622   0.622   0.624   0.629   0.630   0.632   0.634   0.634   0.636
   0.637   0.638   0.639   0.640   0.644   0.646   0.652   0.652   0.654   0.657
   0.659   0.662   0.662   0.664   0.664   0.665   0.666   0.666   0.667   0.668
   0.668   0.670   0.671   0.672   0.677   0.679   0.683   0.686   0.688   0.692
   0.694   0.694   0.696   0.697   0.698   0.699   0.699   0.701   0.702   0.707
   0.708   0.708   0.709   0.715   0.722   0.724   0.725   0.726   0.728   0.733
   0.734   0.737   0.741   0.742   0.743   0.745   0.745   0.746   0.750   0.751
   0.752   0.753   0.754   0.759   0.761   0.761   0.766   0.767   0.783   0.785
   0.785   0.786   0.787   0.790   0.793   0.793   0.795   0.801   0.801   0.803
   0.804   0.805   0.806   0.810   0.810   0.810   0.812   0.814   0.816   0.816
   0.816   0.817   0.827   0.829   0.833   0.835   0.836   0.837   0.840   0.842
   0.842   0.843   0.843   0.844   0.845   0.846   0.851   0.859   0.860   0.860
   0.864   0.867   0.868   0.870   0.870   0.876   0.880   0.882   0.882   0.888
   0.889   0.890   0.892   0.893   0.895   0.904   0.907   0.909   0.909   0.913
   0.913   0.918   0.919   0.923   0.923   0.924   0.926   0.928   0.928   0.931
   0.931   0.933   0.936   0.939   0.940   0.945   0.948   0.950   0.951   0.953
   0.953   0.956   0.956   0.962   0.963   0.963   0.964   0.965   0.967   0.969
   0.969   0.972   0.973   0.973   0.976   0.978   0.982   0.982   0.984   0.984
   0.985   0.988   0.990   0.991   0.994   0.995   0.996   0.996
```

Figure 61.

```
SORTED SAMPLE TWO FOLLOWS

 -1.157   -1.018   -0.737   -0.718   -0.643   -0.602   -0.552   -0.541   -0.476   -0.475
 -0.469   -0.467   -0.410   -0.397   -0.391   -0.381   -0.379   -0.374   -0.373   -0.340
 -0.340   -0.324   -0.313   -0.302   -0.301   -0.299   -0.297   -0.297   -0.283   -0.280
 -0.276   -0.275   -0.264   -0.263   -0.262   -0.249   -0.232   -0.227   -0.206   -0.205
 -0.188   -0.183   -0.179   -0.174   -0.172   -0.159   -0.152   -0.145   -0.144   -0.131
 -0.124   -0.096   -0.088   -0.077   -0.072   -0.072   -0.069   -0.067   -0.064   -0.064
 -0.063   -0.055   -0.038   -0.037   -0.026   -0.025   -0.023   -0.023   -0.021   -0.020
 -0.012   -0.011   -0.011   -0.011   -0.010   -0.004   -0.003    0.0      0.004    0.005
  0.007    0.007    0.009    0.009    0.012    0.025    0.035    0.035    0.037    0.037
  0.048    0.051    0.053    0.053    0.057    0.060    0.064    0.068    0.071    0.078
  0.083    0.088    0.091    0.094    0.098    0.099    0.102    0.109    0.110    0.113
  0.118    0.124    0.125    0.126    0.131    0.133    0.143    0.147    0.149    0.154
  0.160    0.174    0.176    0.177    0.181    0.185    0.190    0.191    0.191    0.196
  0.196    0.196    0.198    0.198    0.203    0.203    0.204    0.204    0.214    0.217
  0.222    0.222    0.224    0.226    0.227    0.231    0.234    0.236    0.248    0.250
  0.257    0.259    0.266    0.272    0.273    0.284    0.287    0.288    0.288    0.297
  0.301    0.301    0.302    0.302    0.303    0.305    0.307    0.310    0.313    0.313
  0.315    0.317    0.317    0.332    0.341    0.346    0.348    0.351    0.351    0.358
  0.359    0.362    0.365    0.372    0.373    0.374    0.374    0.375    0.376    0.378
  0.381    0.387    0.387    0.388    0.388    0.391    0.394    0.399    0.400    0.400
  0.401    0.404    0.406    0.406    0.411    0.414    0.417    0.418    0.418    0.421
  0.422    0.422    0.428    0.430    0.433    0.436    0.438    0.445    0.445    0.446
  0.446    0.447    0.448    0.454    0.457    0.457    0.460    0.462    0.464    0.467
  0.484    0.487    0.490    0.494    0.494    0.497    0.499    0.501    0.506    0.508
  0.512    0.514    0.515    0.515    0.520    0.524    0.524    0.524    0.533    0.533
  0.536    0.536    0.537    0.547    0.559    0.559    0.562    0.563    0.563    0.566
  0.571    0.573    0.578    0.583    0.583    0.583    0.584    0.584    0.586    0.592
  0.601    0.604    0.606    0.606    0.607    0.610    0.610    0.616    0.619    0.622
  0.624    0.629    0.630    0.632    0.632    0.633    0.634    0.635    0.642    0.643
  0.643    0.648    0.653    0.656    0.657    0.658    0.659    0.660    0.663    0.667
  0.673    0.675    0.675    0.680    0.683    0.685    0.686    0.686    0.687    0.687
  0.688    0.690    0.693    0.694    0.695    0.697    0.699    0.699    0.702    0.702
  0.704    0.704    0.707    0.708    0.714    0.718    0.719    0.733    0.735    0.740
  0.743    0.760    0.761    0.762    0.763    0.764    0.770    0.772    0.776    0.782
  0.786    0.787    0.789    0.790    0.798    0.801    0.801    0.806    0.812    0.816
  0.818    0.823    0.828    0.828    0.829    0.832    0.837    0.838    0.839    0.840
  0.849    0.849    0.850    0.852    0.852    0.857    0.858    0.861    0.867    0.868
  0.873    0.874    0.875    0.875    0.880    0.881    0.884    0.884    0.887    0.898
  0.899    0.900    0.902    0.910    0.913    0.916    0.916    0.916    0.917    0.920
  0.920    0.930    0.933    0.936    0.937    0.938    0.951    0.951    0.954    0.954
  0.958    0.963    0.963    0.968    0.975    0.983    0.985    0.988    0.989    0.994
  0.996    1.000    1.004    1.022    1.024    1.034    1.035    1.046    1.049    1.049
  1.050    1.053    1.058    1.063    1.067    1.070    1.075    1.075    1.084    1.086
  1.087    1.108    1.110    1.116    1.119    1.120    1.125    1.127    1.132    1.145
  1.149    1.156    1.163    1.173    1.176    1.176    1.177    1.184    1.188    1.189
  1.190    1.203    1.206    1.207    1.217    1.217    1.221    1.226    1.230    1.238
  1.243    1.253    1.255    1.261    1.261    1.265    1.270    1.295    1.313    1.317
  1.329    1.330    1.339    1.352    1.356    1.366    1.376    1.384    1.394    1.394
  1.435    1.451    1.479    1.486    1.493    1.598    1.600    1.644    1.670    1.709
  1.799    1.838

THE HYPOTHESIS THAT THE TWO SAMPLES ARE FROM THE SAME POPULATION CAN BE REJECTED WITH (ASYMPTOTIC)
PROBABILITY OF BEING INCORRECT OF  0.000.  THE STATISTIC Z IS   0.2590E 01 FOR THESE SAMPLES.

 THE JOB WITH TITLE UNIFORM-GAUSS TEST   WAS COMPLETED.
```

Figure 61. (Continued)

IO -- Switch (used for printing samples)
IR -- Number of samples to be read in current job
IS -- Number of samples to be used in current job (1 or 2)
K -- Counter used to print correct pdf name for pdf used in the test
M -- Size of second sample
N -- Size of first sample
N1 -- 1 or 2, for number of samples tested
P -- Probability of being incorrect if hypothesis is rejected
S2 -- Temporary storage for u and s output
TIT1 -- Current pdf names
TITLE -- Job title
X -- Sample 1
Y -- Sample 2
Z -- Z statistic, from KOLMO or KOLM2

Operating Instructions

This sample program is a standard FORTRAN program and needs no special operating instructions. Data set 5 is used for input, and 6 for output.

Error Messages

The following error conditions will result in messages, followed by the action specified:

1. Neither a one- nor two-sample test is requested, or the size of either sample is larger than 500 -- CC.21, CONTROL CARD, INCORRECT, OR SAMPLE SIZE TOO LARGE. JOB IGNORED. Action: Cards are read until a new job control card is found, or until the hopper is empty.

2. Sample size is less than 100 (not an error) -- NOTE THE REMARKS CONCERNING ASYMPTOTIC RESULTS AND SAMPLE SIZE IN SUBROUTINE SMIRN. Action: None - job continues.

3. The requirement of subroutine KOLMO that certain parameters be nonzero or positive is violated -- AT LEAST ONE (S) ENTRY PARAMETER FOR THE SUBROUTINE KOLMO WAS INCORRECT. THE TEST FOR THE ASSOCIATED CONTINUOUS PDF WAS IGNORED. Action: All tests are made for cases where the parameter s was correct.

4. A case in which an error requires aborting the job, and the succeeding job in the job stack requests a two-sample test where the second sample is to be

compared with a (first) sample which was read on the previous job -- THIS JOB CALLS FOR THE USE OF A PREVIOUSLY READ SAMPLE, AND THE PREVIOUS JOB WAS IGNORED BECAUSE OF ERRORS. JOB IGNORED. Action: Cards are read until a new job control card is found, or until the hopper is empty.

5. The job control card preceding a job is not there or is incorrect -- FIRST CARD IN JOB DECK (JOB CONTROL CARD) IS INCORRECT. Action: Cards are read until a new job control card is found, or until the hopper is empty.

Timing

The execution time of this program on a System/360, Model 40, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 Printer as output, is:

33 seconds for job 1
50 seconds for job 2

```
C                                                                       KOLM  10
C     ..............................................................KOLM  20
C                                                                       KOLM  30
C        SAMPLE MAIN PROGRAM FOR THE KOLMOGOROV-SMIRNOV TEST-KOLM       KOLM  40
C                                                                       KOLM  50
C        PURPOSE                                                        KOLM  60
C           (1) READ THE CONTROL CARD FOR A ONE OR TWO SAMPLE TEST      KOLM  70
C           (2) READ THE SAMPLE DATA AND DETERMINE THE SAMPLE SIZES     KOLM  80
C           (3) PRINT RESULTS                                           KOLM  90
C                                                                       KOLM 100
C        REMARKS                                                        KOLM 110
C           THE USER SHOULD NOTE THE REMARKS GIVEN IN SUBROUTINES       KOLM 120
C           KOLMO, KOLM2, AND SMIRN, AND THE MATHEMATICAL DESCRIPTIONS  KOLM 130
C           FOR THESE SUBROUTINES.                                      KOLM 140
C                                                                       KOLM 150
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  KOLM 160
C           KOLMO                                                       KOLM 170
C           KOLM2                                                       KOLM 180
C           SMIRN                                                       KOLM 190
C           NDTR                                                        KOLM 200
C                                                                       KOLM 210
C        METHOD                                                         KOLM 220
C           REFER TO SUBROUTINES KOLMO, KOLM2, AND SMIRN                KOLM 230
C                                                                       KOLM 240
C     ..............................................................KOLM 250
C                                                                       KOLM 260
C     THE FOLLOWING DIMENSIONS MUST BE GREATER THAN THE NUMBER OF DATA  KOLM 270
C     ELEMENTS IN THE TWO SAMPLES, M AND N                              KOLM 280
C                                                                       KOLM 290
      DIMENSION X(501),Y(501)                                           KOLM 300
C                                                                       KOLM 310
C     ..............................................................KOLM 320
C                                                                       KOLM 330
      DIMENSION TITLE(5),D(12),TIT1(20),DIST(5,3)                       KOLM 340
C                                                                       KOLM 350
C     ..............................................................KOLM 360
C                                                                       KOLM 370
    1 FORMAT(5A4,3I1,5(F1.0,2F5.0))                                     KOLM 380
    2 FORMAT(///'CC.21, CONTROL CARD, INCORRECT, OR SAMPLE SIZE IS TOO LAKOLM 390
     1RGE.  JOB IGNORED.')                                              KOLM 400
    3 FORMAT(12F6.0)                                                    KOLM 410
    4 FORMAT(1H1,5A4)                                                   KOLM 420
    5 FORMAT(//2H A,I2,' SAMPLE TEST WAS REQUESTED')                    KOLM 430
    6 FORMAT(20A4)                                                      KOLM 440
    7 FORMAT(//(10F10.3))                                               KOLM 450
    8 FORMAT(//' SORTED SAMPLE ONE FOLLOWS')                            KOLM 460
    9 FORMAT(//' THE HYPOTHESIS THAT THE SAMPLE IS FROM A(N) ',4A4,  ' DKOLM 470
     1ISTRIBUTION')                                                     KOLM 480
   10 FORMAT(//' SORTED SAMPLE TWO FOLLOWS')                            KOLM 490
   11 FORMAT(//' THE HYPOTHESIS THAT THE TWO SAMPLES ARE FROM THE SAME PKOLM 500
     1OPULATION CAN BE REJECTED WITH (ASYMPTOTIC)',/,' PROBABILITY OF BEKOLM 510
     2ING INCORRECT OF ',F6.3,'.  THE STATISTIC Z IS ',E12.4,' FOR THESEKOLM 520
     3 SAMPLES.')                                                       KOLM 530
   12 FORMAT(//,' THE SIZE OF SAMPLE',I3,' IS',I4,'.')                  KOLM 540
   13 FORMAT(//,' NOTE THE REMARKS CONCERNING ASYMPTOTIC RESULTS AND SAMKOLM 550
     1PLE SIZE IN SUBROUTINE SMIRN')                                    KOLM 560
   14 FORMAT(//,' AT LEAST ONE (S) ENTRY PARAMETER FOR THE SUBROUTINE KDKOLM 570
     1LMO WAS INCORRECT.'/' THE TEST FOR THE ASSOCIATED CONTINUOUS PDF WKOLM 580
     2AS IGNORED.')                                                     KOLM 590
   15 FORMAT(A4)                                                        KOLM 600
   16 FORMAT(//,' THIS JOB CALLS FOR THE USE OF A PREVIOUSLY READ SAMPLEKOLM 610
     1, AND THE PREVIOUS JOB WAS IGNORED BECAUSE OF ERRORS.'/ ' JOB IGNOKOLM 620
     2RED.')                                                            KOLM 630
   17 FORMAT(//,' FIRST CARD IN JOB DECK (JOB CONTROL CARD) IS INCORRECTKOLM 640
     1.')                                                               KOLM 650
   18 FORMAT(1H ,' WITH MEAN',F13.4,' AND VARIANCE',F13.4)              KOLM 660
   19 FORMAT(1H ,' WITH MEDIAN',F13.4,' AND FIRST QUARTILE',F13.4)      KOLM 670
   20 FORMAT(1H ,' IN THE INTERVAL',F13.4,' TO',F13.4,' INCLUSIVE')     KOLM 680
   21 FORMAT(1H ,' CAN BE REJECTED WITH PROBABILITY',F6.3,' OF BEING INCKOLM 690
     1ORRECT.  THE STATISTIC Z',/,'  IS',E12.4,' FOR THIS SAMPLE.')     KOLM 700
   22 FORMAT(//,'  THE JOB WITH TITLE ',5A4,' WAS COMPLETED.')           KOLM 710
C                                                                       KOLM 720
C        READ DISTRIBUTION NAMES AND JOB CONTROL CARD                   KOLM 730
C                                                                       KOLM 740
      IFL=0                                                             KOLM 750
      READ(5,15)DASH                                                    KOLM 760
      READ(5,6)TIT1                                                     KOLM 770
C                                                                       KOLM 780
C        SELECT PROGRAM CONTROLS                                        KOLM 790
C                                                                       KOLM 800
  100 READ(5,15)DAS2                                                    KOLM 810
      IF(DASH-DAS2)101,102,101                                          KOLM 820
  101 WRITE(6,17)                                                       KOLM 830
      GO TO 107                                                         KOLM 840
  102 READ(5,1)TITLE,IS,IR,IO,((DIST(I,J),J=1,3),I=1,5)                 KOLM 850
      IES=0                                                             KOLM 860
      WRITE(6,4)TITLE                                                   KOLM 870
      WRITE(6,5)IS                                                      KOLM 880
C                                                                       KOLM 890
C        NUMBER OF SAMPLES DECISION                                     KOLM 900
C                                                                       KOLM 910
      IF(IR)103,105,103                                                 KOLM 920
  103 IF(IFL)104,115,104                                                KOLM 930
  104 WRITE(6,16)                                                       KOLM 940
      GO TO 107                                                         KOLM 950
  105 IF(IS-1)106,109,109                                               KOLM 960
C                                                                       KOLM 970
C        NOT ONE OR TWO SAMPLES                                         KOLM 980
C                                                                       KOLM 990
  106 WRITE(6,2)                                                        KOLM1000
  107 READ(5,15)DAS2                                                    KOLM1010
      IF(DASH-DAS2)107,108,107                                          KOLM1020
  108 IFL=1                                                             KOLM1030
      GO TO 102                                                         KOLM1040
C                                                                       KOLM1050
C        READ FIRST SAMPLE                                              KOLM1060
C                                                                       KOLM1070
  109 N=0                                                               KOLM1080
      DO 111 I=1,50                                                     KOLM1090
      READ(5,3)D                                                        KOLM1100
      DO 111 J=1,12                                                     KOLM1110
      IF(D(J)-999999.0)110,112,110                                      KOLM1120
  110 N=N+1                                                             KOLM1130
      IF(N-501)111,106,106                                              KOLM1140
  111 X(N)=D(J)                                                         KOLM1150
  112 N1=1                                                              KOLM1160
      WRITE(6,12)N1,N                                                   KOLM1170
C                                                                       KOLM1180
C        CHECK THE SIZE OF N                                            KOLM1190
C                                                                       KOLM1200
      IF(N-100)113,113,114                                              KOLM1210
  113 WRITE(6,13)                                                       KOLM1220
  114 IF(IS-2)121,115,106                                               KOLM1230
C                                                                       KOLM1240
C        READ SECOND SAMPLE                                             KOLM1250
C                                                                       KOLM1260
  115 M=0                                                               KOLM1270
      DO 117 I=1,50                                                     KOLM1280
      READ(5,3)D                                                        KOLM1290
      DO 117 J=1,12                                                     KOLM1300
      IF(D(J)-999999.0)116,118,116                                      KOLM1310
  116 M=M+1                                                             KOLM1320
      IF(M-501)117,106,106                                              KOLM1330
  117 Y(M)=D(J)                                                         KOLM1340
  118 N1=2                                                              KOLM1350
      WRITE(6,12)N1,M                                                   KOLM1360
C                                                                       KOLM1370
C        CHECK THE SIZE OF M                                            KOLM1380
C                                                                       KOLM1390
      IF(M-100)119,119,120                                              KOLM1400
  119 WRITE(6,13)                                                       KOLM1410
  120 IF(IS-1)121,121,133                                               KOLM1420
C                                                                       KOLM1430
C        ONE SAMPLE TEST USING ALL DISTRIBUTIONS REQUESTED              KOLM1440
C                                                                       KOLM1450
  121 DO 130 I=1,5                                                      KOLM1460
      IF(DIST(I,1))130,130,122                                          KOLM1470
  122 CALL KOLMO(X,N,Z,P,I,DIST(I,2),DIST(I,3),IER)                     KOLM1480
      IES=IER+IES                                                       KOLM1490
      IF(IER)130,124,130                                                KOLM1500
  123 WRITE(6,14)                                                       KOLM1510
      GO TO 136                                                         KOLM1520
C                                                                       KOLM1530
C        OUTPUT RESULTS                                                 KOLM1540
C                                                                       KOLM1550
  124 K=4*I-3                                                           KOLM1560
      WRITE(6,9)TIT1(K),TIT1(K+1),TIT1(K+2),TIT1(K+3)                   KOLM1570
      IF(I-3)125,126,127                                                KOLM1580
  125 S2=DIST(I,3)**2                                                   KOLM1590
      WRITE(6,18)DIST(I,2),S2                                           KOLM1600
      GO TO 129                                                         KOLM1610
  126 S2=DIST(I,2)-DIST(I,3)                                            KOLM1620
      WRITE(6,19)DIST(I,2),S2                                           KOLM1630
      GO TO 129                                                         KOLM1640
  127 IF(I-4)128,128,130                                                KOLM1650
  128 WRITE(6,20)DIST(I,2),DIST(I,3)                                    KOLM1660
  129 WRITE(6,21)P,Z                                                    KOLM1670
  130 CONTINUE                                                          KOLM1680
C                                                                       KOLM1690
C        OUTPUT SAMPLE ONE DECISION                                     KOLM1700
C                                                                       KOLM1710
      IF(IO)131,132,131                                                 KOLM1720
  131 WRITE(6,8)                                                        KOLM1730
      WRITE(6,7)(X(J),J=1,N)                                            KOLM1740
  132 IF(IES)123,136,123                                                KOLM1750
C                                                                       KOLM1760
C        TWO SAMPLE TEST                                                KOLM1770
C                                                                       KOLM1780
  133 CALL KOLM2(X,Y,N,M,Z,P)                                           KOLM1790
C                                                                       KOLM1800
C        OUTPUT SAMPLES DECISION                                        KOLM1810
C                                                                       KOLM1820
      IF(IO)134,135,134                                                 KOLM1830
  134 WRITE(6,8)                                                        KOLM1840
      WRITE(6,7)(X(J),J=1,N)                                            KOLM1850
      WRITE(6,10)                                                       KOLM1860
      WRITE(6,7)(Y(J),J=1,M)                                            KOLM1870
  135 WRITE(6,11)P,Z                                                    KOLM1880
  136 IFL=0                                                             KOLM1890
      WRITE(6,22)TITLE                                                  KOLM1900
      GO TO 100                                                         KOLM1910
      END                                                               KOLM1920
```

TRIPLE EXPONENTIAL SMOOTHING

Problem Description

Given a time series X, a smoothing constant, and three coefficients of the prediction equation, this sample program finds the triple exponentially smoothed series S of the time series X.

Program

Description

The sample program for triple exponential smoothing consists of the main program named EXPON, and one subroutine, EXSMO, from the Scientific Subroutine Package.

Capacity

The capacity of the sample program and the format required for data input have been set up as follows:

1. Up to 1000 data points in a given time series
2. (12F6.0) format for input data cards

Therefore, if a problem satisfies the above conditions, the sample program need not be modified. However, if there are more than 1000 data points, the dimension statement in the sample main program must be modified to handle this particular problem. Similarly, if input data cards are prepared using a different format, the input format in the sample main program must be modified. The general rules for program modification are described later.

Input

Control Card

One control card is required for each problem and is read by the main program, EXPON. This card is prepared as follows:

Columns	Contents	For Sample Problem
1- 6	Problem number (may be alphameric)	SAMPLE
7-10	Number of data points in a given time series	0038
11-15	Smoothing constant, α $(0.0 < \alpha < 1.0)$	0.1
16-25	First coefficient (A) of the prediction equation	0.0
26-35	Second coefficient (B) of the prediction equation	0.0
36-45	Third coefficient (C) of the prediction equation	0.0

Smoothing constant and three coefficients must be keypunched with decimal points.

Leading zeros are not required to be keypunched.

Data Card

Time series data are keypunched using the format (12F6.0). This format assumes that each data point is keypunched in a six-column field and twelve fields per card.

Deck Setup

The deck setup is shown in Figure 62.

Sample

The listing of input cards for the sample problem is presented in Figure 63.

Output

Description

The output of the sample program for triple exponential smoothing includes:

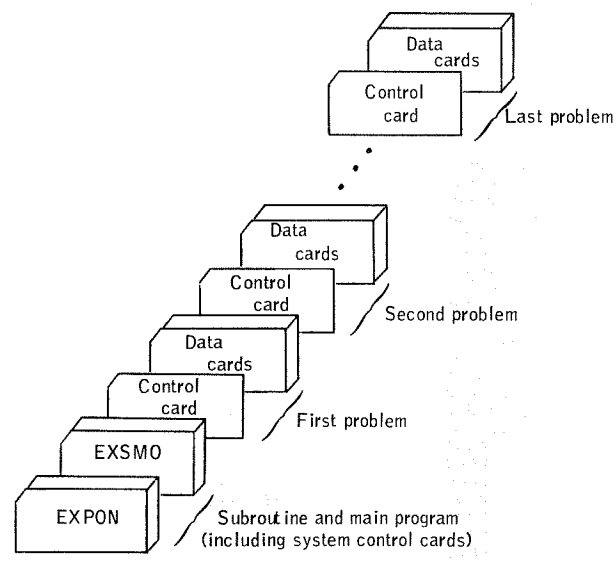


Figure 62. Deck setup (triple exponential smoothing)

```
/DATA                                                                   10
SAMPLE  38  0.1       0.0       0.0       0.0                           20
   430   426   422   419   414   413   412   409   411   417   422   430  30
   438   441   447   455   461   453   448   449   454   463   470   472  40
   476   481   483   487   491   492   485   486   482   479   479   476  50
   472   470                                                             60
```

Figure 63. Input listing (triple exponential smoothing)

1. Original and updated coefficients
2. Time series as input and triple exponentially smoothed time series

Sample

The output listing for the sample problem is shown in Figure 64.

Program Modification

Program capacity can be increased or decreased by making changes in the dimension statement. Input data in a different format can also be handled by providing a specific format statement. In order to familiarize the user with the program modification, the following general rules are supplied in terms of the sample problem.

```
TRIPLE EXPONENTIAL SMOOTHING.....SAMPLE

NUMBER OF DATA POINTS     38
SMOOTHING CONSTANT     0.100

COEFFICIENTS          A              B              C

ORIGINAL            0.0            0.0            0.0

UPDATED         484.80176        1.71309        0.04166

                                SMOOTHED DATA
      INPUT DATA                (FORECAST)
       430.00000                 430.00000
       426.00000                 426.00000
       422.00000                 422.00000
       419.00000                 418.00000
       414.00000                 414.29980
       413.00000                 410.23950
       412.00000                 407.08960
       409.00000                 404.66797
       411.00000                 402.22363
       417.00000                 401.25049
       422.00000                 402.64575
       430.00000                 405.61621
       438.00000                 410.71338
       441.00000                 417.46948
       447.00000                 423.99829
       455.00000                 431.18286
       461.00000                 439.43359
       453.00000                 447.87866
       448.00000                 452.21558
       449.00000                 454.10522
       454.00000                 455.80713
       463.00000                 458.54614
       470.00000                 463.30518
       472.00000                 469.06445
       476.00000                 474.09521
       481.00000                 479.11035
       483.00000                 484.38623
       487.00000                 488.94629
       491.00000                 493.50854
       492.00000                 498.05444
       485.00000                 501.66992
       486.00000                 502.12549
       482.00000                 502.44434
       479.00000                 501.16724
       479.00000                 498.92749
       476.00000                 496.84155
       472.00000                 494.00806
       470.00000                 490.30420
```

Figure 64. Output listing (triple exponential smoothing)

1. Changes in the dimension statement of the main program, EXPON:

The dimension of arrays X and S must be greater than or equal to the number of data points in time series, NX. Since there are 38 data points in the sample problem, the value of NX is 38.

2. Changes in the input format statement of the main program, EXPON:

Only the format statement for input data may be changed. Since sample data are three-digit numbers, rather than using six-column fields as in the sample problem, each data point may be keypunched in a three-column field and 24 fields per card. If so, the format is changed to (24F3.0).

Operating Instructions

The sample program for triple exponential smoothing is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is twelve seconds.

```
C                                                                       EXPO  10
C     ..................................................................EXPO  20
C                                                                       EXPO  30
C        SAMPLE MAIN PROGRAM FOR TRIPLE EXPONENTIAL SMOOTHING - EXPON   EXPO  40
C                                                                       EXPO  50
C        PURPOSE                                                        EXPO  60
C           (1) READ THE PROBLEM PARAMETER CARD AND A TIME SERIES,      EXPO  70
C           (2) CALL THE SUBROUTINE EXSMO TO SMOOTH THE TIME SERIES,    EXPO  80
C           AND (3) PRINT THE RESULT.                                   EXPO  90
C                                                                       EXPO 100
C        REMARKS                                                        EXPO 110
C           A SMOOTHING CONSTANT SPECIFIED IN THE PROBLEM PARAMETER     EXPO 120
C           CARD MUST BE GREATER THAN ZERO BUT LESS THAN ONE IN ORDER   EXPO 130
C           TO OBTAIN REASONABLE RESULTS.                               EXPO 140
C                                                                       EXPO 150
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  EXPO 160
C           EXSMO                                                       EXPO 170
C                                                                       EXPO 180
C        METHOD                                                         EXPO 190
C           REFER TO R. G. BROWN, 'SMOOTHING, FORECASTING AND PREDICTIONEXPO 200
C           OF DISCRETE TIME SERIES', PRENTICE-HALL, N.J., 1963,        EXPO 210
C           PP. 140 TO 144.                                             EXPO 220
C                                                                       EXPO 230
C     ..................................................................EXPO 240
C                                                                       EXPO 250
C     THE FOLLOWING DIMENSION MUST BE GREATER THAN OR EQUAL TO THE      EXPO 260
C     NUMBER OF DATA POINTS IN A GIVEN TIME SERIES..                    EXPO 270
C                                                                       EXPO 280
      DIMENSION X(1000),S(1000)                                         EXPO 290
C                                                                       EXPO 300
C     ..................................................................EXPO 310
C                                                                       EXPO 320
    1 FORMAT(A4,A2,I4,F5.0,3F10.0)                                      EXPO 330
    2 FORMAT(12F6.0)                                                    EXPO 340
    3 FORMAT(34H1TRIPLE EXPONENTIAL SMOOTHING.....,A4,A2//22H NUMBER OF EXPO 350
     1DATA POINTS,I6/19H SMOOTHING CONSTANT,F9.3/)                      EXPO 360
    4 FORMAT(13HOCOEFFICIENTS,9X,1HA,14X,1HB,14X,1HC)                   EXPO 370
    5 FORMAT(9HOORIGINAL,F19.5,2F15.5)                                  EXPO 380
    6 FORMAT(8HOUPDATED,F20.5,2F15.5/)                                  EXPO 390
    7 FORMAT(1H0,27X,13HSMOOTHED DATA/7X,10HINPUT DATA,12X,10H(FORECAST)EXPO 400
     1)                                                                 EXPO 410
    8 FORMAT(F17.5,8X,F15.5)                                            EXPO 420
C                                                                       EXPO 430
C     ..................................................................EXPO 440
C                                                                       EXPO 450
C     READ PROBLEM PARAMETER CARD                                       EXPO 460
C                                                                       EXPO 470
  100 READ (5,1) PR,PR1,NX,AL,A,B,C                                     EXPO 480
C        PR......PROBLEM NUMBER (MAY BE ALPHAMERIC)                     EXPO 490
C        PR1.....PROBLEM NUMBER (CONTINUED)                             EXPO 500
C        NX......NUMBER OF DATA POINTS IN TIME SERIES                   EXPO 510
C        AL......SMOOTHING CONSTANT                                     EXPO 520
C        A,B,C...COEFFICIENTS OF THE PREDICTION EQUATION                EXPO 530
C                                                                       EXPO 540
      WRITE (6,3) PR,PR1,NX,AL                                          EXPO 550
C                                                                       EXPO 560
C     PRINT ORIGINAL COEFFICIENTS                                       EXPO 570
C                                                                       EXPO 580
      WRITE (6,4)                                                       EXPO 590
      WRITE (6,5) A,B,C                                                 EXPO 600
```

```
C                                                                       EXPO 610
C     READ TIME SERIES DATA                                             EXPO 620
C                                                                       EXPO 630
      READ (5,2) (X(I),I=1,NX)                                          EXPO 640
C                                                                       EXPO 650
      CALL EXSMO (X,NX,AL,A,B,C,S)                                      EXPO 660
C                                                                       EXPO 670
C     PRINT UPDATED COEFFICIENTS                                        EXPO 680
C                                                                       EXPO 690
      WRITE (6,6) A,B,C                                                 EXPO 700
C                                                                       EXPO 710
C     PRINT INPUT AND SMOOTHED DATA                                     EXPO 720
C                                                                       EXPO 730
      WRITE (6,7)                                                       EXPO 740
      DO 200 I=1,NX                                                     EXPO 750
  200 WRITE (6,8) X(I),S(I)                                             EXPO 760
      GO TO 100                                                         EXPO 770
      END                                                               EXPO 780
```

MATRIX ADDITION

Problem Description

An input matrix is added to another input matrix to form a resultant matrix. Each set of input matrices and the corresponding output matrix is printed. This procedure is repeated until all sets of input matrices have been processed.

Program

Description

The matrix addition program consists of a main routine, ADSAM, and four subroutines, MATIN, MADD, MXOUT, and LOC. MADD and LOC are from the Scientific Subroutine Package. MATIN and MXOUT are sample subroutines for matrix input and output (see program listings).

Capacity

Matrix size has arbitrarily been set at 1000 data elements.

Input

Control Cards

Each input matrix must be preceded by a parameter card with the following format:

Columns	Contents	For Sample Problem
1-2	Blank	
3-6	Up to four-digit identification code	0001
7-10	Number of rows in matrix	0008
11-14	Number of columns in matrix	0011
15-16	Storage mode of matrix 0 for general matrix 1 for symmetric matrix 2 for diagonal matrix	0

Each input matrix must be followed by a card with a 9-punch in column 1.

A blank card after the last pair of input matrices terminates the run.

Data Cards

Data cards are assumed to have seven fields of ten columns each. The decimal point may appear anywhere in a field. If no decimal point is included, it is assumed that the decimal point is to the right of the last digit. The number in each field may be preceded by blanks. Data elements must be punched by row. A row may continue from card to card. However, each new row must start in the first field of the next card. Only the upper triangular portion of a symmetric matrix or the diagonal elements of a diagonal matrix are contained on data cards. The first element of each new row will be the diagonal element for a matrix with symmetric or diagonal storage mode. Columns 71-80 of data cards may be used for identification, sequence numbering, and so on.

Deck Setup

The deck setup is shown in Figure 65.

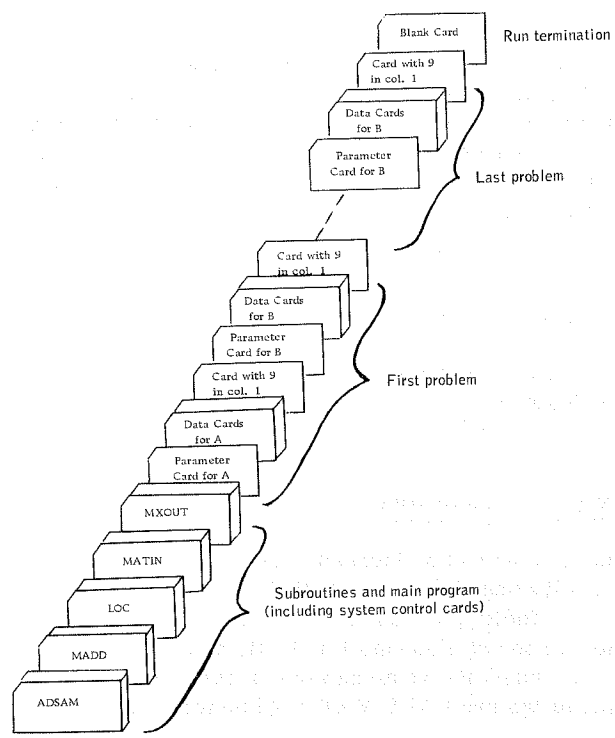


Figure 65. Deck setup (matrix addition)

Sample

A listing of input cards for the sample problem is presented in Figure 66.

```
/DATA
  0001000800110 0
0.7601008 0.6271802 1.0000000 0.7086843 0.4058519 0.0031426 0.6876602
0.6751766 0.8635910 0.7446845 0.6963269
0.6644085 1.0000000 0.6271802 0.6194650 0.3547574 0.0027470 0.6010878
0.5571068 0.7125728 0.6144597 0.5745585
1.0000000 0.6644085 0.7601008 0.7507505 0.4299425 0.0033291 0.7284786
0.6373449 0.8152021 0.7029582 0.6573101
0.6963269 0.5745585 0.6573101 0.6492243 0.3718001 0.0028789 0.6299642
0.6295047 0.8051740 0.6943108 0.6492243
0.7446845 0.6144597 0.7029582 0.6943108 0.3976204 0.0030789 0.6737132
0.3605070 0.4611099 0.3976204 0.3718001
0.6751766 0.5571068 0.6373449 0.6295047 0.3605070 0.0027915 0.6108296
0.0027915 0.0035705 0.0030789 0.0028789
0.0033291 0.0027470 0.0031426 0.0031039 0.0017776 1.0000000 0.0030119
0.6108296 0.7812874 0.6737132 0.6299642
0.4299425 0.3547574 0.4058519 0.4008593 1.0000000 0.0017776 0.3889673
1.0000000 0.7241215 0.6244183 0.5838704
9
  0002000800110 0
0.7507505 0.6194650 0.7086843 1.0000000 0.4008593 0.0031039 0.6792011
0.7241215 1.0000000 0.7986682 0.7468050
0.7446845 0.6144597 0.7029582 0.6943108 0.3976204 0.0030789 0.6737132
0.6244183 0.7986682 1.0000000 0.6439786
0.8635910 0.7125728 0.8152021 0.8051740 0.4611099 0.0035705 0.7812874
0.5838704 0.7468050 0.6439786 1.0000000
0.6963269 0.5745585 0.6573101 0.6492243 0.3718001 0.0028789 0.6299642
1.0000000 0.6644085 0.7601008 0.7507505 0.4299425 0.0033291 0.7284786
0.6751766 0.5571068 0.6373449 0.6295047 0.3605070 0.0027915 0.6108296
0.7284786 0.6010878 0.6876602 0.6792011 0.3889673 0.0030119 1.0000000
0.7446845 0.6144597 0.7029582 0.6943108 0.3976204 0.0030789 0.6737132
0.4299425 0.3547574 0.4058519 0.4008593 1.0000000 0.0017776 0.3889673
0.8635910 0.7125728 0.8152021 0.8051740 0.4611099 0.0035705 0.7812874
0.7601008 0.6271802 1.0000000 0.7086843 0.4058519 0.0031426 0.6876602
0.7601008 0.6271802 1.0000000 0.7086843 0.4058519 0.0031426 0.6876602
0.6963269 0.5745585 0.6573101 0.6492243 0.3718001 0.0028789 0.6299642
9
```

Figure 66. Input listing (matrix addition)

Output

Description

The resultant matrix is printed for any size array as a general matrix regardless of the storage mode. Each page is headed with the matrix code number, dimensions, and storage mode. Columns and rows are headed with their respective number. The code number for the output matrix is derived by adding the code numbers for the input matrices.

Sample

The output listing for the sample problem is shown in Figure 67.

Program Modification

Matrix size can be increased or reduced by making the following changes in ADSAM:

1. Modify the DIMENSION statement to reflect the number of elements for A, B, and R.
2. Insert the same number in the third parameter of the two CALL MATIN statements (20 and 45).

MATRIX 1 8 ROWS 11 COLUMNS STORAGE MODE 0 PAGE 1

COLUMN	1	2	3	4	5	6
ROW 1	0.760101E 00	0.627180E 00	0.100000E 01	0.708684E 00	0.405852E 00	0.314260E-02
ROW 2	0.664408E 00	0.100000E 01	0.627180E 00	0.619465E 00	0.354757E 00	0.274700E-02
ROW 3	0.100000E 01	0.664408E 00	0.760101E 00	0.750750E 00	0.429942E 00	0.332910E-02
ROW 4	0.696327E 00	0.574558E 00	0.657310E 00	0.649224E 00	0.371800E 00	0.287890E-02
ROW 5	0.744684E 00	0.614460E 00	0.702958E 00	0.694311E 00	0.397620E 00	0.307890E-02
ROW 6	0.675177E 00	0.557107E 00	0.637345E 00	0.629505E 00	0.360507E 00	0.279150E-02
ROW 7	0.332510E-02	0.274700E-02	0.314260E-02	0.310390E-02	0.177760E-02	0.100000E 01
ROW 8	0.429942E 00	0.354757E 00	0.405852E 00	0.400859E 00	0.100000E 01	0.177760E-02

MATRIX 1 8 ROWS 11 COLUMNS STORAGE MODE 0 PAGE 2

COLUMN	7	8	9	10	11
ROW 1	0.687660E 00	0.675177E 00	0.863591E 00	0.744684E 00	0.696327E 00
ROW 2	0.601088E 00	0.557107E 00	0.712573E 00	0.614460E 00	0.574558E 00
ROW 3	0.728479E 00	0.637345E 00	0.815202E 00	0.702958E 00	0.657310E 00
ROW 4	0.629964E 00	0.629505E 00	0.805174E 00	0.694311E 00	0.649224E 00
ROW 5	0.673713E 00	0.360507E 00	0.461110E 00	0.397620E 00	0.371800E 00
ROW 6	0.610830E 00	0.279150E-02	0.357050E-02	0.307890E-02	0.287890E-02
ROW 7	0.301190E-02	0.610830E 00	0.781287E 00	0.673713E 00	0.629964E 00
ROW 8	0.388967E 00	0.100000E 01	0.724121E 00	0.624418E 00	0.583870E 00

MATRIX 2 8 ROWS 11 COLUMNS STORAGE MODE 0 PAGE 1

COLUMN	1	2	3	4	5	6
ROW 1	0.750750E 00	0.619465E 00	0.708684E 00	0.100000E 01	0.400859E 00	0.310390E-02
ROW 2	0.744684E 00	0.614460E 00	0.702958E 00	0.694311E 00	0.397620E 00	0.307890E-02
ROW 3	0.863591E 00	0.712573E 00	0.815202E 00	0.805174E 00	0.461110E 00	0.357050E-02
ROW 4	0.696327E 00	0.574558E 00	0.657310E 00	0.649224E 00	0.371800E 00	0.287890E-02
ROW 5	0.675177E 00	0.557107E 00	0.637345E 00	0.629505E 00	0.360507E 00	0.279150E-02
ROW 6	0.744684E 00	0.614460E 00	0.702958E 00	0.694311E 00	0.397620E 00	0.307890E-02
ROW 7	0.863591E 00	0.712573E 00	0.815202E 00	0.805174E 00	0.461110E 00	0.357050E-02
ROW 8	0.760101E 00	0.627180E 00	0.100000E 01	0.708684E 00	0.405852E 00	0.314260E-02

MATRIX 2 8 ROWS 11 COLUMNS STORAGE MODE 0 PAGE 2

COLUMN	7	8	9	10	11
ROW 1	0.679201E 00	0.724121E 00	0.100000E 01	0.798668E 00	0.746805E 00
ROW 2	0.673713E 00	0.624418E 00	0.798668E 00	0.100000E 01	0.643979E 00
ROW 3	0.781287E 00	0.583870E 00	0.746805E 00	0.643979E 00	0.100000E 01
ROW 4	0.629964E 00	0.100000E 01	0.664408E 00	0.760101E 00	0.750750E 00
ROW 5	0.610830E 00	0.728479E 00	0.601088E 00	0.687660E 00	0.679201E 00
ROW 6	0.673713E 00	0.429942E 00	0.354757E 00	0.405852E 00	0.400859E 00
ROW 7	0.781287E 00	0.760101E 00	0.627180E 00	0.100000E 01	0.708684E 00
ROW 8	0.687660E 00	0.696327E 00	0.574558E 00	0.657310E 00	0.649224E 00

MATRIX 3 8 ROWS 11 COLUMNS STORAGE MODE 0 PAGE 1

COLUMN	1	2	3	4	5	6
ROW 1	0.151085E 01	0.124664E 01	0.170868E 01	0.170868E 01	0.806711E 00	0.624650E-02
ROW 2	0.140909E 01	0.161446E 01	0.133014E 01	0.131378E 01	0.752378E 00	0.582590E-02
ROW 3	0.186359E 01	0.137698E 01	0.157530E 01	0.155592E 01	0.891052E 00	0.689960E-02
ROW 4	0.139265E 01	0.114912E 01	0.131462E 01	0.129845E 01	0.743600E 00	0.575780E-02
ROW 5	0.141986E 01	0.117157E 01	0.134030E 01	0.132382E 01	0.758127E 00	0.587040E-02
ROW 6	0.141986E 01	0.117157E 01	0.134030E 01	0.132382E 01	0.758127E 00	0.587040E-02
ROW 7	0.866920E 00	0.715320E 00	0.818345E 00	0.808278E 00	0.462887E 00	0.100357E 01
ROW 8	0.119004E 01	0.981938E 00	0.140585E 01	0.110954E 01	0.140585E 01	0.492020E-02

MATRIX 3 8 ROWS 11 COLUMNS STORAGE MODE 0 PAGE 2

COLUMN	7	8	9	10	11
ROW 1	0.136686E 01	0.139930E 01	0.186359E 01	0.154335E 01	0.144313E 01
ROW 2	0.127480E 01	0.118152E 01	0.151124E 01	0.161446E 01	0.121854E 01
ROW 3	0.150977E 01	0.122122E 01	0.156201E 01	0.134694E 01	0.165731E 01
ROW 4	0.125993E 01	0.162950E 01	0.146958E 01	0.145441E 01	0.139997E 01
ROW 5	0.128454E 01	0.108899E 01	0.106220E 01	0.108528E 01	0.105100E 01
ROW 6	0.128454E 01	0.432734E 00	0.358328E 00	0.408931E 00	0.403738E 00
ROW 7	0.784299E 00	0.137093E 01	0.140847E 01	0.167371E 01	0.133865E 01
ROW 8	0.107663E 01	0.169633E 01	0.129868E 01	0.128173E 01	0.123309E 01

END OF CASE

Figure 67. Output listing (matrix addition)

The output listing is set for 60 print lines per page, 132 print positions across the page, and double spacing. This can be changed by means of the last three parameters in the three-call MXOUT statements in ADSAM (statements 40, 80, 90).

Operating Instructions

The matrix addition sample program is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Error Messages

The following error conditions will result in messages:

1. Reserved storage area is too small for matrix -- DIMENSIONED AREA TOO SMALL FOR INPUT MATRIX (matrix code no.). GO ON TO NEXT CASE.
2. Input matrices do not have the same dimensions -- MATRIX DIMENSIONS NOT CONSISTENT. GO ON TO NEXT CASE.
3. Number of data cards does not correspond to that required by parameter card -- INCORRECT NUMBER OF DATA CARDS FOR MATRIX (matrix code no.). EXECUTION TERMINATED.

Error conditions 1 and 2 allow the computer run to continue. Error condition 3, however, terminates execution and requires another run to process succeeding cases.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 36 seconds.

```
C                                                                       ADSA  10
C     ......................................................................ADSA  20
C                                                                       ADSA  30
C        SAMPLE MAIN PROGRAM FOR MATRIX ADDITION - ADSAM                ADSA  40
C                                                                       ADSA  50
C        PURPOSE                                                        ADSA  60
C           MATRIX ADDITION SAMPLE PROGRAM                              ADSA  70
C                                                                       ADSA  80
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  ADSA  90
C           MADD                                                        ADSA 100
C           MATIN                                                       ADSA 110
C           MXOUT                                                       ADSA 120
C           LOC                                                         ADSA 130
C                                                                       ADSA 140
C        METHOD                                                         ADSA 150
C           TWO INPUT MATRICES ARE READ FROM THE STANDARD INPUT DEVICE. ADSA 160
C           THEY ARE ADDED AND THE RESULTANT MATRIX IS LISTED ON        ADSA 170
C           THE STANDARD OUTPUT DEVICE. THIS CAN BE REPEATED FOR ANY    ADSA 180
C           NUMBER OF PAIRS OF MATRICES UNTIL A BLANK CARD IS           ADSA 190
C           ENCOUNTERED                                                 ADSA 200
C                                                                       ADSA 210
C     ......................................................................ADSA 220
C                                                                       ADSA 230
C        MATRICES ARE DIMENSIONED FOR 1000 ELEMENTS. THEREFORE, PRODUCT ADSA 240
C        OF NUMBER OF ROWS BY NUMBER OF COLUMNS CANNOT EXCEED 1000.     ADSA 250
C                                                                       ADSA 260
      DIMENSION A(1000),B(1000),R(1000)                                 ADSA 270
C                                                                       ADSA 280
   10 FORMAT(1H1,15HMATRIX ADDITION)                                    ADSA 290
   11 FORMAT(1H0,44HDIMENSIONED AREA TOO SMALL FOR INPUT MATRIX ,I4)    ADSA 300
   12 FORMAT(1H0,20HEXECUTION TERMINATED)                               ADSA 310
   13 FORMAT(1H0,32HMATRIX DIMENSIONS NOT CONSISTENT)                   ADSA 320
   14 FORMAT(1H0,42HINCORRECT NUMBER OF DATA CARDS FOR MATRIX ,I4)      ADSA 330
   15 FORMAT(1H0,18HGO ON TO NEXT CASE)                                 ADSA 340
   16 FORMAT(1H0,11HEND OF CASE)                                        ADSA 350
C                                                                       ADSA 360
C     ......................................................................ADSA 370
C                                                                       ADSA 380
      WRITE(6,10)                                                       ADSA 390
   20 CALL MATIN(ICODA,A,1000,NA,MA,MSA,IER)                            ADSA 400
      IF( NA ) 25,95,25                                                 ADSA 410
   25 IF(IER-1) 40,30,35                                                ADSA 420
   30 WRITE(6,11) ICODA                                                 ADSA 430
      GO TO 45                                                          ADSA 440
   35 WRITE(6,14) ICODA                                                 ADSA 450
   37 WRITE(6,12)                                                       ADSA 460
      GO TO 95                                                          ADSA 470
   40 CALL MXOUT(ICODA,A,NA,MA,MSA,60,120,2)                            ADSA 480
   45 CALL MATIN(ICODB,B,1000,NB,MB,MSB,IER)                            ADSA 490
      IF(IER-1) 60,50,55                                                ADSA 500
   50 WRITE(6,11) ICODB                                                 ADSA 510
      WRITE(6,15)                                                       ADSA 520
      GO TO 20                                                          ADSA 530
   55 WRITE(6,14) ICODB                                                 ADSA 540
      GO TO 37                                                          ADSA 550
   60 IF(NA-NB) 75,70,75                                                ADSA 560
   70 IF(MA-MB) 75,80,75                                                ADSA 570
   75 WRITE(6,13)                                                       ADSA 580
      WRITE(6,15)                                                       ADSA 590
      GO TO 20                                                          ADSA 600
   80 CALL MXOUT(ICODB,B,NB,MB,MSB,60,120,2)                            ADSA 610
      ICODR=ICODA+ICODB                                                 ADSA 620
      CALL MADD(A,B,R,NA,MA,MSA,MSB)                                    ADSA 630
      MSR=MSA                                                           ADSA 640
      IF(MSA-MSB) 90,90,85                                              ADSA 650
   85 MSR=MSB                                                           ADSA 660
   90 CALL MXOUT(ICODR,R,NA,MA,MSR,60,120,2)                            ADSA 670
      WRITE(6,16)                                                       ADSA 680
      GO TO 20                                                          ADSA 690
   95 RETURN                                                            ADSA 700
      END                                                               ADSA 710
```

NUMERICAL QUADRATURE INTEGRATION

Problem Description

The tabulated values of a function for a given spacing are integrated. Multiple sets of tabulated values may be processed.

Program

Description

The numerical quadrature integration program consists of a main routine, QDINT, and one subroutine, QSF, from the Scientific Subroutine Package.

Capacity

Up to 500 tabulated values of a function may be integrated.

Input

Control Cards

Each integration requires a parameter card with the following format:

Columns	Contents	For Sample Problem
1- 5	Up to 5-digit numeric identification code	12345
6-10	Number of tabulated values for this function	0020
11-20	Interval between tabulated values	1.0

The first two parameters consist of up to five digits with no decimal point (FORMAT (2I5)). Note that the second parameter may not exceed 500. The third parameter consists of up to ten digits with decimal point. If no decimal point is included, it is assumed that the decimal point is to the right of the field (FORMAT (F10.0)). A blank card following the last set of data terminates the run.

Data Cards

Data cards are assumed to be seven fields of ten columns each. The decimal point may appear anywhere in the field. The number in each field may be preceded by blanks. Columns 71 through 80 of the data cards may be used for identification, sequence numbering, and so on. If there are more than seven tabulated values, the values should continue from card to card with seven values per card until the number of values specified in the parameter card has been reached.

Deck Setup

The deck setup is shown in Figure 68.

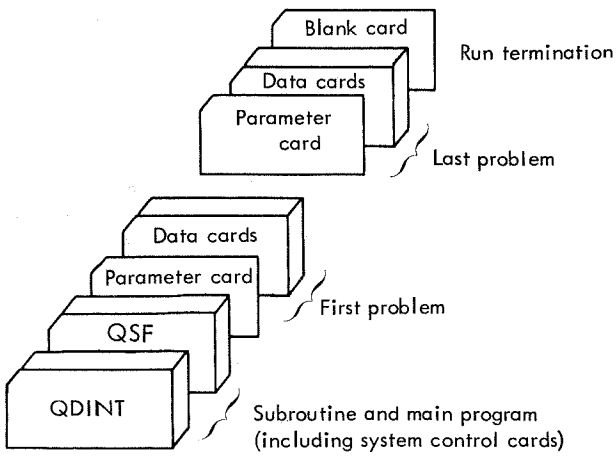


Figure 68 Deck setup (numerical quadrature integration)

Sample

A listing of input cards for the sample program is presented in Figure 69.

```
/DATA                                                                   10
12345    20    1.0                                                      20
         2.0       2.0       2.0       2.0       2.0       2.0       2.0 30
         2.0       2.0       2.0       2.0       2.0       2.0       2.0 40
         2.0       2.0       2.0       2.0       2.0       2.0           50
  543    10        1.0                                                   60
         1.0       2.0       3.0       4.0       5.0       6.0       7.0 70
         8.0       9.0      10.0                                         80
                                                                         90
```

Figure 69. Input listing (numerical quadrature integration)

Output

Description

The identification code number, number of tabulated input values, the interval for the tabulated values, and the resultant integral values at each step are printed.

Sample

The output listing for the sample program is shown in Figure 70.

```
INTEGRATION OF TABULATED VALUES FOR DY/DX USING SUBROUTINE QSF

FUNCTION  12345      20 TABULATED VALUES      INTERVAL = 0.10000000E 01

      RESULTANT VALUE OF INTEGRAL AT EACH STEP IS
 0.0          0.19999990E 01 0.39999990E 01 0.59999981E 01 0.79999981E 01 0.99999971E 01
 0.11999997E 02 0.13999996E 02 0.15999996E 02 0.17999985E 02 0.19999985E 02 0.21999969E 02
 0.23999969E 02 0.25999954E 02 0.27999954E 02 0.29999939E 02 0.31999939E 02 0.33999924E 02
 0.35999924E 02 0.37999909E 02


INTEGRATION OF TABULATED VALUES FOR DY/DX USING SUBROUTINE QSF

FUNCTION    543      10 TABULATED VALUES      INTERVAL = 0.10000000E 01

      RESULTANT VALUE OF INTEGRAL AT EACH STEP IS
 0.0          0.14999955E 01 0.39999990E 01 0.74999876E 01 0.11999997E 02 0.17499985E 02
 0.23999985E 02 0.31499969E 02 0.39999969E 02 0.49499954E 02
```

Figure 70. Output listing (numerical quadrature integration)

Program Modification

The maximum number of tabulated values acceptable to the sample program may be changed by modifying the dimension statement in QDINT. The format of the parameter cards and data cards may be changed by modifying statements 10 and 40, respectively, in QDINT.

Operating Instructions

The numerical quadrature integration sample program is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Error Messages

The following conditions will result in error messages:

1. The number of tabulated values specified in the parameter card is less than or equal to one -- ILLEGAL CONDITION. NUMBER OF TABULATED VALUES IS LESS THAN THREE. The program will go on to the next set of data.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is six seconds.

```
C                                                                       QDIN  10
C     ...................................................................QDIN  20
C                                                                       QDIN  30
C        SAMPLE PROGRAM FOR INTEGRATION OF A TABULATED FUNCTION BY      QDIN  40
C        NUMERICAL QUADRATURE - QDINT                                   QDIN  50
C                                                                       QDIN  60
C        PURPOSE                                                        QDIN  70
C           INTEGRATES A SET OF TABULATED VALUES FOR F(X) GIVEN THE     QDIN  80
C           NUMBER OF VALUES AND THEIR SPACING                          QDIN  90
C                                                                       QDIN 100
C        REMARKS                                                        QDIN 110
C           THE NUMBER OF VALUES MUST BE MORE THAN TWO AND THE SPACING  QDIN 120
C           GREATER THAN ZERO                                           QDIN 130
C                                                                       QDIN 140
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  QDIN 150
C           QSF                                                         QDIN 160
C                                                                       QDIN 170
C        METHOD                                                         QDIN 180
C           READS CONTROL CARD CONTAINING THE CODE NUMBER, NUMBER OF    QDIN 190
C           VALUES, AND THE SPACING OF THE FUNCTION VALUES CONTAINED    QDIN 200
C           ON THE FOLLOWING DATA CARDS. DATA CARDS ARE THEN READ AND   QDIN 210
C           INTEGRATION IS PERFORMED. MORE THAN ONE CONTROL CARD AND    QDIN 220
C           CORRESPONDING DATA CAN BE INTEGRATED IN ONE RUN. EXECUTION  QDIN 230
C           IS TERMINATED BY A BLANK CONTROL CARD.                      QDIN 240
C                                                                       QDIN 250
C     ...................................................................QDIN 260
C                                                                       QDIN 270
C        THE FOLLOWING DIMENSION MUST BE AS LARGE AS THE MAXIMUM NUMBER QDIN 280
C        OF TABULATED VALUES TO BE INTEGRATED                           QDIN 290
C                                                                       QDIN 300
      DIMENSION Z(500)                                                  QDIN 310
C                                                                       QDIN 320
C     ...................................................................QDIN 330
C                                                                       QDIN 340
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  QDIN 350
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      QDIN 360
C        STATEMENT WHICH FOLLOWS.                                       QDIN 370
C                                                                       QDIN 380
C     DOUBLE PRECISION Z,SPACE                                          QDIN 390
C                                                                       QDIN 400
C     ...................................................................QDIN 410
C                                                                       QDIN 420
C                                                                       QDIN 430
   10 FORMAT (2I5,F10.0)                                                QDIN 440
   20 FORMAT(1H1,62HINTEGRATIGN OF TABULATED VALUES FOR DY/DX USING SUBRQDIN 450
     1OUTINE QSF//1H ,10HFUNCTION  ,I5,3X,I5,17H TABULATED VALUES,      QDIN 460
     25X,10HINTERVAL =,E15.8//)                                         QDIN 470
   22 FORMAT(1H ,17HILLEGAL CONDITION/)                                 QDIN 480
   23 FORMAT(1H ,45HNUMBER OF TABULATED VALUES IS LESS THAN THREE)      QDIN 490
   30 FORMAT(1H ,7X,'RESULTANT VALUE OF INTEGRAL AT EACH STEP IS ',     QDIN 500
     1(1H ,6E15.8))                                                     QDIN 510
   32 FORMAT(7F10.0)                                                    QDIN 520
C                                                                       QDIN 530
   35 READ(5,10)ICOD,NUMBR,SPACE                                        QDIN 540
      IF(ICOD+NUMBR)70,70,38                                            QDIN 550
   38 WRITE(6,20)ICOD,NUMBR,SPACE                                       QDIN 560
   50 READ(5,32)(Z(I),I=1,NUMBR)                                        QDIN 570
      IF(NUMBR-3)100,55,55                                              QDIN 580
   55 CALL QSF(SPACE,Z,Z,NUMBR)                                         QDIN 590
   60 WRITE(6,30)(Z(I),I=1,NUMBR)                                       QDIN 600
      GO TO 35                                                          QDIN 610
   70 RETURN                                                            QDIN 620
  100 WRITE(6,22)                                                       QDIN 630
      WRITE (6,23)                                                      QDIN 640
      GO TO 35                                                          QDIN 650
  200 WRITE(6,22)                                                       QDIN 660
      GO TO 35                                                          QDIN 670
      END
```

RUNGE-KUTTA INTEGRATION

Problem Description

A differential equation of the form:

$$\frac{dy}{dx} = f\ (x, y)$$

is integrated with initial conditions as specified in a parameter card. The differential equation is defined in the form of a function subprogram that is provided by the user.

Program

Description

The Runge-Kutta integration program consists of a main routine RKINT, one subroutine RK2 from the Scientific Subroutine Package, and one user-supplied function subprogram FUN, which defines the differential equation to be integrated.

Capacity

Up to 500 values of the integral may be tabulated.

Input

Control Cards

Each integration requires a parameter card with the following format:

Columns	Contents	For Sample Problem
1-10	Initial value of $X = X_0$	1.0
11-20	Initial value of $Y = Y(X_0)$	0.0
21-30	Step size	0.01
31-35	Number of steps required between tabulated values	10
36-40	Total number of tabulated values required	30

The first three parameters consist of up to ten digits with decimal point. If no decimal point is included, it is assumed that the decimal point is to the right of the field.

(FORMAT (F10.0))

The last two parameters consist of up to four digits plus a blank with no decimal point.

(FORMAT (I5))

Multiple parameter cards may be used. A blank card terminates the run.

Data Cards

None.

Deck Setup

The deck setup is shown in Figure 71.

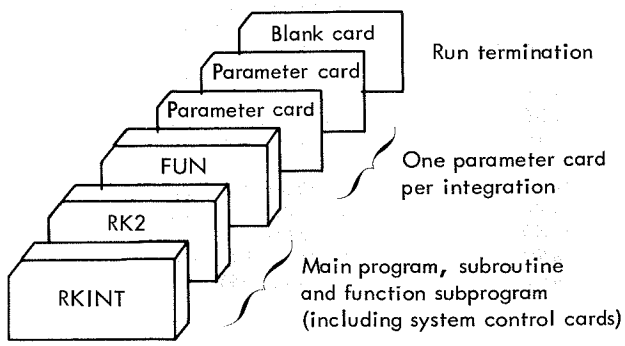


Figure 71. Deck setup (Runge-Kutta integration)

Sample

A listing of the input cards for the sample problem is presented in Figure 72.

```
/DATA
      1.0       0.0       .01   10   30                                  10
                                                                         20
                                                                         30
```

Figure 72. Input listing (Runge-Kutta integration)

Output

Description

The values for the initial conditions and the tabulated values of the integral are printed.

Sample

The output listing for the sample problem is shown in Figure 73.

```
SOLUTION OF DY/DX=FUN(X,Y) BY RK2 SUBROUTINE

   H=  0.010   XO=  1.000   YO=  0.0

    X                    Y(X)


   1.10              0.95310390E-01
   1.20              0.18232232E 00
   1.30              0.26236582E 00
   1.40              0.33647484E 00
   1.50              0.40546888E 00
   1.60              0.47000873E 00
   1.70              0.53063482E 00
   1.80              0.58779466E 00
   1.90              0.64186341E 00
   2.00              0.69315827E 00
   2.10              0.74194998E 00
   2.20              0.78847140E 00
   2.30              0.83292466E 00
   2.40              0.87548572E 00
   2.50              0.91630906E 00
   2.60              0.95553130E 00
   2.70              0.99327296E 00
   2.80              0.10296392E 01
   2.90              0.10647268E 01
   3.00              0.10986242E 01
   3.10              0.11314125E 01
   3.20              0.11631584E 01
   3.30              0.11939259E 01
   3.40              0.12237749E 01
   3.50              0.12527580E 01
   3.60              0.12809258E 01
   3.70              0.13083200E 01
   3.80              0.13349857E 01
   3.90              0.13609591E 01
   4.00              0.13862734E 01
```

Figure 73. Output listing (Runge-Kutta integration)

Program Modification

The function subprogram FUN may be replaced by any function subprogram having the same name and parameter list. In this way, the user may define any desired first-order differential equation. The maximum number of tabulated values may be changed by altering the dimension statement in the sample program RKINT.

Operating Instructions

The sample program for Runge-Kutta integration is a standard FORTRAN program. Special operating instructions are not required. Data set reference 5 is used for input, and data set reference 6 is used for output.

Error Messages

None.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is twelve seconds.

```
C                                                                       RKIN  10
C     ..................................................................RKIN  20
C                                                                       RKIN  30
C        SAMPLE PROGRAM FOR RUNGE-KUTTA INTEGRATION OF A GIVEN FUNCTION RKIN  40
C        WITH TABULATED OUTPUT - RKINT                                  RKIN  50
C                                                                       RKIN  60
C        PURPOSE                                                        RKIN  70
C           INTEGRATES THE FUNCTION SUBPROGRAM FUN USING THE INITIAL    RKIN  80
C           CONDITIONS CONTAINED IN CONTROL CARDS.  PRODUCES TABULATED  RKIN  90
C           OUTPUT.                                                     RKIN 100
C                                                                       RKIN 110
C        REMARKS                                                        RKIN 120
C           NONE                                                        RKIN 130
C                                                                       RKIN 140
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  RKIN 150
C           RK2                                                         RKIN 160
C           FUN - USER-SUPPLIED FUNCTION SUBPROGRAM GIVING              RKIN 170
C                 DY/DX=FUN(X,Y)                                        RKIN 180
C                                                                       RKIN 190
C        METHOD                                                         RKIN 200
C           READS CONTROL CARD CONTAINING INITIAL VALUES OF X AND Y,    RKIN 210
C           STEP SIZE, NUMBER OF STEPS DESIRED BETWEEN TABULATED VALUES,RKIN 220
C           AND NUMBER OF TABULATED VALUES REQUIRED. PROGRAM THEN ENTERSRKIN 230
C           RK2 TO PERFORM INTEGRATION. MULTIPLE CONTROL CARDS CAN BE   RKIN 240
C           USED ON THE SAME FUNCTION.                                  RKIN 250
C                                                                       RKIN 260
C     ..................................................................RKIN 270
C                                                                       RKIN 280
C        THE FOLLOWING DIMENSION MUST BE AS LARGE AS THE MAXIMUM        RKIN 290
C        NUMBER OF TABULATED VALUES DESIRED                             RKIN 300
C                                                                       RKIN 310
      DIMENSION A(500)                                                  RKIN 320
C                                                                       RKIN 330
      EXTERNAL FUN                                                      RKIN 340
C                                                                       RKIN 350
C        ...............................................................RKIN 360
C                                                                       RKIN 370
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  RKIN 380
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      RKIN 390
C        STATEMENT WHICH FOLLOWS.                                       RKIN 400
C                                                                       RKIN 410
C     DOUBLE PRECISION H,XO,YO,A,FUN                                    RKIN 420
C                                                                       RKIN 430
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    RKIN 440
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      RKIN 450
C        ROUTINE.                                                       RKIN 460
C                                                                       RKIN 470
C        USER FUNCTION SUBPROGRAM, FUN, MUST BE IN DOUBLE PRECISION.    RKIN 480
C                                                                       RKIN 490
C        ...............................................................RKIN 500
C                                                                       RKIN 510
    1 FORMAT (3F10.0,2I5)                                               RKIN 520
    2 FORMAT(1H1,7X,44HSOLUTION OF DY/DX=FUN(X,Y) BY RK2 SUBROUTINE//   RKIN 530
     11H ,1CX,2HH=,F7.3,2X,3HXO=,F7.3, 2X,3HYO=,F7.3//1H ,12X,1HX,18X,  RKIN 540
     24HY(X)//)                                                         RKIN 550
    3 FORMAT(1H ,1CX,F5.2,10X,E15.8)                                    RKIN 560
C                                                                       RKIN 570
C        READ CONTROL CARD CONTAINING ITEMS LISTED UNDER METHOD.        RKIN 580
C                                                                       RKIN 590
   10 READ(5,1)XO,YO,H,JNT,IENT                                         RKIN 600
C                                                                       RKIN 610
C        CHECK IF CARD IS BLANK. IF SO, RETURN.                         RKIN 620
C                                                                       RKIN 630
      IF(IENT)20,40,20                                                  RKIN 640
C                                                                       RKIN 650
C        WRITE HEADING INFORMATION.                                     RKIN 660
C                                                                       RKIN 670
   20 WRITE(6,2)H,XO,YO                                                 RKIN 680
C                                                                       RKIN 690
C        PERFORM INTEGRATION                                            RKIN 700
C                                                                       RKIN 710
      CALL RK2(FUN,H,XO,YO,JNT,IENT,A)                                  RKIN 720
C                                                                       RKIN 730
C        WRITE OUTPUT                                                   RKIN 740
C                                                                       RKIN 750
      STEP=FLOAT(JNT)*H                                                 RKIN 760
      X=XO                                                              RKIN 770
```

```
      DO 30 I=1,IENT                                                    RKIN 780
      X=X+STEP                                                          RKIN 790
   30 WRITE(6,31)X,A(I)                                                 RKIN 800
C                                                                       RKIN 810
C        GO BACK AND CHECK FOR ADDITIONAL CONTROL CARD.                 RKIN 820
C                                                                       RKIN 830
      GO TO 10                                                          RKIN 840
   40 RETURN                                                            RKIN 850
      END                                                               RKIN 860
```

POLYNOMIAL ROOTS

Problem Description

The real and complex roots are computed for a real polynomial with given coefficients. Multiple sets of coefficients may be processed.

Program

Description

The polynomial roots sample program consists of a main routine, SMPRT, and one subroutine POLRT from the Scientific Subroutine Package.

Capacity

Roots for polynomials of order 36 or less may be computed.

Input

Control Cards

Each set of data requires a control card with the following format:

Columns	Contents	For Sample Problem
1	Blank	
2-5	Up to four-digit identification code	360
6-8	Blank	
9-10	Order of polynomial	9

The first parameter consists of up to four digits without decimal point. The second parameter consists of up to two digits with no decimal point. The order of the polynomial must be less than or equal to 36.

Data Cards

Data cards are assumed to have seven fields of ten columns each. The decimal point may appear anywhere in the field. If no decimal point is included, it is assumed to be to the right of the field. The number in each field may be preceded by blanks. Columns 71 to 80 of the data cards may be used for identification, sequence numbering, and so on. If there are more than seven coefficients, the values should continue from card to card with seven values per card until the number of values has been reached that is one greater than the order of the polynomial.

The first coefficient is for the constant term of the polynomial and the last coefficient for the highest order term. Fields with zero coefficients may be left blank.

Deck Setup

The deck setup is shown in Figure 74.

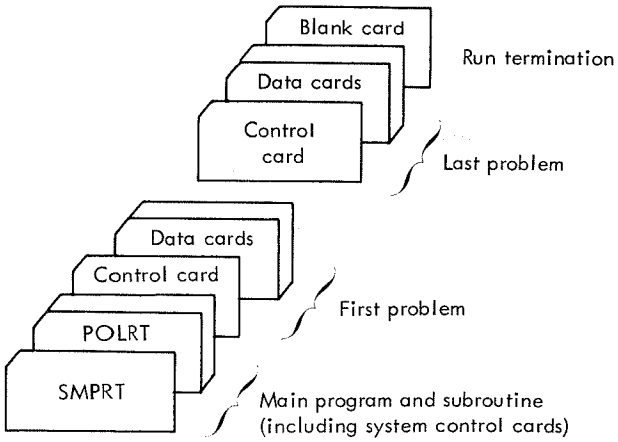


Figure 74. Deck setup (polynomial roots)

Sample

A listing of the input cards for the sample program is shown in Figure 75.

```
/DATA
 360    9
     -1.0                                                          10
                 1.0                                   1.0         20
                                                                   30
                                                                   40
                                                                   50
```

Figure 75. Input listing (polynomial roots)

Output

Description

The identification code, the polynomial order, the input coefficients, and the real and complex roots are printed.

Sample

The output listing of the sample program is shown in Figure 76.

```
REAL AND COMPLEX ROOTS OF A POLYNOMIAL USING SUBROUTINE POLRT

FOR POLYNOMIAL   360  OF ORDER   9

THE INPUT COEFFICIENTS ARE

 -0.1000000E 01   0.0            0.0            0.0            0.0            0.0
  0.1000000E 01   0.0            0.0            0.1000000E 01

     REAL ROOT       COMPLEX ROOT

   0.7206228E 00   0.7609009E 00
   0.7206228E 00  -0.7609009E 00
  -0.1019271E 01  -0.2436270E 00
  -0.1019271E 01   0.2436270E 00
   0.9105257E 00   0.0
  -0.4552628E 00  -0.7885384E 00
  -0.4552628E 00   0.7885384E 00
   0.2986479E 00  -0.1004528E 01
   0.2986479E 00   0.1004528E 01
```

Figure 76. Output listing (polynomial roots)

Program Modification

The maximum order of the polynomial is fixed by the subroutine POLRT. The sample program can accept polynomials up to the maximum of 36th order, which is allowed by the subroutine. The format of the control card and data cards may be changed by modifying statements 10 and 40, respectively, in SMPRT.

Operating Instructions

The polynomial roots sample program is a standard FORTRAN program. Special operating instructions are not required. Data set 5 is used for input, and data set 6 is used for output.

Error Messages

The following conditions will result in error messages:

1. The order of the polynomial specified in the control card is less than one -- ORDER OF POLYNOMIAL LESS THAN ONE.

   The program will go on to the next set of data.

2. The order of the polynomial specified in the control card is greater than 36 -- ORDER OF POLYNOMIAL GREATER THAN 36.

   The program will go on to the next set of data.

3. The subroutine POLRT is unable to determine a root after 500 iterations on eight different starting values -- UNABLE TO DETERMINE ROOT. THOSE ALREADY FOUND ARE....

   The program will print all the roots that were computed and then go on to the next set of data.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is eleven seconds.

```
C                                                                       SMPR  10
C     ..................................................................SMPR  20
C                                                                       SMPR  30
C        SAMPLE PROGRAM FOR REAL AND COMPLEX ROOTS OF A REAL POLY-      SMPR  40
C        NOMIAL - SMPRT                                                 SMPR  50
C                                                                       SMPR  60
C        PURPOSE                                                        SMPR  70
C           COMPUTES THE REAL AND COMPLEX ROOTS OF A REAL POLYNOMIAL    SMPR  80
C           WHOSE COEFFICIENTS ARE INPUT.                               SMPR  90
C                                                                       SMPR 100
C        REMARKS                                                        SMPR 110
C           THE ORDER OF THE POLYNOMIAL MUST BE GREATER THAN ONE AND    SMPR 120
C           LESS THAN THIRTY SEVEN                                      SMPR 130
C                                                                       SMPR 140
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SMPR 150
C           POLRT                                                       SMPR 160
C                                                                       SMPR 170
C        METHOD                                                         SMPR 180
C           READS A CONTROL CARD CONTAINING THE IDENTIFICATION CODE AND SMPR 190
C           THE ORDER OF THE POLYNOMIAL WHOSE COEFFICIENTS ARE          SMPR 200
C           CONTAINED ON THE FOLLOWING DATA CARDS. THE COEFFICIENTS     SMPR 210
C           ARE THEN READ AND THE ROOTS ARE COMPUTED.                   SMPR 220
C           MORE THAN ONE CONTROL CARD AND CORRESPONDING DATA CAN BE    SMPR 230
C           PROCESSED. EXECUTION IS TERMINATED BY A BLANK CONTROL CARD. SMPR 240
C                                                                       SMPR 250
C     ..................................................................SMPR 260
C                                                                       SMPR 270
      DIMENSION A(37),W(37),ROOTR(37),ROOTI(37)                         SMPR 280
C                                                                       SMPR 290
C        ...............................................................SMPR 300
C                                                                       SMPR 310
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  SMPR 320
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      SMPR 330
C        STATEMENT WHICH FOLLOWS.                                       SMPR 340
C                                                                       SMPR 350
C     DOUBLE PRECISION A,W,ROOTR,ROOTI                                  SMPR 360
C                                                                       SMPR 370
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    SMPR 380
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      SMPR 390
C        ROUTINE.                                                       SMPR 400
C                                                                       SMPR 410
C        ...............................................................SMPR 420
C                                                                       SMPR 430
    5 READ(5,10)ID,IORD                                                 SMPR 440
   10 FORMAT(1X,I4,3X,I2)                                               SMPR 450
      IF(ID+IORD)100,100,20                                             SMPR 460
   20 WRITE(6,30)ID,IORD                                                SMPR 470
   30 FORMAT(1H1,61HREAL AND COMPLEX ROOTS OF A POLYNOMIAL USING SUBROUTSMPR 480
     1INE POLRT///  17H FOR POLYNOMIAL  ,I4,2X,10HOF ORDER  ,I2//1H ,26HSMPR 490
     2THE INPUT COEFFICIENTS ARE//)                                     SMPR 500
      J=IORD+1                                                          SMPR 510
      READ(5,40)(A(I),I=1,J)                                            SMPR 520
   40 FORMAT(7F10.0)                                                    SMPR 530
      WRITE(6,50)(A(I),I=1,J)                                           SMPR 540
   50 FORMAT(6E16.7)                                                    SMPR 550
      CALL POLRT(A,W,IORD,ROOTR,ROOTI,IER)                              SMPR 560
      IF(IER-1)90,60,70                                                 SMPR 570
   60 WRITE(6,65)                                                       SMPR 580
   65 FORMAT(//1H ,33HORDER OF POLYNOMIAL LESS THAN ONE)                SMPR 590
      GO TO 5                                                           SMPR 600
   70 IF(IER-3)75,80,78                                                 SMPR 610
   75 WRITE(6,77)                                                       SMPR 620
   77 FORMAT(//1H ,35HORDER OF POLYNOMIAL GREATER THAN 36)              SMPR 630
      GO TO 5                                                           SMPR 640
   78 WRITE(6,79)                                                       SMPR 650
   79 FORMAT(//1H ,31H HIGH ORDER COEFFICIENT IS ZERO)                  SMPR 660
      GO TO 5                                                           SMPR 670
   80 WRITE(6,85)                                                       SMPR 680
   85 FORMAT(//1H ,49HUNABLE TO DETERMINE ROOT. THOSE ALREADY FOUND ARE)SMPR 690
   90 WRITE(6,95)                                                       SMPR 700
   95 FORMAT(//1H ,5X,9HREAL ROOT,6X,12HCOMPLEX ROOT//)                 SMPR 710
      DO 96 I=1,IORD                                                    SMPR 720
   96 WRITE(6,97)ROOTR(I),ROOTI(I)                                      SMPR 730
   97 FORMAT(1H ,2E16.7)                                                SMPR 740
      GO TO 5                                                           SMPR 750
  100 RETURN                                                            SMPR 760
      END                                                               SMPR 770
```

SOLUTION OF SIMULTANEOUS EQUATIONS

Problem Description

A solution is obtained for a set of simultaneous equations by the method of elimination using largest pivotal divisor. Both the input data and the solution values are printed. This procedure is repeated until all sets of input data have been processed.

Program

Description

The solution of the simultaneous equations program consists of a main routine, SOLN, and four subroutines, MATIN, SIMQ, MXOUT, and LOC. SIMQ and LOC are from the Scientific Subroutine Package. MATIN and MXOUT are sample subroutines for matrix input and output (see program listings).

Capacity

The program will solve up to 50 equations. This can easily be changed as described under "Program Modification" below.

Input

Control Cards

A parameter card with the following format must precede each matrix of coefficients:

Columns	Contents	For Sample Problem
1-2	Blank	
3-6	Up to four-digit identification code (numeric only)	1
7-10	Number of rows in matrix	10
11-14	Number of columns in matrix (same as number of rows)	10

Each matrix must be followed by a card with a 9-punch in column 1. This, in turn, is followed by the constant vector.

A blank card after the last set of input data terminates the run.

Data Cards

Data Cards are assumed to have seven fields of ten columns each. The decimal point may appear anywhere in a field. If no decimal point is included, it is assumed that the decimal point is to the right of the last digit. The number in each field may be preceded by blanks. Equation coefficients must be punched by row. A row may continue from card to card. However, each new row must start in the first field of the next card. The vector of constants is punched in continuous data fields following the 9 card. Columns 71-80 of data cards may be used for identification, sequence numbering, and so on.

Deck Setup

The deck setup is shown in Figure 77.

Sample

A listing of input cards for the sample problem is presented in Figure 78.

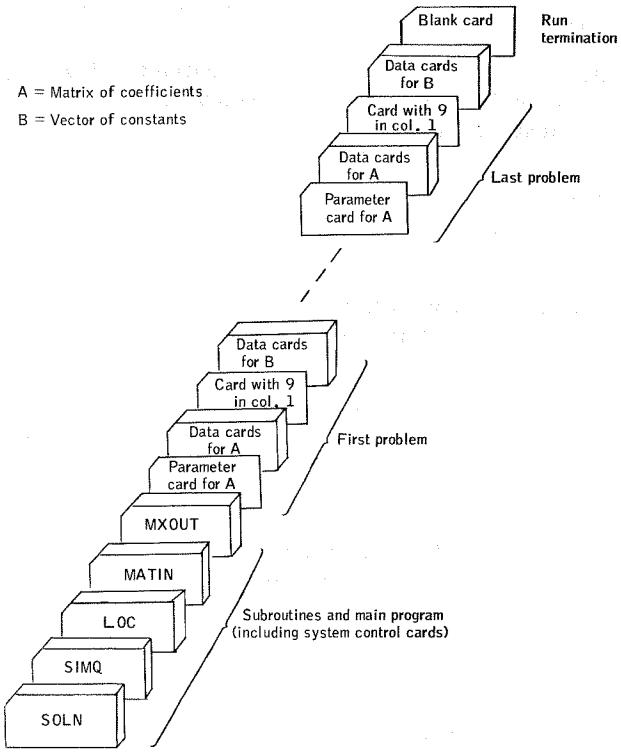


Figure 77. Deck setup (solution of simultaneous equations)

```
/DATA
  000100100010
 1.0000000 0.6644085 0.7601008 0.7507505 0.4299425 0.0033291 0.7284786
 0.6751766 0.8635910 0.7446845
 0.6644085 1.0000000 0.6271802 0.6194650 0.3547574 0.0027470 0.6010878
 0.5571068 0.7125728 0.6144597
 0.7601008 0.6271802 1.0000000 0.7086843 0.4058519 0.0031426 0.6876602
 0.6373449 0.8152021 0.7029582
 0.7507505 0.6194650 0.7086843 1.0000000 0.4008593 0.0031039 0.6792011
 0.6295047 0.8051740 0.6943108
 0.4299423 0.3547574 0.4058519 0.4008593 1.0000000 0.0017776 0.3889673
 0.3605070 0.4611099 0.3976204
 0.0033291 0.0027470 0.0031426 0.0031039 0.0017776 1.0000000 0.0030119
 0.0027915 0.0035705 0.0030789
 0.7284786 0.6010878 0.6876602 0.6792011 0.3889673 0.0030119 1.0000000
 0.6108296 0.7812874 0.6737132
 0.6751766 0.5571068 0.6373449 0.6295047 0.3605070 0.0027915 0.6108296
 1.0000000 0.7241215 0.6244183
 0.8635910 0.7125728 0.8152021 0.8051740 0.4611099 0.0035705 0.7812874
 0.7241215 1.0000000 0.7986682
 0.7446845 0.6144597 0.7029582 0.6943108 0.3976204 0.0030789 0.6737132
 0.6244183 0.7986682 1.0000000
9
 110.     -120.      10.      145.      -50.      44.2      -14.
  38.5      22.     1650.
```

Figure 78. Input listing (solution of simultaneous equations)

Output

Description

The original matrix is printed for any size array. Each page is headed with the matrix code number, dimensions, and storage mode (always 0 in this sample program). Columns and rows are headed with their respective number. The original vector of constants is also printed. The solution values are then listed. This output is given for each case to be processed.

Sample

The output listing for the sample problem is shown in Figure 79.

```
MATRIX    1      10 ROWS      10 COLUMNS      STORAGE MODE 0      PAGE  1
     COLUMN      1            2            3            4            5            6
  ROW   1   0.100000E 01   0.664408E 00   0.760101E 00   0.750750E 00   0.429942E 00   0.332910E-02
  ROW   2   0.664408E 00   0.100000E 01   0.627180E 00   0.619465E 00   0.354757E 00   0.274700E-02
  ROW   3   0.760101E 00   0.627180E 00   0.100000E 01   0.708684E 00   0.405852E 00   0.314260E-02
  ROW   4   0.750750E 00   0.619465E 00   0.708684E 00   0.100000E 01   0.400859E 00   0.310390E-02
  ROW   5   0.429942E 00   0.354757E 00   0.405852E 00   0.400859E 00   0.100000E 01   0.177760E-02
  ROW   6   0.332910E-02   0.274700E-02   0.314260E-02   0.310390E-02   0.177760E-02   0.100000E 01
  ROW   7   0.728479E 00   0.601088E 00   0.687660E 00   0.679201E 00   0.388967E 00   0.301190E-02
  ROW   8   0.675177E 00   0.557107E 00   0.637345E 00   0.629505E 00   0.360507E 00   0.279150E-02
  ROW   9   0.863591E 00   0.712573E 00   0.815202E 00   0.805174E 00   0.461110E 00   0.357050E-02
  ROW  10   0.744684E 00   0.614460E 00   0.702958E 00   0.694311E 00   0.397620E 00   0.307890E-02

MATRIX    1      10 ROWS      10 COLUMNS      STORAGE MODE 0      PAGE  2
     COLUMN      7            8            9           10
  ROW   1   0.728479E 00   0.675177E 00   0.863591E 00   0.744684E 00
  ROW   2   0.601088E 00   0.557107E 00   0.712573E 00   0.614460E 00
  ROW   3   0.687660E 00   0.637345E 00   0.815202E 00   0.702958E 00
  ROW   4   0.679201E 00   0.629505E 00   0.805174E 00   0.694311E 00
  ROW   5   0.388967E 00   0.360507E 00   0.461110E 00   0.397620E 00
  ROW   6   0.301190E-02   0.279150E-02   0.357050E-02   0.307890E-02
  ROW   7   0.100000E 01   0.610830E 00   0.781287E 00   0.673713E 00
  ROW   8   0.610830E 00   0.100000E 01   0.724121E 00   0.624418E 00
  ROW   9   0.781287E 00   0.724121E 00   0.100000E 01   0.798668E 00
  ROW  10   0.673713E 00   0.624418E 00   0.798668E 00   0.100000E 01

ORIGINAL B VECTOR

 1            0.110000E 03
 2           -0.120000E 03
 3            0.100000E 02
 4            0.145000E 03
 5           -0.500000E 02
 6            0.442000E 02
 7           -0.140000E 02
 8            0.385000E 02
 9            0.220000E 02
10            0.165000E 04

SOLUTION VALUES

 1           -0.283124E 03
 2           -0.567240E 03
 3           -0.516456E 03
 4           -0.299155E 02
 5           -0.179352E 03
 6            0.435176E 02
 7           -0.479274E 03
 8           -0.230430E 03
 9           -0.210172E 04
10            0.480574E 04

END OF CASE
```

Figure 79. Output listing (solution of simultaneous equations)

Program Modification

The size of the maximum problem to be solved can be increased or decreased by making the following changes in SOLN:

1. Modify the DIMENSION statement to reflect the maximum number of elements for matrix A (= N x N) and the maximum number of equations (N) for B.

2. Insert the dimension size for A in the third argument of the CALL MATIN statement (statement 25).

The matrix listing is set for 60 print lines per page, 120 print positions across the page, and double spacing. This can be changed by means of the last three arguments in the CALL MXOUT statement in SOLN (statement 65). The format of solution values can be altered by changing statement 21 in SOLN.

Operating Instructions

The sample program for the solution of simultaneous equations is a standard FORTRAN program. Special operating instructions are not required. Data set reference 5 is used for input, and data set reference 6 is used for output.

Error Messages

The following error conditions will result in messages:

1. Reserved storage area is too small for matrix -- DIMENSIONED AREA TOO SMALL FOR INPUT MATRIX (matrix code no.). GO ON TO NEXT CASE.
2. Matrix of coefficients is not square -- ROW AND COLUMN DIMENSIONS NOT EQUAL FOR MATRIX (matrix code no.). GO ON TO NEXT CASE.
3. Number of data cards does not correspond to that required by parameter card -- INCORRECT NUMBER OF DATA CARDS FOR MATRIX (matrix code no.). EXECUTION TERMINATED.
4. Singular input matrix -- MATRIX IS SINGULAR. GO ON TO NEXT CASE.

Error conditions 1, 2, and 4 allow the computer run to continue. Error condition 3, however, terminates execution and requires another run to process succeeding cases.

Timing

The execution time of this sample program on a System/360, Model 30, using an IBM 2540 Card Reader as input and an IBM 1403, Model 3 as output, is 21 seconds.

```
C                                                                       SOLN  10
C     ..................................................................SOLN  20
C                                                                       SOLN  30
C        SAMPLE MAIN PROGRAM - SOLN                                     SOLN  40
C                                                                       SOLN  50
C        PURPOSE                                                        SOLN  60
C           SOLUTION OF A SET OF SIMULTANEOUS EQUATIONS                 SOLN  70
C                                                                       SOLN  80
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  SOLN  90
C           SIMQ                                                        SOLN 100
C           MATIN                                                       SOLN 110
C           MXOUT                                                       SOLN 120
C           LOC                                                         SOLN 130
C                                                                       SOLN 140
C        METHOD                                                         SOLN 150
C           A MATRIX OF SIMULTANEOUS EQUATIONS COEFFICIENTS AND A VECTORSOLN 160
C           OF CONSTANTS ARE READ FROM THE STANDARD INPUT DEVICE. THE   SOLN 170
C           SOLUTION IS OBTAINED AND LISTED ON THE STANDARD OUTPUT      SOLN 180
C           DEVICE. THIS PROCEDURE IS REPEATED FOR OTHER SETS OF        SOLN 190
C           EQUATIONS UNTIL A BLANK CARD IS ENCOUNTERED.                SOLN 200
C                                                                       SOLN 210
C     ..................................................................SOLN 220
C                                                                       SOLN 230
C        MATRIX IS DIMENSIONED FOR 2500 ELEMENTS. THEREFORE, NUMBER OF  SOLN 240
C        EQUATIONS TO BE SOLVED CANNOT EXCEED 50 UNLESS DIMENSION       SOLN 250
C        STATEMENT IS CHANGED                                           SOLN 260
C                                                                       SOLN 270
      DIMENSION A(2500),B(50)                                           SOLN 280
C                                                                       SOLN 290
   10 FORMAT(1H1,34HSOLUTION OF SIMULTANEOUS EQUATIONS)                 SOLN 300
   11 FORMAT(1H0,44HDIMENSIONED AREA TOO SMALL FOR INPUT MATRIX ,I4)    SOLN 310
   12 FORMAT(1H0,20HEXECUTION TERMINATED)                               SOLN 320
   13 FORMAT(1H0,47HROW AND COLUMN DIMENSIONS NOT EQUAL FOR MATRIX ,I4) SOLN 330
   14 FORMAT(1H0,42HINCORRECT NUMBER OF DATA CARDS FOR MATRIX ,I4)      SOLN 340
   15 FORMAT(1H0,18HGO ON TO NEXT CASE)                                 SOLN 350
   16 FORMAT(1H0,38HSTRUCTURE CODE IS NOT ZERO FOR MATRIX ,I4)          SOLN 360
   17 FORMAT(1H1,17HORIGINAL B VECTOR,////)                             SOLN 370
   18 FORMAT(1H1,15HSOLUTION VALUES,////)                               SOLN 380
   19 FORMAT(1H0,18HMATRIX IS SINGULAR)                                 SOLN 390
   20 FORMAT(7F10.0)                                                    SOLN 400
   21 FORMAT(I3,10X,E16.6)                                              SOLN 410
   22 FORMAT(1H0,11HEND OF CASE)                                        SOLN 420
C                                                                       SOLN 430
C     ..................................................................SOLN 440
C                                                                       SOLN 450
      WRITE (6,10)                                                      SOLN 460
   25 CALL MATIN(ICOD,A,2500,N,M,MS,IER)                                SOLN 470
      IF(N) 30,95,30                                                    SOLN 480
   30 IF(IER-1) 45,35,40                                                SOLN 490
   35 WRITE(6,11) ICOD                                                  SOLN 500
      GO TO 90                                                          SOLN 510
   40 WRITE(6,14) ICOD                                                  SOLN 520
      GO TO 95                                                          SOLN 530
   45 IF(M-N) 50,55,50                                                  SOLN 540
   50 WRITE(6,13) ICOD                                                  SOLN 550
      GO TO 90                                                          SOLN 560
   55 IF(MS) 60,65,60                                                   SOLN 570
   60 WRITE(6,16) ICOD                                                  SOLN 580
      GO TO 90                                                          SOLN 590
   65 CALL MXOUT(ICOD,A,N,M,MS,60,120,2)                                SOLN 600
      READ(5,20)(B(I),I=1,N)                                            SOLN 610
      WRITE(6,17)                                                       SOLN 620
      DO 70 I=1,N                                                       SOLN 630
   70 WRITE(6,21) I,B(I)                                                SOLN 640
      CALL SIMQ(A,B,N,KS)                                               SOLN 650
      IF(KS-1) 80,75,80                                                 SOLN 660
   75 WRITE(6,19)                                                       SOLN 670
      WRITE(6,15)                                                       SOLN 680
      GO TO 25                                                          SOLN 690
   80 WRITE(6,18)                                                       SOLN 700
      DO 85 I=1,N                                                       SOLN 710
   85 WRITE(6,21) I,B(I)                                                SOLN 720
      WRITE(6,22)                                                       SOLN 730
      GO TO 25                                                          SOLN 740
   90 READ(5,20)(B(I),I=1,N)                                            SOLN 750
      WRITE(6,15)                                                       SOLN 760
      GO TO 25                                                          SOLN 770
   95 WRITE(6,12)                                                       SOLN 780
      RETURN                                                            SOLN 790
      END                                                               SOLN 800
```

SPECIAL SUBROUTINES USED WITH SAMPLE PROGRAMS

```
C                                                                       BOOL  10
C     ..................................................................BOOL  20
C                                                                       BOOL  30
C        USER-SUPPLIED SPECIAL SUBROUTINE - BOOL                        BOOL  40
C                                                                       BOOL  50
C        THIS SPECIAL SUBROUTINE ILLUSTRATES AN EXTERNAL SUBROUTINE     BOOL  60
C        CALLED BY SUBROUTINE SUBST.                                    BOOL  70
C                                                                       BOOL  80
C        IF DIFFERENT PROPOSITIONS ARE USED FOR DIFFERENT PROBLEMS IN   BOOL  90
C        THE SAME RUN, DIFFERENT SUBROUTINES WITH APPROPRIATE PROPOSI-  BOOL 100
C        TIONS MUST BE COMPILED UNDER DIFFERENT NAMES.  IF SO, THESE    BOOL 110
C        SUBROUTINE NAMES MUST BE DEFINED BY AN EXTERNAL STATEMENT      BOOL 120
C        APPEARING IN THE MAIN PROGRAM WHICH CALLS SUBST.  THEN, FOR    BOOL 130
C        EACH PROBLEM, SUBST IS CALLED WITH A PROPER SUBROUTINE NAME    BOOL 140
C        IN ITS ARGUMENT LIST.                                          BOOL 150
C                                                                       BOOL 160
C     ..................................................................BOOL 170
C                                                                       BOOL 180
      SUBROUTINE BOOL(R,T)                                              BOOL 190
      DIMENSION R(1)                                                    BOOL 200
C                                                                       BOOL 210
      T=R(1)*R(2)                                                       BOOL 220
C                                                                       BOOL 230
      RETURN                                                            BOOL 240
      END                                                               BOOL 250
```

```
C                                                                       DATA  10
C     ..................................................................DATA  20
C                                                                       DATA  30
C        SAMPLE INPUT SUBROUTINE - DATA                                 DATA  40
C                                                                       DATA  50
C        PURPOSE                                                        DATA  60
C           READ AN OBSERVATION (M DATA VALUES) FROM INPUT DEVICE.      DATA  70
C           THIS SUBROUTINE IS CALLED BY THE SUBROUTINE CORRE AND MUST  DATA  80
C           BE PROVIDED BY THE USER.  IF SIZE AND LOCATION OF DATA      DATA  90
C           FIELDS ARE DIFFERENT FROM PROBLEM TO PROBLEM, THIS SUB-     DATA 100
C           ROUTINE MUST BE RECOMPILED WITH A PROPER FORMAT STATEMENT.  DATA 110
C                                                                       DATA 120
C        USAGE                                                          DATA 130
C           CALL DATA (M,D)                                             DATA 140
C                                                                       DATA 150
C        DESCRIPTION OF PARAMETERS                                      DATA 160
C           M - THE NUMBER OF VARIABLES IN AN OBSERVATION.              DATA 170
C           D - OUTPUT VECTOR OF LENGTH M CONTAINING THE OBSERVATION    DATA 180
C                 DATA.,                                                DATA 190
C                                                                       DATA 200
C        REMARKS                                                        DATA 210
C           THE TYPE OF CONVERSION SPECIFIED IN THE FORMAT MUST BE      DATA 220
C           EITHER F OR E.                                              DATA 230
C                                                                       DATA 240
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  DATA 250
C           NONE                                                        DATA 260
C     ..................................................................DATA 270
C                                                                       DATA 280
      SUBROUTINE DATA (M,D)                                             DATA 290
C                                                                       DATA 300
      DIMENSION D(1)                                                    DATA 310
C                                                                       DATA 320
    1 FORMAT(12F6.0)                                                    DATA 330
C                                                                       DATA 340
C     READ AN OBSERVATION FROM INPUT DEVICE.                            DATA 350
C                                                                       DATA 360
      READ (5,1) (D(I),I=1,M)                                           DATA 370
C                                                                       DATA 380
C     INPUT DATA ARE WRITTEN ON LOGICAL TAPE 13 FOR THE RESIDUAL ANALY- DATA 390
C     SIS PERFORMED IN THE SAMPLE MULTIPLE REGRESSION PROGRAM.          DATA 400
C                                                                       DATA 410
      WRITE (13) (D(I),I=1,M)                                           DATA 420
      RETURN                                                            DATA 430
      END                                                               DATA 440
```

```
      FUNCTION FUN(X,Y)                                                 FUN   10
C                                                                       FUN   20
      FUN=1./X                                                          FUN   30
      RETURN                                                            FUN   40
      END                                                               FUN   50
```

```
C                                                                       HIST  10
C     ..................................................................HIST  20
C                                                                       HIST  30
C        SUBROUTINE HIST                                                HIST  40
C                                                                       HIST  50
C        PURPOSE                                                        HIST  60
C           PRINT A HISTOGRAM OF FREQUENCIES VERSUS INTERVALS           HIST  70
C                                                                       HIST  80
C        USAGE                                                          HIST  90
C           CALL HIST(NU,FREQ,IN)                                       HIST 100
C                                                                       HIST 110
C        DESCRIPTION OF PARAMETERS                                      HIST 120
C           NU   - HISTOGRAM NUMBER (3 DIGITS MAXIMUM)                  HIST 130
C           FREQ - VECTOR OF FREQUENCIES                                HIST 140
C           IN   - NUMBER OF INTERVALS AND LENGTH OF FREQ (MAX IS 20)   HIST 150
C                  NORMALLY, FREQ(1) CONTAINS THE FREQUENCY SMALLER THAN HIST 160
C                  THE LOWER BOUND AND FREQ(IN) CONTAINS THE FREQUENCY  HIST 170
C                  LARGER THAN THE UPPER BOUND                          HIST 180
C                                                                       HIST 190
C        REMARKS                                                        HIST 200
C           FREQUENCIES MUST BE POSITIVE NUMBERS                        HIST 210
C                                                                       HIST 220
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  HIST 230
C           NONE                                                        HIST 240
C                                                                       HIST 250
C        METHOD                                                         HIST 260
C           THE LARGEST FREQUENCY IS DETERMINED AND SCALING IS USED     HIST 270
C           IF REQUIRED                                                 HIST 280
C                                                                       HIST 290
C     ..................................................................HIST 300
C                                                                       HIST 310
      SUBROUTINE HIST(NU,FREQ,IN)                                       HIST 320
      DIMENSION JOUT(20),FREQ(20)                                       HIST 330
C                                                                       HIST 340
    1 FORMAT(6H EACH ,A1,0H EQUALS ,I2,7H POINTS,/)                     HIST 350
    2 FORMAT(I6,4X,20(4X,A1))                                           HIST 360
    3 FORMAT(9H0INTERVAL,4X,19(I2,3X),I2)                               HIST 370
    4 FORMAT(1H1,47X,11H HISTOGRAM ,I3)                                 HIST 380
    5 FORMAT(10H0FREQUENCY,20I5)                                        HIST 390
    6 FORMAT(6H CLASS)                                                  HIST 400
    7 FORMAT(113H  ------------------------------------------------------HIST 410
     1-----------------------------------------------------)            HIST 420
    8 FORMAT(1H )                                                       HIST 430
    9 FORMAT(A1)                                                        HIST 440
   10 FORMAT(1H*)                                                       HIST 450
C                                                                       HIST 460
      REWIND 13                                                         HIST 470
      WRITE(13,10)                                                      HIST 480
      REWIND 13                                                         HIST 490
      READ(13,9) K                                                      HIST 500
      REWIND 13                                                         HIST 510
      WRITE(13,8)                                                       HIST 520
      REWIND 13                                                         HIST 530
      READ(13,9) NOTH                                                   HIST 540
      REWIND 13                                                         HIST 550
C                                                                       HIST 560
C        PRINT TITLE AND FREQUENCY VECTOR                               HIST 570
C                                                                       HIST 580
      WRITE(6,4) NU                                                     HIST 590
      DO 12 I=1,IN                                                      HIST 600
   12 JOUT(I)=FREQ(I)                                                   HIST 610
      WRITE(6,5)(JOUT(I),I=1,IN)                                        HIST 620
      WRITE(6,7)                                                        HIST 630
C                                                                       HIST 640
C        FIND LARGEST FREQUENCY                                         HIST 650
C                                                                       HIST 660
      FMAX=0.0                                                          HIST 670
      DO 20 I=1,IN                                                      HIST 680
      IF(FREQ(I)-FMAX) 20,20,15                                         HIST 690
   15 FMAX=FREQ(I)                                                      HIST 700
   20 CONTINUE                                                          HIST 710
C                                                                       HIST 720
C        SCALE IF NECESSARY                                             HIST 730
C                                                                       HIST 740
      JSCAL=1                                                           HIST 750
      IF(FMAX-50.0) 40,40,30                                            HIST 760
   30 JSCAL=(FMAX+49.0)/50.0                                            HIST 770
      WRITE(6,1)K,JSCAL                                                 HIST 780
C                                                                       HIST 790
C        CLEAR OUTPUT AREA TO BLANKS                                    HIST 800
C                                                                       HIST 810
   40 DO 50 I=1,IN                                                      HIST 820
   50 JOUT(I)=NOTH                                                      HIST 830
C                                                                       HIST 840
C        LOCATE FREQUENCIES IN EACH INTERVAL                            HIST 850
C                                                                       HIST 860
      MAX=FMAX/FLOAT(JSCAL)                                             HIST 870
      DO 80 I=1,MAX                                                     HIST 880
      X=MAX-(I-1)                                                       HIST 890
      DO 70 J=1,IN                                                      HIST 900
      IF(FREQ(J)/FLOAT(JSCAL)-X) 70,60,60                               HIST 910
   60 JOUT(J)=K                                                         HIST 920
   70 CONTINUE                                                          HIST 930
      IX=X*FLOAT(JSCAL)                                                 HIST 940
C                                                                       HIST 950
C        PRINT LINE OF FREQUENCIES                                      HIST 960
C                                                                       HIST 970
   80 WRITE(5,2)IX,(JOUT(J),J=1,IN)                                     HIST 980
C                                                                       HIST 990
C        GENERATE CONSTANTS                                             HIST1000
C                                                                       HIST1010
      DO 90 I=1,IN                                                      HIST1020
   90 JOUT(I)=I                                                         HIST1030
C                                                                       HIST1040
C        PRINT INTERVAL NUMBERS                                         HIST1050
C                                                                       HIST1060
      WRITE(6,7)                                                        HIST1070
      WRITE(6,3)(JOUT(J),J=1,IN)                                        HIST1080
      WRITE(6,6)                                                        HIST1090
      RETURN                                                            HIST1100
      END                                                               HIST1110
```

```
C                                                                       PLOT  10
C     ..................................................................PLOT  20
C                                                                       PLOT  30
C        SUBROUTINE PLOT                                                PLOT  40
C                                                                       PLOT  50
C        PURPOSE                                                        PLOT  60
C           PLOT SEVERAL CROSS-VARIABLES VERSUS A BASE VARIABLE         PLOT  70
C                                                                       PLOT  80
C        USAGE                                                          PLOT  90
C           CALL PLOT (NO,A,N,M,NL,NS)                                  PLOT 100
C                                                                       PLOT 110
C        DESCRIPTION OF PARAMETERS                                      PLOT 120
C           NO - CHART NUMBER (3 DIGITS MAXIMUM)                        PLOT 130
C           A  - MATRIX OF DATA TO BE PLOTTED. FIRST COLUMN REPRESENTS  PLOT 140
C                BASE VARIABLE AND SUCCESSIVE COLUMNS ARE THE CROSS-    PLOT 150
C                VARIABLES (MAXIMUM IS 9).                              PLOT 160
C           N  - NUMBER OF ROWS IN MATRIX A                             PLOT 170
C           M  - NUMBER OF COLUMNS IN MATRIX A (EQUAL TO THE TOTAL      PLOT 180
C                NUMBER OF VARIABLES). MAXIMUM IS 10.                   PLOT 190
C           NL - NUMBER OF LINES IN THE PLOT. IF 0 IS SPECIFIED, 50     PLOT 200
C                LINES ARE USED.                                        PLOT 210
C           NS - CODE FOR SORTING THE BASE VARIABLE DATA IN ASCENDING   PLOT 220
C                ORDER                                                  PLOT 230
C                  0  SORTING IS NOT NECESSARY (ALREADY IN ASCENDING    PLOT 240
C                     ORDER).                                           PLOT 250
C                  1  SORTING IS NECESSARY.                             PLOT 260
C                                                                       PLOT 270
C        REMARKS                                                        PLOT 280
C           NONE                                                        PLOT 290
C                                                                       PLOT 300
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  PLOT 310
C           NONE                                                        PLOT 320
C                                                                       PLOT 330
C     ..................................................................PLOT 340
C                                                                       PLOT 350
      SUBROUTINE PLOT(NO,A,N,M,NL,NS)                                   PLOT 360
      DIMENSION OUT(101),YPR(11),ANG(9),A(1)                            PLOT 370
C                                                                       PLOT 380
    1 FORMAT(1H1,60X,7H CHART ,I3,//)                                   PLOT 390
    2 FORMAT(1H ,F11.4,5X,101A1)                                        PLOT 400
    3 FORMAT(1H )                                                       PLOT 410
    4 FORMAT(10H 1234567890)                                            PLOT 420
    5 FORMAT(10A1)                                                      PLOT 430
    7 FORMAT(1H ,16X,101H.         .         .         .         .      PLOT 440
     1         .         .         .         .         .)               PLOT 450
    8 FORMAT(1H0,9X,11F10.4)                                            PLOT 460
C                                                                       PLOT 470
C     ..................................................................PLOT 480
```

```
C
      NLL=NL
C
      IF(NS) 16, 16, 10
C
C        SORT BASE VARIABLE DATA IN ASCENDING ORDER
C
   10 DO 15 I=1,N
      DO 14 J=I,N
      IF(A(I)-A(J)) 14, 14, 11
   11 L=I-N
      LL=J-N
      DO 12 K=1,M
      L=L+N
      LL=LL+N
      F=A(L)
      A(L)=A(LL)
   12 A(LL)=F
   14 CONTINUE
   15 CONTINUE
C
C        TEST NLL
C
   16 IF(NLL) 20, 18, 20
   18 NLL=50
C
C        PRINT TITLE
C
   20 WRITE(6,1)NO
C
C        DEVELOP BLANK AND DIGITS FOR PRINTING
C
      REWIND 13
      WRITE (13,4)
      REWIND 13
      READ (13,5) BLANK,(ANG(I),I=1,9)
      REWIND 13
C
C        FIND SCALE FOR BASE VARIABLE
C
      XSCAL=(A(N)-A(1))/(FLOAT(NLL-1))
C
C        FIND SCALE FOR CROSS-VARIABLES
C
      M1=N+1
      YMIN=A(M1)
      YMAX=YMIN
      M2=M*N
      DO 40 J=M1,M2
      IF(A(J)-YMIN) 28,26,26
   26 IF(A(J)-YMAX) 40,40,30
   28 YMIN=A(J)
      GO TO 40
   30 YMAX=A(J)
   40 CONTINUE
      YSCAL=(YMAX-YMIN)/100.0
C
C        FIND BASE VARIABLE PRINT POSITION
C
      XB=A(1)
      L=1
      MY=M-1
      I=1
   45 F=I-1
      XPR=XB+F*XSCAL
      IF(A(L)-XPR) 50,50,70
C
C        FIND CROSS-VARIABLES
C
   50 DO 55 IX=1,101
   55 OUT(IX)=BLANK
      DO 60 J=1,MY
      LL=L+J*N
      JP=((A(LL)-YMIN)/YSCAL)+1.0
      OUT(JP)=ANG(J)
   60 CONTINUE
C
C        PRINT LINE AND CLEAR, OR SKIP
C
      WRITE(6,2)XPR,(OUT(IZ),IZ=1,101)
      L=L+1
      GO TO 80
   70 WRITE(6,3)
   80 I=I+1
      IF(I-NLL) 45, 84, 86
   84 XPR=A(N)
      GO TO 50
C
C        PRINT CROSS-VARIABLES NUMBERS
C
   86 WRITE(6,7)
      YPR(1)=YMIN
      DO 90 KN=1,9
   90 YPR(KN+1)=YPR(KN)+YSCAL*10.0
      YPR(11)=YMAX
      WRITE(6,8)(YPR(IP),IP=1,11)
      RETURN
      END
```

```
C                                                                       MATI  10
C     ..................................................................MATI  20
C                                                                       MATI  30
C        SUBROUTINE MATIN                                               MATI  40
C                                                                       MATI  50
C        PURPOSE                                                        MATI  60
C           READS CONTROL CARD AND MATRIX DATA ELEMENTS FROM LOGICAL    MATI  70
C           UNIT 5                                                      MATI  80
C                                                                       MATI  90
C        USAGE                                                          MATI 100
C           CALL MATIN(ICODE,A,ISIZE,IROW,ICOL,IS,IER)                  MATI 110
C                                                                       MATI 120
C        DESCRIPTION OF PARAMETERS                                      MATI 130
C           ICODE-UPON RETURN, ICODE WILL CONTAIN FOUR DIGIT            MATI 140
C                 IDENTIFICATION CODE FROM MATRIX PARAMETER CARD        MATI 150
C           A    -DATA AREA FOR INPUT MATRIX                            MATI 160
C           ISIZE-NUMBER OF ELEMENTS DIMENSIONED BY USER FOR AREA A     MATI 170
C           IROW -UPON RETURN, IROW WILL CONTAIN ROW DIMENSION FROM     MATI 180
C                 MATRIX PARAMETER CARD                                 MATI 190
C           ICOL -UPON RETURN, ICOL WILL CONTAIN COLUMN DIMENSION FROM  MATI 200
C                 MATRIX PARAMETER CARD                                 MATI 210
C           IS   -UPON RETURN, IS WILL CONTAIN STORAGE MODE CODE FROM   MATI 220
C                 MATRIX PARAMETER CARD WHERE                           MATI 230
C                 IS=0 GENERAL MATRIX                                   MATI 240
C                 IS=1 SYMMETRIC MATRIX                                 MATI 250
C                 IS=2 DIAGONAL MATRIX                                  MATI 260
C           IER  -UPON RETURN, IER WILL CONTAIN AN ERROR CODE WHERE     MATI 270
C                 IER=0   NO ERROR                                      MATI 280
C                 IER=1   ISIZE IS LESS THAN NUMBER OF ELEMENTS IN      MATI 290
C                         INPUT MATRIX                                  MATI 300
C                 IER=2   INCORRECT NUMBER OF DATA CARDS                MATI 310
C                                                                       MATI 320
C        REMARKS                                                        MATI 330
C           NONE                                                        MATI 340
C                                                                       MATI 350
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MATI 360
C           LOC                                                         MATI 370
C                                                                       MATI 380
C        METHOD                                                         MATI 390
C           SUBROUTINE ASSUMES THAT INPUT MATRIX CONSISTS OF PARAMETER  MATI 400
C           CARD FOLLOWED BY DATA CARDS                                 MATI 410
C           PARAMETER CARD HAS THE FOLLOWING FORMAT                     MATI 420
C             COL. 1- 2 BLANK                                           MATI 430
C             COL. 3- 6 UP TO FOUR DIGIT IDENTIFICATION CODE            MATI 440
C             COL. 7-10 NUMBER OF ROWS IN MATRIX                        MATI 450
C             COL.11-14 NUMBER OF COLUMNS IN MATRIX                     MATI 460
C             COL.15-16 STORAGE MODE OF MATRIX WHERE                    MATI 470
C                  0 - GENERAL MATRIX                                   MATI 480
C                  1 - SYMMETRIC MATRIX                                 MATI 490
C                  2 - DIAGONAL MATRIX                                  MATI 500
C           DATA CARDS ARE ASSUMED TO HAVE SEVEN FIELDS OF TEN COLUMNS  MATI 510
C           EACH.  DECIMAL POINT MAY APPEAR ANYWHERE IN A FIELD.  IF NO MATI 520
C           DECIMAL POINT IS INCLUDED, IT IS ASSUMED THAT THE DECIMAL   MATI 530
C           POINT IS AT THE END OF THE 10 COLUMN FIELD. NUMBER IN EACH  MATI 540
C           FIELD MAY BE PRECEDED BY BLANKS.  DATA ELEMENTS MUST BE     MATI 550
C           PUNCHED BY ROW.  A ROW MAY CONTINUE FROM CARD TO CARD.      MATI 560
C           HOWEVER EACH NEW ROW MUST START IN THE FIRST FIELD OF THE   MATI 570
C           NEXT CARD.  ONLY THE UPPER TRIANGULAR PORTION OF A SYMMETRICMATI 580
C           OR THE DIAGONAL ELEMENTS OF A DIAGONAL MATRIX ARE CONTAINED MATI 590
C           ON DATA CARDS.  THE FIRST ELEMENT OF EACH NEW ROW WILL BE   MATI 600
C           THE DIAGONAL ELEMENT FOR A MATRIX WITH  SYMMETRIC OR        MATI 610
C           DIAGONAL STORAGE MODE. COLUMNS 71-80 OF DATA CARDS MAY BE   MATI 620
C           USED FOR IDENTIFICATION, SEQUENCE NUMBERING, ETC..          MATI 630
C           THE LAST DATA CARD FOR ANY MATRIX MUST BE FOLLOWED BY A CARDMATI 640
C           WITH A 9 PUNCH IN COLUMN 1.                                 MATI 650
C                                                                       MATI 660
C.....................................................................MATI 670
C                                                                       MATI 680
      SUBROUTINE MATIN(ICODE,   A,ISIZE,IROW,ICOL,IS,IER)               MATI 690
      DIMENSION A(1)                                                    MATI 700
      DIMENSION CARD(8)                                                 MATI 710
    1 FORMAT(7F10.0)                                                    MATI 720
    2 FORMAT(I6,2I4,I2)                                                 MATI 730
C                                                                       MATI 740
      IDC=7                                                             MATI 750
      IER=0                                                             MATI 760
      READ( 5,2)ICODE,IROW,ICOL,IS                                      MATI 770
      CALL LOC(IROW,ICOL,ICNT,IROW,ICOL,IS)                             MATI 780
      IF(ISIZE-ICNT)6,7,7                                               MATI 790
    6 IER=1                                                             MATI 800
    7 IF (ICNT)38,38,8                                                  MATI 810
    8 ICOLT=ICOL                                                        MATI 820
      IROCR=1                                                           MATI 830
C                                                                       MATI 840
C        COMPUTE NUMBER OF CARDS FOR THIS ROW                           MATI 850
C                                                                       MATI 860
   11 IRCDS=(ICOLT-1)/IDC+1                                             MATI 870
      IF(IS-1)15,15,12                                                  MATI 880
   12 IRCDS=1                                                           MATI 890
C                                                                       MATI 900
C        SET UP LOOP FOR NUMBER OF CARDS IN ROW                         MATI 910
C                                                                       MATI 920
   15 DO 31 K=1,IRCDS                                                   MATI 930
      READ(5,1)(CARD(I),I=1,IDC)                                        MATI 940
C                                                                       MATI 950
C        SKIP THROUGH DATA CARDS IF INPUT AREA TOO SMALL                MATI 960
C                                                                       MATI 970
      IF(IER)16,16,31                                                   MATI 980
   16 L=0                                                               MATI 990
C                                                                       MATI1000
C        COMPUTE COLUMN NUMBER FOR FIRST FIELD IN CURRENT CARD          MATI1010
C                                                                       MATI1020
      JS=(K-1)*IDC+ICOL-ICOLT+1                                         MATI1030
      JE=JS+IDC-1                                                       MATI1040
      IF(IS-1)19,19,17                                                  MATI1050
   17 JE=JS                                                             MATI1060
C                                                                       MATI1070
C        SET UP LOOP FOR DATA ELEMENTS  WITHIN CARD                     MATI1080
C                                                                       MATI1090
   19 DO 30 J=JS,JE                                                     MATI1100
      IF(J-ICOL)20,20,31                                                MATI1110
   20 CALL LOC(IROCR ,J,IJ,IROW,ICOL,IS)                                MATI1120
      L=L+1                                                             MATI1130
   30 A(IJ)=CARD(L)                                                     MATI1140
   31 CONTINUE                                                          MATI1150
      IROCR=IROCR+1                                                     MATI1160
      IF(IROW-IROCR) 38,35,35                                           MATI1170
   35 IF(IS-1)37,36,36                                                  MATI1180
   36 ICOLT=ICOLT-1                                                     MATI1190
   37 GO TO 11                                                          MATI1200
   38 READ(5,1) CARD(1)                                                 MATI1210
      IF(CARD(1)-9.E9)39,40,39                                          MATI1220
   39 IER=2                                                             MATI1230
   40 RETURN                                                            MATI1240
      END                                                               MATI1250
```

```
C                                                                       MXOU  10
C     ..................................................................MXOU  20
C                                                                       MXOU  30
C        SUBROUTINE MXOUT                                               MXOU  40
C                                                                       MXOU  50
C        PURPOSE                                                        MXOU  60
C           PRODUCES AN OUTPUT LISTING OF ANY SIZED ARRAY ON            MXOU  70
C           LOGICAL UNIT 6                                              MXOU  80
C                                                                       MXOU  90
C        USAGE                                                          MXOU 100
C           CALL MXOUT(ICODE,A,N,M,MS,LINS,IPOS,ISP)                    MXOU 110
C                                                                       MXOU 120
C        DESCRIPTION OF PARAMETERS                                      MXOU 130
C           ICODE- INPUT CODE NUMBER TO BE PRINTED ON EACH OUTPUT PAGE  MXOU 140
C           -A-NAME OF OUTPUT MATRIX                                    MXOU 150
C           N-NUMBER OF ROWS IN A                                       MXOU 160
C           M-NUMBER OF COLUMNS IN A                                    MXOU 170
C           MS-STORAGE MODE OF A WHERE MS=                              MXOU 180
C                  0-GENERAL                                            MXOU 190
C                  1-SYMMETRIC                                          MXOU 200
C                  2-DIAGONAL                                           MXOU 210
C           LINS-NUMBER OF PRINT LINES ON THE PAGE (USUALLY 60)         MXOU 220
C           IPOS-NUMBER OF PRINT POSITIONS ACROSS THE PAGE (USUALLY 132)MXOU 230
C           ISP-LINE SPACING CODE, 1 FOR SINGLE SPACE, 2 FOR DOUBLE     MXOU 240
C               SPACE                                                   MXOU 250
C                                                                       MXOU 260
C        REMARKS                                                        MXOU 270
C           NONE                                                        MXOU 280
C                                                                       MXOU 290
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MXOU 300
C           LOC                                                         MXOU 310
C                                                                       MXOU 320
C        METHOD                                                         MXOU 330
C           THIS SUBROUTINE CREATES A STANDARD OUTPUT LISTING OF ANY    MXOU 340
C           SIZED ARRAY WITH ANY STORAGE MODE. EACH PAGE IS HEADED WITH MXOU 350
C           THE CODE NUMBER,DIMENSIONS AND STORAGE MODE OF THE ARRAY.   MXOU 360
C           EACH COLUMN AND ROW IS ALSO HEADED WITH ITS RESPECTIVE      MXOU 370
C           NUMBER.                                                     MXOU 380
C                                                                       MXOU 390
C     ..................................................................MXOU 400
C                                                                       MXOU 410
      SUBROUTINE MXOUT (ICODE,A,N,M,MS,LINS,IPOS,ISP)                   MXOU 420
      DIMENSION A(1),B(8)                                               MXOU 430
    1 FORMAT(1H1,5X, 7HMATRIX ,I5,6X,I3,5H ROWS,6X,I3,8H COLUMNS,       MXOU 440
     18X,13HSTORAGE MODE ,I1,8X,5HPAGE ,I2,/)                           MXOU 450
    2 FORMAT(12X,8HCOLUMN  ,7(3X,I3,10X))                               MXOU 460
    3 FORMAT(1H )                                                       MXOU 470
    4 FORMAT(1H ,7X,4HROW ,I3,7(E16.6))                                 MXOU 480
    5 FORMAT(1H0,7X,4HROW ,I3,7(E16.6))                                 MXOU 490
C                                                                       MXOU 500
      J=1                                                               MXOU 510
C                                                                       MXOU 520
C        WRITE HEADING                                                  MXOU 530
C                                                                       MXOU 540
      NEND=IPOS/16-1                                                    MXOU 550
      LEND=(LINS/ISP)-2                                                 MXOU 560
      IPAGE=1                                                           MXOU 570
   10 LSTRT=1                                                           MXOU 580
   20 WRITE(6,1)ICODE,N,M,MS,IPAGE                                      MXOU 590
      JNT=J+NEND-1                                                      MXOU 600
      IPAGE=IPAGE+1                                                     MXOU 610
   31 IF(JNT-M)33,33,32                                                 MXOU 620
   32 JNT=M                                                             MXOU 630
   33 CONTINUE                                                          MXOU 640
      WRITE(6,2)(JCUR,JCUR=J,JNT)                                       MXOU 650
      IF(ISP-1) 35,35,40                                                MXOU 660
   35 WRITE(6,3)                                                        MXOU 670
   40 LTEND=LSTRT+LEND-1                                                MXOU 680
      DO 80 L=LSTRT,LTEND                                               MXOU 690
C                                                                       MXOU 700
C        FORM OUTPUT ROW LINE                                           MXOU 710
C                                                                       MXOU 720
      DO 55 K=1,NEND                                                    MXOU 730
      KK=K                                                              MXOU 740
      JT = J+K-1                                                        MXOU 750
      CALL LOC(L,JT,IJNT,N,M,MS)                                        MXOU 760
      B(K)=0.0                                                          MXOU 770
      IF(IJNT)50,50,45                                                  MXOU 780
   45 B(K)=A(IJNT)                                                      MXOU 790
   50 CONTINUE                                                          MXOU 800
C                                                                       MXOU 810
C        CHECK IF LAST COLUMN.  IF YES GO TO 60                         MXOU 820
C                                                                       MXOU 830
      IF(JT-M) 55,60,60                                                 MXOU 840
   55 CONTINUE                                                          MXOU 850
C                                                                       MXOU 860
C        END OF LINE, NOW WRITE                                         MXOU 870
C                                                                       MXOU 880
   60 IF(ISP-1)65,65,70                                                 MXOU 890
   65 WRITE(6,4)L,(B(JW),JW=1,KK)                                       MXOU 900
      GO TO 75                                                          MXOU 910
   70 WRITE(6,5)L,(B(JW),JW=1,KK)                                       MXOU 920
C                                                                       MXOU 930
C        IF END OF ROWS,GO CHECK COLUMNS                                MXOU 940
C                                                                       MXOU 950
   75 IF(N-L)85,85,80                                                   MXOU 960
   80 CONTINUE                                                          MXOU 970
C                                                                       MXOU 980
C        END OF PAGE, NOW CHECK FOR MORE OUTPUT                         MXOU 990
C                                                                       MXOU1000
      LSTRT=LSTRT+LEND                                                  MXOU1010
      GO TO 20                                                          MXOU1020
C                                                                       MXOU1030
C        END OF COLUMNS, THEN RETURN                                    MXOU1040
C                                                                       MXOU1050
   85 IF(JT-M)90,95,95                                                  MXOU1060
   90 J=JT+1                                                            MXOU1070
      GO TO 10                                                          MXOU1080
   95 RETURN                                                            MXOU1090
      END                                                               MXOU1100
```

```
C                                                                       STOU  10
C     ..................................................................STOU  20
C                                                                       STOU  30
C        SAMPLE OUTPUT SUBROUTINE STOUT                                 STOU  40
C                                                                       STOU  50
C        PURPOSE                                                        STOU  60
C           PRINT THE RESULT OF A STEP-WISE MULTIPLE REGRESSION.  THIS  STOU  70
C           SUBROUTINE IS CALLED BY THE SUBROUTINE STPRG.               STOU  80
C                                                                       STOU  90
C        USAGE                                                          STOU 100
C           CALL STOUT (NSTEP,ANS,L,B,S,T,NSTOP)                        STOU 110
C                                                                       STOU 120
C        DESCRIPTION OF PARAMETERS                                      STOU 130
C           NSTEP - INPUT VECTOR OF LENGTH 5 CONTAINING THE FOLLOWING   STOU 140
C                   INFORMATION..                                       STOU 150
C                     NSTEP(1)  DEPENDENT VARIABLE                      STOU 160
C                     NSTEP(2)  NUMBER OF VARIABLES FORCED TO ENTER     STOU 170
C                               IN THE REGRESSION                       STOU 180
C                     NSTEP(3)  NUMBER OF VARIABLES DELETED             STOU 190
C                     NSTEP(4)  THE LAST STEP NUMBER                    STOU 200
C                     NSTEP(5)  THE LAST VARIABLE ENTERED               STOU 210
C           ANS   - INPUT VECTOR OF LENGTH 11 CONTAINING THE FOLLOWING  STOU 220
C                   INFORMATION FOR THE LAST STEP..                     STOU 230
C                     ANS(1)  SUM OF SQUARES REDUCED                    STOU 240
C                     ANS(2)  PROPORTION REDUCED                        STOU 250
C                     ANS(3)  CUMULATIVE SUM OF SQUARES REDUCED         STOU 260
C                     ANS(4)  CUMULATIVE PROPORTION REDUCED             STOU 270
C                     ANS(5)  SUM OF SQUARES OF THE DEPENDENT VARIABLE  STOU 280
C                     ANS(6)  MULTIPLE CORRELATION COEFFICIENT          STOU 290
C                     ANS(7)  F-VALUE FOR ANALYSIS VARIANCE (FOR THE    STOU 300
C                             REGRESSION)                               STOU 310
C                     ANS(8)  STANDARD ERROR OF ESTIMATE                STOU 320
C                     ANS(9)  INTERCEPT                                 STOU 330
C                     ANS(10) ADJUSTED MULTIPLE R                       STOU 340
C                     ANS(11) ADJUSTED STANDARD ERROR OF ESTIMATE       STOU 350
C           L     - INPUT VECTOR OF LENGTH K (K=M-NSTEP(3)-1) CONTAIN-  STOU 360
C                   ING VARIABLES ENTERED IN THE REGRESSION.  L(1)=FIRSTSTOU 370
C                   VARIABLE ENTERED, L(2)=SECOND VARIABLE ENTERED, ETC.STOU 380
C           B     - INPUT VECTOR OF LENGTH K (K=M-NSTEP(3)-1) CONTAIN-  STOU 390
C                   ING REGRESSION COEFFICIENTS CORRESPONDING TO THE    STOU 400
C                   VARIABLES IN VECTOR L                               STOU 410
C           S     - INPUT VECTOR OF LENGTH K (K=M-NSTEP(3)-1) CONTAIN-  STOU 420
C                   ING STANDARD ERRORS OF REGRESSION COEFFICIENTS      STOU 430
C                   CORRESPONDING TO THE VARIABLES IN VECTOR L          STOU 440
C           T     - INPUT VECTOR OF LENGTH K (K=M-NSTEP(3)-1) CONTAIN-  STOU 450
C                   ING COMPUTED T-VALUES CORRESPONDING TO THE VARIABLESSTOU 460
C                   IN VECTOR L                                         STOU 470
C           NSTOP - OUTPUT OPTION CODE TO STOP THE STEP-WISE REGRESSION STOU 480
C                     1 - IF THE STEP-WISE REGRESSION IS TO BE TERMI-   STOU 490
C                         NATED BY SOME CRITERIA OTHER THAN PROPORTION  STOU 500
C                         OF SUM OF SQUARES, SUCH AS F-TEST AND SO ON,  STOU 510
C                         THIS SUBROUTINE MAY BE MODIFIED TO PERFORM    STOU 520
C                         DESIRED TESTS.  WHEN IT BECOMES NO LONGER     STOU 530
C                         NECESSARY TO CONTINUE THE STEP-WISE REGRES-   STOU 540
C                         SION, SET NSTOP EQUAL TO 1.                   STOU 550
C                     0 - IF THE STEP-WISE REGRESSION IS TO BE CONTINUEDSTOU 560
C                                                                       STOU 570
C        REMARKS                                                        STOU 580
C           THE CONTENTS OF THE VECTORS NSTEP, ANS, L ARE REQUIRED IN   STOU 590
C           SUBSEQUENT STEPS AND MUST NOT BE DESTROYED.                 STOU 600
C                                                                       STOU 610
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  STOU 620
C           NONE                                                        STOU 630
C                                                                       STOU 640
C                                                                       STOU 650
C     ..................................................................STOU 660
C                                                                       STOU 670
      SUBROUTINE STOUT (NSTEP,ANS,L,B,S,T,NSTOP)                        STOU 680
C                                                                       STOU 690
      DIMENSION NSTEP(1),ANS(1),L(1),B(1),S(1),T(1)                     STOU 700
C                                                                       STOU 710
C     ..................................................................STOU 720
C                                                                       STOU 730
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  STOU 740
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      STOU 750
C        STATEMENT WHICH FOLLOWS.                                       STOU 760
C                                                                       STOU 770
C     DOUBLE PRECISION ANS,B,S,T                                        STOU 780
C                                                                       STOU 790
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    STOU 800
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      STOU 810
C        ROUTINE.                                                       STOU 820
C                                                                       STOU 830
C     ..................................................................STOU 840
C                                                                       STOU 850
    1 FORMAT(/5H1STEP I3)                                               STOU 860
    2 FORMAT(22H0VARIABLE ENTERED.....I2)                               STOU 870
    3 FORMAT(40H0SUM OF SQUARES REDUCED IN THIS STEP....F13.3)          STOU 880
    4 FORMAT(40H PROPORTION REDUCED IN THIS STEP........F13.3)          STOU 890
    5 FORMAT(40H0CUMULATIVE SUM OF SQUARES REDUCED......F13.3)          STOU 900
    6 FORMAT(40H CUMULATIVE PROPORTION REDUCED..........F13.3,4H  OFF13.STOU 910
     13)                                                                STOU 920
    7 FORMAT(4H0FOR I3,18H VARIABLES ENTERED)                           STOU 930
    8 FORMAT(38H   MULTIPLE CORRELATION COEFFICIENT...F9.3)             STOU 940
    9 FORMAT(38H   F-VALUE FOR ANALYSIS OF VARIANCE...F9.3)             STOU 950
   10 FORMAT(38H   STANDARD ERROR OF ESTIMATE.........F9.3)             STOU 960
   11 FORMAT(/57H   VARIABLE     REGRESSION     STD. ERROR OF     COMPUTSTOU 970
     1ED/56H    NUMBER      COEFFICIENT      REG. COEFF.       T-VALUE)  STOU 980
   12 FORMAT(5X,I3,F18.5,F16.5,F14.3)                                   STOU 990
   13 FORMAT(12H    INTERCEPTF14.5)                                     STOU1000
   14 FORMAT(31H0DEPENDENT VARIABLE.............I2)                      STOU1010
   15 FORMAT(31H NUMBER OF VARIABLES FORCED....I2)                      STOU1020
   16 FORMAT(31H NUMBER OF VARIABLES DELETED...I2)                      STOU1030
   17 FORMAT(20H   (FORCED VARIABLE))                                   STOU1040
   18 FORMAT(38H        (ADJUSTED FOR D.F.)............F9.3)            STOU1050
C                                                                       STOU1060
C        TEST WHETHER THIS IS THE FIRST STEP                            STOU1070
C                                                                       STOU1080
      IF(NSTEP(4)-1) 30, 30, 35                                         STOU1090
   30 WRITE (6,14) NSTEP(1)                                             STOU1100
      WRITE (6,15) NSTEP(2)                                             STOU1110
      WRITE (6,16) NSTEP(3)                                             STOU1120
C                                                                       STOU1130
C        PRINT THE RESULT OF A STEP                                     STOU1140
C                                                                       STOU1150
   35 WRITE (6,1) NSTEP(4)                                              STOU1160
      WRITE (6,2) NSTEP(5)                                              STOU1170
      IF(NSTEP(4)-NSTEP(2)) 37, 37, 38                                  STOU1180
   37 WRITE (6,17)                                                      STOU1190
   38 WRITE (6,3) ANS(1)                                                STOU1200
      WRITE (6,4) ANS(2)                                                STOU1210
      WRITE (6,5) ANS(3)                                                STOU1220
      WRITE (6,6) ANS(4), ANS(5)                                        STOU1230
      WRITE (6,7) NSTEP(4)                                              STOU1240
      WRITE (6,8) ANS(6)                                                STOU1250
      WRITE(6,18)ANS(10)                                                STOU1260
      WRITE (6,9) ANS(7)                                                STOU1270
      WRITE (6,10) ANS(8)                                               STOU1280
      WRITE(6,18)ANS(11)                                                STOU1290
      WRITE (6,11)                                                      STOU1300
      N=NSTEP(4)                                                        STOU1310
      DO 40 I=1,N                                                       STOU1320
   40 WRITE (6,12) L(I),B(I),S(I),T(I)                                  STOU1330
      WRITE (6,13) ANS(9)                                               STOU1340
C                                                                       STOU1350
      NSTOP=0                                                           STOU1360
      RETURN                                                            STOU1370
      END                                                               STOU1380
```