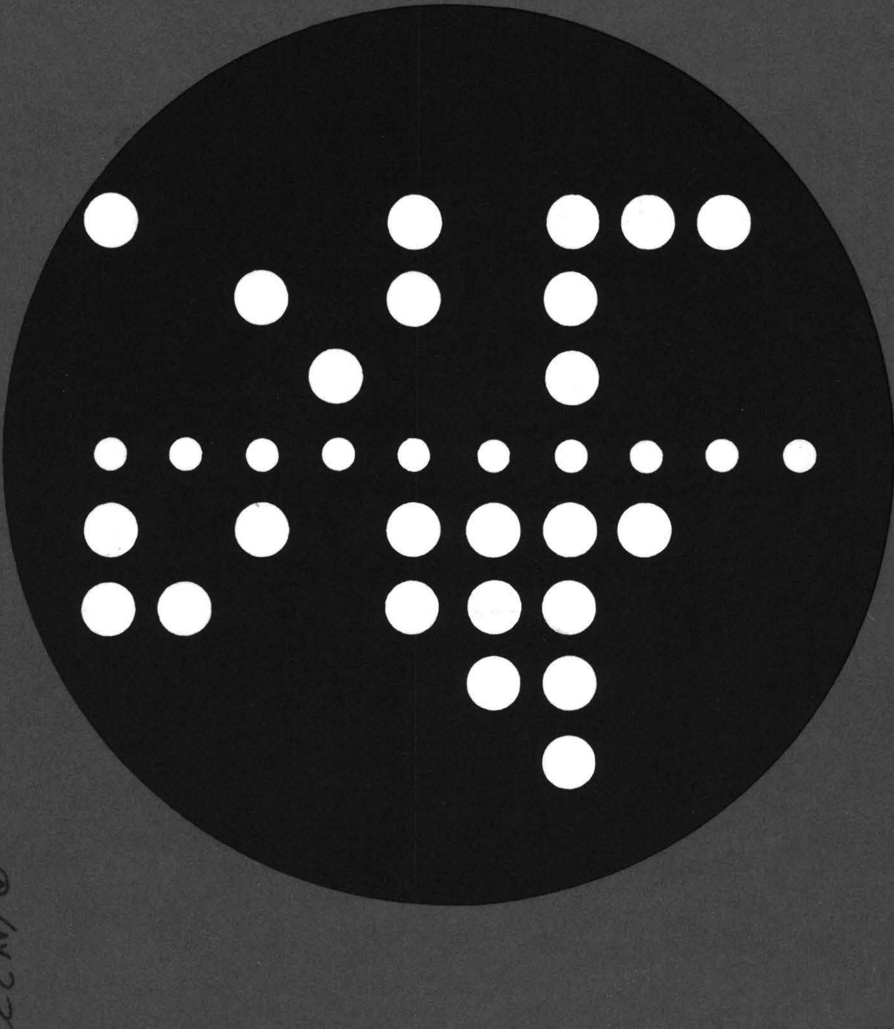
COMPUTING CENTRE NEWSLETTER

Using the IMSL & NAG Libraries



Commission of the European Communities

JOINT RESEARCH CENTRE

Ispra Establishment

SPECIAL ISSUE

Using the IMSL & NAG Libraries

A handbook describing how to use the IMSL* and NAG**. Libraries of numerical mathematical and statistical subroutines as installed at the JRC Computing Centre, Ispra.

Author: **Martyn D. Dowell**

Version 1, September 1981

* IMSL is the trademark of the International Mathematical and Statistical Library Inc. (Houston, USA)

** NAG is the trademark of the Numerical Algorithms Group Ltd.(Oxford, UK)

C O N T E N T S

1. INTRODUCTION

1.1 General Introduction

In the design and implementation of computer programs there is always a requirement for the inclusion of modules (procedures, subroutines, functions) which perform specific well defined tasks. The most obvious examples of this are modules for performing transfers from peripherals and generally handling input/output devices. The program writer would almost never consider writing his own routine to read a card from the card reader or write a record to a lineprinter. Similarly, basic trigonometric and mathematical functions such as sin(x), log(x) and e^x are always provided as standard. However, in the field of more advanced numerical mathematical and statistical calculations there has been a tradition of users writing their own subroutines to provide specific facilities. This has occurred for several reasons; the two most important are:

1) No good, comprehensive, well tested, well documented sets of routines have been available.
2) Users have always considered that they are capable of producing good routines suitable for their own needs.

In recent years the first of these reasons has become much less valid with the advent and subsequent development of two competing and yet complementary libraries of numerical mathematical and statistical subroutines (The International Mathematical and Statistical Library IMSL and the Numerical Algorithms Group NAG Library).

FORTRAN versions of both of these libraries are available for use on the JRC-Ispra Computing Centre Service. These libraries are rented from the organizations on an annual basis and are freely available for use to all of the local users of the JRC-Ispra Computing Centre Service. External and commercial users of the service should seek advice as to the conditions under which they may use these libraries from the Computer Manager (see the JRC Newsletter list of personnel for details).

Note. Users should note that single routines of IMSL and NAG may absolutely not be distributed outside the JRC, Ispra Establishment. However, complete programs or software systems which make use of the libraries may be distributed. For these cases users may request only object decks of the incorporated routines.
The person who makes the request becomes responsible for any misuse of the requested deck.

1.2 The Pitfalls in Writing Numerical Software!

The second reason why users have habitually written their own numerical mathematical subroutines (as given in the previous section) is in almost all cases false! Perhaps a few program writers produce adequate numerical mathematical subroutines for their programs. However, very many more (by far the majority) produce subroutines which are inadequate and often produce results which are unecessarily erroneous.

This may be displayed by the following example (first described in the Newsletter of the Computer Center of Purdue University (USA)).

The object of this example is to illustrate the quality which has been built into the IMSL & NAG Libraries. We do this by solving a problem, using the algorithm many people would use, and then by comparig the results we obtain with those of the correpsonding IMSL routine.

The problem we choose is to find the roots of a quadratic equation: given real numbers a,b, and c, find X such that $aX^2 + bX + c = 0$. For simplicity, we assume that a,b and c are such that the solution is also real. The two roots of a quadratic, equation may be found by the well-known 'Quadratic Formula'

$$X = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

where one root is obtained by using the "+" of the "±", and the other is obtained with the "-". The assumption that the roots are real means that $b^2-4ac \geq 0$. We can solve this problem with the following straightforward subroutine:

```
SUBROUTINE QUAD (A,B,C,X1,X2)
D = SQRT(B*B-4.0*A*C)
X1 = 0.5*(-B+D)/A
X2 = 0.5*(-B-D)/A
RETURN
END
```

When we use this subroutine to solve the rather difficult quadratic $X^2+(2^{13}+2^{-13})X+1=0$ we obtain

```
X1 = 0.0
X2 = -0.8192 x 10^4
```

This is not the correct solution, however. The corresponding IMSL routine ZQADR (single precision) does compute a much better approximation to the actual solution as follows:

$$X1 = -0.12070 \times 10^{-3}$$
$$X2 = -0.81920 \times 10^{4}$$

(These results are exact to 5 significant figures)

Where did QUAD go wrong? The second statement computes D. Then the third statement forms the difference between D and B, losing almost all significance in the process. ZQADR is more careful than QUAD and thus is able to retain full significance.

Now we touch briefly on several additional problems with QUAD. The first deals with problem scaling. If the quadratic equation above is multiplied by a constant, the solution is not changed mathematically, but it is changed computationally. For example, the quadratic

$$10^{40}X^2 + 3x10^{40}X + 2x10^{40} = 0$$

results in an overflow error because 10^{80} cannot be represented by the computer. Similarly, the quadratic

$$10^{-40}X^2 + 3x10^{-40}X + 2x10^{-40} = 0$$

produces an underflow and then gives results

$$X1 = X2 = -1.5$$

because 10^{-80} cannot be represented by the computer and is treated as zero. However, in both cases ZQADR still computes the same, correct solution.

Finally, consider QUAD's actions if the coefficient of X in the quadratic is zero. In this case QUAD returns an infinite value for one root and an indefinite value for the other. ZQADR returns the mathematically correct value -c/b for one root and infinity for the other.

The reader may ask: "What is the point of this if I never intend to solve difficult quadratic equations?". The answer is that this example shows the problems of trying to write a subroutine for solving a simple problem. It is much more difficult to write a good or even adequate subroutine to solve more complicated problems.

2. THE LIBRARIES

2.1 The IMSL Library

The International Mathematical and Statistical Library (IMSL) is produced by IMSL INC. of Houston, Texas (USA).
The library is a FORTRAN source library which contains over 400 user-callable subroutines. For IBM sites (such as the JRC-Ispra) two load module versions of the library are available, which contain sets of the subroutines with single and double precision real parameters.
The company was founded in the early 1970's and has now a very strong world-wide user base. It is especially strong in the North American continent. In total, the number of installations subscribing to the IMSL Library exceeds 1700 (located in 36 different countries). IMSL has an estimated user base of 106,000 persons.
A board of technical advisors to IMSL which consists of many famous experts in numerical mathematics, statistics and computer science, is responsible for ensuring that the library maintains a set of high quality subroutines which reflects the current state of the science.
The library is divided into a number of chapters, each of which covers an area of numerical mathematics or statistics. A brief list of the topics included follows:

- Analysis of Experimental Design Data
- Basic Statistics
  - Data Screening: Transgeneration
  - Elementary Classical Inference
  - Elementary Bayesian Inference
- Categorized Data Analysis
- Differential Equations; Quadrature; Differentiation
- Eigenanalysis
- Forecasting; Econometrics; Time Series
- Generation and Testing of Random Numbers; Goodness of Fit
- Interpolation; Approximation; Smoothing
- Linear Algebraic Equations
- Mathematical and Statistical Functions
  - Probability Distribution Functions
  - Special Functions of Mathematical Physics
- Non-Parametric Statistics
  - Analyses of Variance
  - Binomial or Multi-nomial Bases
  - Hyper (or Multi-hyper) Geometric Bases
  - Kolmogorov-Smirnov Tests
  - Other Bases
  - Randomization Bases

Observation Structure
   Canonical Analysis
   Cluster Analysis
   Discriminant Analysis
   Factor Analysis
   Principal Components Analysis
Regression Analysis
   Linear Models
Special Non-linear Models
Sampling
   Acceptance Sampling
   Preference Testing
   Survey Sampling
Utility Functions
   Error Detection
   Special I/O Routines
Vector, Matrix Arithmetic
Zeros and Extrema; Linear Programming

Subroutines in the IMSL library have alphanumeric names of up to six characters in length. The first character of the name is always that of the chapter in which the subroutine is located.

2.2 The NAG Library

The NAG Library is produced and distributed by the British company NAG (Numerical Algorithms Group) Ltd.
The NAG project began in 1970 when six British computer centres decided to jointly develop a library of mathematical routines. Later, many other universities and research organizations became involved in the development and use of the library.
In 1975 the library distribution service to commercial subscribers began. The library now consists of approximately 400 user-callable FORTRAN subroutines (there are also Algol 60 and Algol 68 versions of the library available, but not at the JRC-Ispra). There are FORTRAN implementations of the library on approximately 50 machine ranges. The total number of installations using the library world-wide exceeds 300. The library is most popular in Europe although there is an increasing usage in the USA and Canada. The library is always distributed in compiled form (i.e. as a load module library for IBM users). Source copies of individual routines are available for inspection.
For IBM users (such as the JRC-Ispra Computing Centre Service) there are two load module versions of the library using single and double precision real parameters.

The aim of NAG has always been to produce a comprehensive library of subroutines and to have as main criteria for selection of these routines the concepts of:

i) usefulness ii) robustness iii) numerical stability iv) accuracy v) speed

Contributors to the NAG Library are expert numerical mathematicians and computer scientists in the UK and throughout the world. These are backed by the NAG Central Office staff who co-ordinate and control the work of the contributions.
The NAG Library is structured in chapters using the conventions adopted by the American A.C.M. (Association for Computing Machines) modified SHARE Classification Index.

Summary of the Chapters of the NAG FORTRAN Library

A02 - COMPLEX ARITHMETIC
C02 - ZEROS OF POLYNOMIALS
C05 - ROOTS OF ONE OR MORE TRASCENDENTAL EQUATIONS
C06 - SUMMATION OF SERIES
D01 - QUADRATURE
D02 - ORDINARY DIFFERENTIAL EQUATIONS
D04 - NUMERICAL DIFFERENTIATION
D05 - INTEGRAL EQUATIONS
E01 - INTERPOLATION
E02 - CURVE AND SURFACE FITTING
E04 - MINIMIZING OR MAXIMIZING A FUNCTION
F01 - MATRIX OPERATIONS INCLUDING INVERSION
F02 - EIGENVALUES AND EIGENVECTORS
F03 - DETERMINANTS
F04 - SIMULTANEOUS LINEAR EQUATIONS
F05 - ORTHOGONALISATION
G01 - SIMPLE CALCULATIONS ON STATISTICAL DATA
G02 - CORRELATION AND REGRESSION ANALYSIS
G04 - ANALYSIS OF VARIANCE
G05 - RANDOM NUMBER GENERATORS
H - OPERATIONS RESEARCH
M01 - SORTING
P01 - ERROR TRAPPING
S - MATHEMATICAL CONSTANTS
X01 - MACHINE CONSTANTS
X03 - INNERPRODUCTS

Subroutines in the NAG Library have names which are defined in the following manner:

* The name is 6 alphanumeric characters
* The first 3 characters are the name of the chapter in which the subroutine is found (see previous page). For example C05 for a routine which is concerned with a technique in the subject area of roots of transcendental equations.
* The 4th and 5th characters are alphabetic and serve to distinguish between different subroutines in the same chapter.
* The 6th character defines the language type of the subroutine (A for Algol, F for FORTRAN etc.); see section 5.2 for more details.

3. COMPARISON OF THE LIBRARIES

3.1 Which Library Should I Use?

In some situations this question will be easy to answer. If a certain algorithm is only implemented in the NAG Library (for example) then the user must obviously make use of this subroutine.
If, however, equivalent subroutines are available in both the NAG Library and the IMSL Library, then other factors are involved in the choice.

* For cases in which the finished program or package is to be transported to another computer site or another computing system, the decision may depend on which of the two libraries will be available at the other site or system.

* For packages which are to be made generally available to a large number of users, the location of the users may be important. As previously stated, the NAG Library is widely available at centres in Europe, whilst the IMSL Library is more predominant in the USA and Canada.

* For cases in which such constraints do not apply, for example a small program to be run on the JRC-Ispra Computing Centre Service, then personal preference will be the deciding factor. The user should compare the specification and description of the equivalent routines and decide which is the more appropriate.

In the following section a brief summary and comparison of the content of the two libraries is given.

3.2 Comparative Summary of the Content of the IMSL and NAG Libraries

The following comparison gives a very basic idea of the strengths and weaknesses of the two libraries. It is very much the author's opinion on the subject and does not in any way express the views of either IMSL or NAG. The list is ordered by the chapter of the IMSL Library. It gives details of each chapter with some idea of the coverage of the NAG Library in each subject area. At the end of this list details of NAG Library subject areas not covered by the IMSL chapters are given.

<u>IMSL Chapter A - Analysis of Experimental Design Data</u>
This subject area is well covered in the IMSL Library. In the NAG Library the G04 chapter contains only a subset of the material covered by IMSL.

<u>IMSL Chapter B - Basic Statistics</u>
IMSL Chapter C - Categorized Data Analysis
Are equivalent to the material covered in the NAG G01 and G02 chapters. In general, there is a more varied coverage in the IMSL Library, although the NAG Library implements some algorithms which are not included in the IMSL Library.

<u>IMSL Chapter D - Differential Equations, Quadrature, Differentiation</u>
This IMSL chapter is equivalent to the four NAG chapters:

- D01 - Quadrature
- D02 - Ordinary Differential Equations
- D03 - Partial Differential Equations
- D04 - Numerical Differentiation

Both libraries cover this area well. However, the NAG coverage is better.

<u>IMSL Chapter E - Eigenanalysis</u>
The IMSL chapter is equivalent to the NAG Chapter F02. Both of the libraries have good subroutines available for all the commonly required form of the eigenvalue problem. The NAG chapter is of an especially high quality.

<u>IMSL Chapter F - Forecasting, Time Series Analysis, Fourier Transforms</u>
The Fourier transform section is equivalent to the NAG C06 chapter. The forecasting and time series analysis subroutines are not available in the NAG Library (but are planned for a future release in the G13 chapter).

<u>IMSL Chapter G - Generation and Testing of Pseudo-random Numbers</u>
The NAG G05 chapter contains the equivalent subroutines. Both libraries give an excellent coverage of this area.

<u>IMSL Chapter I - Interpolation, Approximation, Smoothing</u>
The NAG E01 chapter covers the subject area of interpolation.
The NAG E02 chapter covers the subject area of approximation and smoothing.

IMSL Chapter L - Linear Algebraic Equations

The NAG F04 chapter contains the equivalent subroutines.
As with the chapter on eigenanalysis, this subject area is given excellent coverage by both libraries.

IMSL Chapter M - Mathematical and Statistical Special Functions

The NAG S chapter contains the equivalent subroutines. There is a wide diversity of the special functions which are covered. The user will generally need to consult both libraries to find an implementation of the subroutine required. IMSL has more statistical spcial functions and NAG has more mathematical special functions.

IMSL Chapter N - Non-parameteric Statistics

Subroutines for this subject area are only available in the IMSL Library.

IMSL Chapter O - Observation Structure, Multivariate Statistics

The NAG G08 chapter covers this area. There is a much wider coverage in the IMSL Library.

IMSL Chapter R - Regression Analysis

The NAG G02 chapter contains some subroutines which cover part of the subject area. The NAG G02 chapter does not include subroutines for stepwise regression analysis or curvilinear regression analysis. For curvilinear regression analysis the NAG documentation suggest the choice of a model and then the use of a least squares fit subroutine for the E04 (minimization) chapter.

IMSL Chapter S - Sampling

Subroutines for this subject area are only available in the IMSL Library.

IMSL Chapter U - Utility Subroutines

This chapter has no equivalent chapter in the NAG Library. It consists of two separate subsets:

1) Subroutines for input/output in various special forms such as matrix input/output lineprinter histogram drawing.
2) HELP subroutines, to obtain information about various IMSL Library aspects

IMSL Chapter V - Vector Arithmetic & Sorting

The NAG F01 chapter contains the equivalent subroutines for vector arithmetic. The coverage of the subject area in both libraries is good. The NAG M chapter contains the equivalent sorting routines.

IMSL Chapter Z Zeros and Extrema, Linear Programming

The NAG C02, C05 and E04 chapters contains the subroutines for zeros and extrema.

The NAG H Chapter contains the linear programming subroutines.

The NAG E04 Chapter gives a much wider coverage of the general problem of finding local maxima or minima of a function.

The NAG H Chapter (Operations Research) contains more than simply linear programming subroutines.

Chapters and Facilities in NAG which are not available in IMSL.

NAG has explicit chapters for mathematical and machine constants (X01 and X02).

Full chapters or determinants (F03) and orthonormalization (F05) are present in the NAG Library.

A chapter on integral equations (D05) is provided in the NAG Library.

In appendix B a full comparative list of the various subroutines in the NAG & IMSL libraries is given.

Both organizations actively encourage users to request inclusion of any algorithms which are not present. IMSL even provides a formal "RAI" request service (Request for Ability Inclusion) with a specific form for the users to complete.

4. DOCUMENTATION

4.1 Documentation Overview

Before using a subroutine from either of the libraries it is necessary to read the relevant documentation. This, and the following sections, give information which will assist the programmer to use the available documentation effectively.

The documentation may be considered in five separate parts.

1) General introductory information.
   Information of a general introductory type may be found in this document and in the general introductions to the NAG Library Manual and the IMSL Library Manual. Both of these manuals are available for reference in the Computing Support Library (Room 1871 building A36)

2) Subject introduction and algorithm choice.
   Documentation which gives background information about a subject area in numerical mathematics or statistics, together with advice on choice of subroutines for different problems within the subject area.
   This information may be found in the introduction to the relevant chapter of the NAG Library Manual or the IMSL Library Manual. Also, in the case of the NAG Library only, there is a publication titled the NAG Mini-Manual, which is simply a collection of all the chapter introduction documents. Again this manual is available for reference in the Computing Support Library.

3) Individual Subroutines Documentation.
   Documentation about individual subroutines which explains in detail how to use the subroutines in a standard FORTRAN manner (i.e. without details of how to use it on the JRC-Ispra Computing Centre Service. This information may be found in the individual routine documents in the IMSL Library Manual and the NAG Library Manual.

4) Implementation Specific Documentation.
   Documentation which relates the documentation described in 2) and 3) to a particular implementation of the IMSL or NAG Library (e.g. documentation giving details of IBM specific features of certain aspects of the particular library).

This information is provided in the form of a short document by both IMSL and NAG. Copies of these documents have been included in the reference copies of both the IMSL Library Manual and the NAG Library Manual. Also, some of the more important information of this type is presented in the following sections of this document.

5) Use of the libraries at the JRC-Ispra
Documentation giving details of how to use the library on the JRC-Ispra Computing Centre Service. This information is available in section 6. of this document.

4.2 IMSL Documentation

The IMSL documentation is in the form of the IMSL Library Manual. This is at present in three volumes. The manual structure is as follows:

Introduction
A general introduction which also gives some information about specific features relating to different computer-compiler environments.

Contents
A brief description of such subroutines, which is ordered by chapter.

KWIC Index
A keyword in Context (KWIC) index of the subroutines to help the user located the chapter/subroutine required.

Chapters A-Z
Each chapter is split into two parts:

a) The general introduction which contains:

Chapter Name
Quick Reference Guide to Chapter Facilities
Featured Abilities
Name Conventions for this Chapter
Special Instructions on Usage (Optional)
Subtleties to Note (Optional)
Pitfalls to Avoid (Optional)

The chapter introduction is followed by individual subroutine documentation.

b) Individual subroutine documents
Subroutine documentation consists of two parts. The first is a copy of comment lines that appear at the beginning of each subroutine source deck.

The comment lines are as follows:

IMSL ROUTINE NAME - routine name

PURPOSE - a statement of the purpose of the routine

USAGE - the form of the subprogram CALL with arguments listed

ARGUMENTS - a description of the arguments in the order of their occurrence in USAGE

PRECISION/HARDWARE - environment specific information giving the precision of the routine - SINGLE, or DOUBLE

REQD. IMSL ROUTINES - a list of all IMSL routines called (directly and indirectly) by this routine

NOTATION - reference to manual introduction and IMSL routine UHELP

REMARKS (optional) - details pertaining to code usage

The second part of the document (which does not appear in the source code) includes the following sections:

ALGORITHM - a brief statement of the algorithm and references to detailed information

PROGRAMMING NOTES (optional) - programming details not covered elsewhere

ACCURACY (optional) - a statement about the accuracy of the routine

EXAMPLE - an example showing subroutine input, required dimension and type statements and output

4.3 NAG Documentation

The NAG documentation for use at the JRC-Ispra Computing Centre Service is in the form of three publications:

1) the NAG Library Manual (at present in 6 volumes)
2) the NAG Mini-Manual (the chapter introduction for the NAG Library Manual)
3) the NAG IBM FORTRAN Implementation Documents (single and double precision)

The NAG Library Manual
The manual structure is as follows:

Foreword
Written by Professor L. Fox (Oxford University) and Dr. J.H. Wilkinson (N.P.L.) This is interesting and educational reading.

Introduction
Contains a great deal of important information.

Chapters A02-X04
Each is split into two parts:

a) the chapter introduction which contains:
   1. The scope of the chapter
   2. Background to the problems
   3. Recommendations on choice and use of routines

b) the individual routine documents which are structured as follows:

   All routine documents have 13 numbered sections with the following headings:
   1. Purpose
   2. Specification
   3. Description
   4. References
   5. Parameters
   6. Error Indicators
   7. Auxiliary Routines
   8. Timing
   9. Storage
   10. Accuracy
   11. Further Comments
   12. Keywords
   13. Example

The NAG Mini-Manual
This is formed from the foreword, introduction and all of the chapter introductions. This is a useful manual for an introduction to the NAG Library and for helping the user to find the routine which he requires. Actual routine specifications are, however, only found in the NAG Library Manual.

The NAG IBM FORTRAN Implementation Documents
These documents contain details of how the IBM FORTRAN implementation of the NAG Library should be used (in a general sense) and how the implementation differs from the standard implementation (as defined by the NAG Library Manual). It is important that all prospective library users read the appropriate single or double precision documents. Both documents are inserted at the beginning of each NAG Library Manual.

5. IMPLEMENTATION SPECIFIC DETAILS

5.1 IMSL Implementation Details

The subroutine naming convention in the IMSL libraries is the same for both single and double precision versions. This implies that in one program it is not normally possible to include one routine from the single precision library and another routine from the double precision library.

Although most subroutines are available in both single and double precision versions, there are some exceptions. In each routine specification document there is a section "PRECISION/HARDWARE". The user must check this to make sure that the routine is implemented in the required environment.

SINGLE/H32	means that the subroutine is available in the IBM FORTRAN single-precision library.
DOUBLE/H32	means that the subroutine is available in the IBM FORTRAN double-precision library.

Examples given in the routine specification will normally require some modifications before being suitable for use with either the single or double precision libraries. Normally, the examples will have been written for the single precision version, and therefore the normal changes (of REAL to DOUBLE PRECISION etc.) will be necessary to run them with the double precision version of the subroutines.

5.2 NAG Implementation Details

The naming convention for NAG subroutines is given in Section 2.2. The sixth character of the NAG subroutine name is a letter which defines the language type of the subroutine. This letter is also used to create a difference between the single and double precision IBM FORTRAN subroutines.

For the single precision library the sixth character of the routine name is always E.
For the double precision library the sixth character of the routine name is always F.

Therefore, all subroutines in the single precision library and double precision library have different names. Thus, mixing of the use of different single and double precision NAG subroutines in one program is possible.

For double precision:

CALL E04CGF(N,X,F,IW,LIW,W,LW,IFAIL)

For single precision:

CALL E04CGE(N,X,F,IW,LIW,W,LW,IFAIL)

In the NAG Library Manual there are certain terms which are italicized. The implication is that these terms are implementation dependent and should be replaced by the appropriate actual term for the implementation being used (ie for the single precision IBM FORTRAN implementation or the double precision IBM FORTRAN implementation).

Italicized term	IBM FORTRAN single precision	IBM FORTRAN double precision
real	(REAL*4)	DOUBLE PRECISION (REAL*8)
complex	COMPLEX*8	COMPLEX*16
basic precision	single precision	double precision
additional precision	double precision (REAL*8)	quadruple precision (REAL*16)

Example programs published in the Library Manual are in single precision. Therefore, for use with the single precision library they should seldom require any modification.
For use of example programs with the double precision library may require modifications in the following area:

1) Inserting as appropriate REAL*8 and COMPLEX*16 statements
2) Changing any intrinsic functions to double precision version e.g. SQRT to DSQRT
3) Specifying real constants in double precision (D) format.
4) Explicitly converting any implicit integer to real conversions using the DFLOAT function
5) Changing any E formats to D formats

6. USING THE LIBRARIES

The IMSL Library and NAG Library are available for use on the JRC-Ispra Computing Centre Service. The following sections describe their use in FORTRAN programs, both in batch and from a TSO foreground session. In section 7. a brief description is given of how to include NAG and IMSL subroutines in programs written in other languages.

6.1 Library data set definitions

There are four IMSL and NAG load module libraries available for use. These are the single and double precision versions of both libraries.
The names of the data sets are given in the following table:

Library	Data set name
IMSL single precision	SYS1.LIBMASXS
IMSL double precision	SYS1.LIBMASXD
NAG single precision	SYS1.LIBNAGS
NAG double precision	SYS1.LIBNAGD

ALL OF THE DATA SETS IN THE ABOVE TABLE ARE CATALOGED

6.2 Use of the Libraries in Batch

Users may easily access one of the libraries by using one of the standard FORTRAN G1 compiler procedures in the following manner:

```
//    EXEC FTG1CG,PRN=abcd
```

where abcd is replaced as follows:

NAGS - NAG single precision library
NAGD - NAG double precision library
MASXS - IMSL single precision library
MASXD - IMSL double precision library

So, for example:

```
//    EXEC FG1CG,PRN=NAGS
```

implies a FORTRAN compilation, load and go with subroutines in the program from the NAG single-precision library.

Note. The use of the FG1CG procedure is only an example. The use of these libraries in this manner is possible for all of the following FORTRAN G1 & HE procedures.

FTG1CL	**FTHECL**
FTG1CLG	**FTHECLG**
FTG1CG	**FTHECG**
FTG1L	**FTHEL**
FTG1LG	**FTHELG**
FTG1G	**FTHEG**

In appendix A examples (with explanation) are given of the use of the above mentioned system.

Use of More then One Library in the Same Program

As has been stated in section 5.1, it is not normally possible to use different subroutines from both the IMSL single and double precision libraries in the same program. This is due to the fact that the naming conventions are the same for the two libraries. However, it is possible to mix subroutines from the two NAG library and also the mix subroutines from the NAG libraries with one of the IMSL libraries. In this case it is not possible to use the job control command specifed in the previous section because it is necessary to include subroutines from more than one of the libraries.
In this case the libraries are included into the standard SYSLIB by concatenation.

e.g.

```
//      EXEC  FTG1CLG
//CMP.SYSIN  DD *
.  )
.  |
.  }       FORTRAN source program
.  |
.  )
/*
//LKED.SYSLIB DD
//          DD
//          DD
//          DD
//          DD DSN=SYS1.LIBNAGS,DISP=SHR
//          DD DSN=SYS1.LIBNAGD,DISP=SHR
```

In general, any 2 or 3 of the 4 libraries (except combinations including the two IMSL libraries) may be included by concatenation in this manner (see Appendix A for an example).

6.3 Use of Library from a TSO Session

There are two different ways in which subroutines from the mathematical libraries may be used in TSO test FORTRAN compilations.

a) Using the FG1CLG command procedure to perform the compilation, link edit and execution. With this procedure it is possible to obtain the necessary subroutines from one of the libraries by including a specific parameter.

b) Using the FORT TSO command followed by either LOADGO or LINK it is possible to include subroutines from one or more of the libraries.

Note. It is also possible to make use of the mathematical libraries by using the CONCAT TSO command. For further details see the IBM Manual TSO Command Reference Manual (GC28-6732).

a) Using FG1CLG

The TSO command procedure FG1CLG has a parameter PRN (---). This parameter is equivalent to the use of the PRN= parameter of FG1CLG in a batch mode. This allows the inclusion of subroutines from one of the mathematical libraries.
The PRN(---) parameter has as operand the xxxx part of the library name SYS1.LIBxxxx (e.g. for SYS1.LIBNAGS the user should include the parameter PRN(NAGS)).
An example of the use of this technique is given in the following example (example 1).

b) Use of FORT followed by either LOADGO or LINK

Using this method the compilation of the FORTRAN program is split into two separate phases:

1) The compilation to produce an object module.

2) The use of either the link-editor or the loader to take the object module together with any appropriate subroutine from the various subroutine libraries and produce an executable program.

Note. If the link editor procedure is used then a load module version of the program is created. This must be executed using the TSO CALL command.

In the second stage of this operation it is possible to include subroutines from one or more of the mathematical libraries by using the LIB(---) parameter.
Examples of the use of these techniques are given in examples 2, 3 and 4.

Note for NAG Users

Users are reminded that in the case of NAG library routines the routine names are different for single and double precision libraries.

> E04CAF on the double precision library becomes:
> E04CAE on the single precision library (i.e. the final character is changed from F to E)

However, for the IMSL library the routine names are the same for the single and double precision versions. Therefore, in general, it will not be possible to use subroutines from the single & double precision IMSL libraries in the same program.

Examples of TSO Usage of NAG & IMSL Subroutines

In the following examples lines typed by the user are shown in lower case.
The carriage return/ENTER character at the end of each input line is marked by a (CR).

The following type of examples are given:

Example 1:
This example shows the use of the FG1CLG TSO command procedure for a program which uses the IMSL single precision library.

Example 2:
This example shows the use of the NAG double precision library using FORT followed by LOADGO.

Example 3:
This example shows the compilation link edit and execution of a program which uses the NAG single precision library.

Example 4:
This example shows the compilation, load and execution of a program which uses both the NAG and IMSL single precision libraries.

Example 1

This example shows the use of the FG1CLG TSO command procedure for a program which uses the IMSL single precision library.

```
    qed newcomp1 fortgi new (CR)
    INPUT
    00010c  example of imsl single precision library (CR)
    00020c  analysis of two-way classification design data (CR)
    00030      integer i,ndf(5),ier (CR)
    00040      real y(6),em(11),gm,s(5) (CR)
    00050      data y/73.,90.,98.,107.,94.,49./ (CR)
    00060      call arcban(y,1,3,2,em,gm,s,ndf,ier) (CR)
    00070      write(6,99999) (em(i),i=1,3) (CR)
    00080      write(6,99998) (em(i),i=4,5) (CR)
A   00090      stop (CR)
    0010099999 format(18h block means are : ,11f7.2) (CR)
    0011099998 format(22h treatment means are : ,11f7.2) (CR)
    00120      end (CR)
    00130 (CR)
    QED
    scan (CR)
    QED
    end save (CR)
    SAVED
    READY
    fg1clg newcomp1 prn(masxs) (CR)

    DATA SET NEWCOMP1.LIST NOT IN CATALOG
    DATA SET NEWCOMP1.OBJ NOT IN CATALOG
    DATA SET NEWCOMP1.DECK NOT IN CATALOG
    DATA SET NEWCOMP1.SYSUT1 NOT IN CATALOG
    DATA SET NEWCOMP1.DUMP NOT IN CATALOG
    DATA SET NEWCOMP1.LOAD NOT IN CATALOG
B   UTILITY DATA SET NOT FREED, IS NOT ALLOCATED
    UTILITY DATA SET NOT FREED, IS NOT ALLOCATED
    ENTER CONTROL STATEMENTS-
    (CR)
    END OF CONTROL STATEMENTS

    BLOCK MEANS ARE :  81.50 102.50  71.50
    TREATMENT MEANS ARE :  88.33  82.00
    READY
```

In stage A a FORTRAN program is typed by the user (using the QED editor). Note the use of the SCAN subcommand of QED to check the validity of the FORTRAN program.

In stage B the FG1CLG TSO command procedure is used to compile link edit and execute the program. Note, that link editor control statements may be input. Typing a (CR) without any other information ends these control statements.

Example 2

This example shows the use of the NAG double precision library using FORT followed by LOADGO.

```
     list newcomp2.fort (CR)
      NEWCOMP2.FORT
      C     EXAMPLE OF NAG DOUBLE PRECISION LIBRARY
      C     USES F03AAF - MATRIX DETERMINANT CALCULATION
            DOUBLE PRECISION DETERM,A(4,4),WKSPCE(18)
            INTEGER I,N,J,IA,IFAIL
            READ(5,99999) (WKSPCE(I),I=1,7)
            N=3
            READ(5,99998) ((A(I,J),J=1,N),I=1,N)
            IA=4
            IFAIL=1
            WRITE(6,99997) (WKSPCE(I),I=1,6)
            CALL F03AAF(A,IA,N,DETERM,WKSPCE,IFAIL)
A           IF(IFAIL.EQ.0) GOTO 20
            WRITE(6,99996)IFAIL
            STOP
      20    WRITE(6,99995)DETERM
            STOP
      99999 FORMAT(6A4,A3)
      99998 FORMAT(3F5.0)
      99997 FORMAT(4(1X/),1H ,5A4,A3,7HRESULTS/1X)
      99996 FORMAT(25HOERROR IN F03AAF IFAIL = ,I2)
      99995 FORMAT(24HOVALUE OF DETERMINANT = ,F4.1)
            END
     READY
     fort newcomp2 (CR)
     G1 COMPILER ENTERED
B    SOURCE ANALYZED
     PROGRAM NAME = MAIN
     * NO DIAGNOSTICS GENERATED
     READY
     loadgo newcomp2.obj lib('sys1.libnagd') fortlib (CR)
     f03aaf example program data (CR)
        33   16   72 (CR)
C1     -24  -10  -57 (CR)
        -8   -4  -17 (CR)

C

     F03AAF EXAMPLE PROGRAM RESULTS
C2
     VALUE OF DETERMINANT =  6.0
```

In stage A a previously created data set is listed.
In stage B the program stored in the data set is compiled.
In stage C the load and execution of the program is performed.
In C1 the data is input.
In C2 the output is produced.

Example 3

This example shows the compilation link edit and execution of a program which uses the NAG single precision library.

```
   list newcomp3.fort (CR)
    NEWCOMP3.FORT
          INTEGER MAXDIV,IFAIL,NOFUN
          REAL A,B,EPS,ACC,ANS,ERROR,FUN
          EXTERNAL FUN
          A=0.0
          B=1.0
          MAXDIV=20
          EPS=1.0E-8
          ACC=0.0
          IFAIL=1
          CALL D01AGE(A,B,FUN,MAXDIV,EPS,ACC,ANS,ERROR,NOFUN,
         * IFAIL)
A         WRITE(6,99998)ANS,ERROR,NOFUN
          IF(IFAIL)20,40,20
    20    WRITE(6,99997)
    40    STOP
    99998 FORMAT(/12H INTEGRAL = ,F11.4,3X,9H ERROR = ,E11.4,3X,
         * 20H NUMBER OF POINTS = ,I3)
    99997 FORMAT(43H METHOD WAS UNABLE TO EVALUATE THE INTEGRAL)
          END
          REAL FUNCTION FUN(X)
          REAL X
          FUN=4.0/(1.0+X*X)
          RETURN
          END
   READY
   fort newcomp3 (CR)
   G1 COMPILER ENTERED
   SOURCE ANALYZED
   PROGRAM NAME = MAIN
B  *  NO DIAGNOSTICS GENERATED
   SOURCE ANALYZED
   PROGRAM NAME = FUN
   *  NO DIAGNOSTICS GENERATED
      *STATISTICS*  NO DIAGNOSTICS THIS STEP
   READY
   link newcomp3.obj lib('sys1.libnags') fortlib (CR)
C
   READY
   call newcomp3 (CR)
   TEMPNAME ASSUMED AS A MEMBER NAME
D
   INTEGRAL =    3.1416   ERROR =  0.3052E-04   NUMBER OF POINTS = 9
```

In stage A a previously created data set is listed.
In stage B the program stored in the data set is compiled.
In stage C the link edit of the program is performed.
The output load module of the program is stored in NEWCOMP3.LOAD(TEMPNAME).
In stage D the library program is executed.

Example 4

This example shows the compilation, load and execution of a program which uses both the NAG and IMSL single precision libraries.

```
   list newcomp4.fort (CR)
    NEWCOMP4.FORT
   C      EXAMPLE OF THE USE OF TWO LIBRARIES
   C      NAG & IMSL (BOTH SINGLE PRECISION)
   C      THE PROGRAM FINDS THE ROOT OF A FUNCTION
          INTEGER MAXFN,IER
          REAL A,B,FUN
          EXTERNAL FUN
          A=-10.
          B=10.
          MAXFN=100
          CALL ZBRENT(FUN,0.0,8,A,B,MAXFN,IER)
          IF(IER.EQ.0)GOTO 20
A         WRITE(6,99998)IER
          STOP
     20   WRITE(6,99999)B
          STOP
   99999 FORMAT(20HOESTIMATE OF ROOT = ,E10.3)
   99998 FORMAT(25HOERROR IN ZBRENT IER    = ,I2)
          END
          REAL FUNCTION FUN(X)
          REAL X
          IFAIL=0
          FUN=S15ACE(X,IFAIL)-0.75
          RETURN
          END
   READY
   fort newcomp4 (CR)
   1 COMPILER ENTERED
   SOURCE ANALYZED
   PROGRAM NAME = MAIN
B    NO DIAGNOSTICS GENERATED
   SOURCE ANALYZED
   PROGRAM NAME = FUN
     NO DIAGNOSTICS GENERATED
      *STATISTICS*  NO DIAGNOSTICS THIS STEP
   READY
   loadgo newcomp4.obj lib('sys1.libnags' 'sys1.libmasxs') fortlib (CR)
C  ESTIMATE OF ROOT = -0.674E+00
   READY
```

In stage A a previously created data set is listed.
In stage B the program stored in the data set is compiled.
In stage C the load and execution of the program is performed.
(Note, in particular, the use of the LIB parameter with two libraries.)

7. INCLUSION OF LIBRARIES IN OTHER LANGUAGE PROGRAMS

> Information regarding the use of the IMSL and NAG libraries in programs written in other programming languages (i.e. non-FORTRAN programs) will be included in a later version of this document.

APPENDIX A

Examples of the Use of the Libraries in Batch Jobs

1. Example of Use of the IMSL Library

The example shows the use of the IMSL single precision library using the IMSL subroutine ZX3LP which is an "easy-to-use" linear programming subroutine which uses the revised Simplex algorithm (see Hadley, G. "Linear Programming", Addison-Wesley, Reading, Massachusetts, 1962).

The problem is to maximise $x_1 + 3x_2 = S$

Subject to the constaints:

$$
\begin{array}{lcl}
x_1 & & \leqslant 1 \\
 & x_2 & \leqslant 1 \\
x_1 + & x_2 & \leqslant 1.5 \\
x_1 + & x_2 & \geqslant 0.5 \\
x_1 \geqslant 0 & x_2 \geqslant 0 &
\end{array}
$$

Results of Example

```
ZX3LP EXAMPLE PROGRAM RESULTS
VALUE OF OBJECTIVE FUNCTION=     3.500
SOLUTION VECTOR=       0.500     1.000
```

See next page for listing of example

Listing of Example 1 Job

```
//          JOB (YOUR JOB CARD)
$       CLASS 2
//      EXEC FTG1CG,PRN=MASXS
//CMP.SYSIN DD *
C       ZX3LP EXAMPLE PROGRAM
C
        INTEGER IA,N,M1,M2,IW(16),IER
        REAL    A(6,2),B(6),C(2),RW(52),PSOL(4),DSOL(6),S
C       NUMBER OF UNKNOWNS
        N=2
C       M1=NUMBER OF INEQUALITY CONSTRAINTS
        M1=4
C       M2=NUMBER OF EQUALITY CONSTRAINTS
        M2=0
C       IA=FIRST DIMENSION OF A
        IA=6
C       SET UP MATRIX OF CONSTRAINTS
        A(1,1)=1.0
        A(1,2)=0.0
        A(2,1)=0.0
        A(2,2)=1.0
        A(3,1)=1.0
        A(3,2)=1.0
        A(4,1)=-1.0
        A(4,2)=-1.0
C       VECTOR OF RIGHT-HAND SIDES OF CONSTRAINT EQUATIONS
        B(1)=1.0
        B(2)=1.0
        B(3)=1.5
        B(4)=-0.5
C       COEFFICIENTS OF OBJECTIVE FUNCTIONS
        C(1)=1.0
        C(2)=3.0
        CALL ZX3LP (A,IA,B,C,N,M1,M2,S,PSOL,DSOL,RW,IW,IER)
C       CHECK IF ERROR (IER NOT EQUAL TO ZERO)
        IF(IER.NE.0)WRITE(6,100)IER
        IF(IER.NE.0)GOTO 20
C       WRITE RESULTS
        WRITE(6,1001)S,PSOL(1),PSOL(2)
20      STOP
100     FORMAT(I4/F8.2/2F8.2/4F8.2)
1000    FORMAT(' ERROR IN ZX3LP    IER= ',I5)
1001    FORMAT(' ZX3LP EXAMPLE PROGRAM RESULTS'/
     1        ' VALUE OF OBJECTIVE FUNCTION=',F8.3/
     2        ' SOLUTION VECTOR=',2F10.3)
        END
/*
```

2. Example of the Use of the NAG Library

The example shows the use of the NAG double precision library. The example shows the use of the NAG subroutine E04CGF which implements an easy-to-use "quasi-Newton" algorithm (see Gill P.E. & Murray W., "Quasi-Newton methods for unconstrained optimization", Journal of the Institute of Mathematics and its Applications, 1972, Vol. 9,91-108) for finding an unconstrained minimum of a function $F(X_1,X_2,\ldots\ldots X_n)$ of the N independent variables $X_1,X_2,\ldots\ldots X_n$ using function values only.

In the example the function which is minimized is

$$F(X_1,X_2) = e^{X_1} \cdot (4X_1^2 + 2X_2^2 + 4X_1X_2 + 2X_1 + 1)$$

starting from an initial guess of $X_1 = -1$ and $X_2 = 1$.

Results of Example

```
E04CGF EXAMPLE PROGRAM RESULTS

FUNCTION VALUE ON EXIT IS            0.0000
AT THE POINT          0.5000       -1.0000
```

See next page for listing of example

Listing of Example 2 Job

```
//        JOB(YOUR JOB CARD)
$      CLASS 2
//     EXEC FTG1CG,PRN=NAGD
//CMP.SYSIN DD *
C      E04CGF EXAMPLE PROGRAM TEXT
C      ..LOCAL SCALARS..
       DOUBLE PRECISION F
       INTEGER I, IFAIL, LIW, LW, N, NOUT
C      ..LOCAL ARRAYS..
       DOUBLE PRECISION W(29), X(2)
       INTEGER IW(4)
C      ..SUBROUTINE REFERENCES..
C      E04CGF
C      ..
       DATA NOUT /6/
       WRITE(NOUT,99999)
       N=2
       X(1)=-1.0D+0
       X(2)= 1.0D+0
       LIW=4
       LW=29
       IFAIL=1
       CALL E04CGF(N,X,F,IW,LIW,W,LW,IFAIL)
C      SINCE IFAIL WAS SET TO 1 BEFORE ENTERING E04CGF, IT IS
C      ESSENTIAL TO TEST WHETHER IFAIL IS NON-ZERO ON EXIT
       IF(IFAIL.NE.0) WRITE(NOUT,99999) IFAIL
       IF(IFAIL.EQ.1) GO TO 20
       WRITE(NOUT,99997) F
       WRITE(NOUT,99996) (X(I),I=1,N)
20     STOP
C      END OF E04CGF EXAMPLE MAIN PROGRAM
99999 FORMAT (////31H E04CGF EXAMPLE PROGRAM RESULTS/)
99998 FORMAT (16H ERROR EXIT TYPE,I3, 23H - SEE ROUTINE DOCUMENT)
99997 FORMAT (27H FUNCTION VALUE ON EXIT IS ,F12.4)
99996 FORMAT (13H AT THE POINT, 2F12.4)
       END
C
       SUBROUTINE FUNCT1(N,XC,FC)
C      FUNCTION EVALUATION ROUTINE FOR E04CGF EXAMPLE PROGRAM -
C      THIS ROUTINE MUST BE CALLED FUNCT1
C      ..SCALAR ARGUMENTS..
       DOUBLE PRECISION FC
       INTEGER N
C      ..ARRAY ARGUMENTS..
       DOUBLE PRECISION XC(N)
C      ..
C      ..LOCAL SCALARS..
       DOUBLE PRECISION X1, X2
C      ..FUNCTION REFERENCES..
       DOUBLE PRECISION DEXP
C      ..
       X1=XC(1)
       X2=XC(2)
       FC=DEXP(X1)*(4.0D+0*X1*(X1+X2)+2.0D+0*X2*(X2+1.0D+0)+1.0D+0)
       RETURN
C      END OF FUNCTION EVALUATION ROUTINE
       END
/*
```

3. An Example of More than one Library in a Batch Job

The test program which is shown as example 4 in section 6.3 (for a TSO session) may be executed using the following batch job.

```
//          JOB(YOUR JOB CARD)
$      CLASS 2
//     EXEC  FTG1CLG
//CMP.SYSIN DD *
C      EXAMPLE OF THE USE OF TWO LIBRARIES
C      NAG & IMSL (BOTH SINGLE PRECISION)
C      THE PROGRAM FINDS THE ROOT OF A FUNCTION
       INTEGER MAXFN,IER
       REAL A,B,FUN
       EXTERNAL FUN
       A=-10.
       B=10.
       MAXFN=100
       CALL ZBRENT(FUN,0.0,8,A,B,MAXFN,IER)
       IF(IER.EQ.0)GOTO 20
       WRITE(6,99998)IER
       STOP
20     WRITE(6,99999)B
       STOP
99999 FORMAT(20HOESTIMATE OF ROOT = ,E10.3)
99998 FORMAT(25HOERROR IN ZBRENT IER   = ,I2)
       END
       REAL FUNCTION FUN(X)
       REAL X
       IFAIL=0
       FUN=S15ACE(X,IFAIL)-0.75
       RETURN
       END
/*
//LKED.SYSLIB DD
//          DD
//          DD
//          DD
//          DD DSN=SYS1.LIBNAGS,DISP=SHR
//          DD DSN=SYS1.LIBMASXS,DISP=SHR
END OF DATA
```

APPENDIX B

Detailed Comparison of Content of the Libraries

(based on a table produced by Dr. P. Kemp, University of Newcastle (U.K.))

	NAG	IMSL
A02 COMPLEX ARITHMETIC		
square root	A02AAF	-
modulus	A02ABF	-
quotient	A02ACF	-
C02 ZEROS OF POLYNOMIALS		
complex coefficients	C02ADF	ZCPOLY
real coefficients	C02AEF	ZRPOLY
		ZPOLR
quadratic, real coeffs	-	ZQADR
, complex coeffs	-	ZQADC
C05 ROOTS OF ONE OR MORE TRANSCENDENTAL EQUATIONS		
real function of one variable	C05AAF	ZBRENT
	C05ABF	ZFALSE
	C05ACF	ZREAL1
	C05AZF	ZREAL2
, Bus & Dekker alg.	C05ADF	-
, bin. search B &D	C05AGF	
, continuation secant	C05AJF	-
, bin. search, reverse comm.	C05AVF	-
, as C05AJF, reverse comm.	C05AXF	-
complex analytic function	-	ZANLYT
n equations, n variables, functions	C05NAF	ZSYSTM
		ZSCNT
C06 SUMMATION OF SERIES, FOURIER TRANSFORMS		
FFT, 2**m real data values	C06AAF	-
FFT, real data values	C06EAF	FFTRC
(uses extra workspace)	C06FAF	-
FFT, Hermitian sequence	C06EBF	-
(uses extra workspace)	C06FBF	-
FFT, 2**m complex data values	C06ABF	FFT2C
FFT, complex data values	C06ADF	FFTCC
	C06ECF	
(uses extra workspace)	C06FCF	-
FFT estimates. power, cross spectra	-	FTFPS
real circular convolution, period 2m	C06ACF	-
sin, cos transforms. real series	-	FFTSC

sum of Chebyshev series	C06DBF	-
conjugate of Hermitian sequence	C06GBF	-
conjugate of complex sequence	C06GCF	-
inverse Laplace transform	-	FLINV
FFT of array (1,2 or 3 dim)	-	FFTT3D

D01 QUADRATURE

finite interval	D01ACF	DCADRE
	D01AGF	DCSQDU
(Patterson algorithm)	D01AHF	-
(de Doncker algorithm)	D01AJF	-
(for oscillating fns.)	D01AKF	-
(user-specified singularities)	D01ALF	-
(log-type end point sing.)	D01APF	-
(Cauchy princ. value)	D01AQF	-
(non-adaptive)	D01BDF	-
Gauss. integral	D01BAF	-
weights and abscissae	D01BBF	-
	D01BCF	
infinite interval	D01ANF	-
double integral	D01DAF	DBCQDU
		DBLINT
multiple integral, Monte Carlo	D01FAF	-
, Gauss	D01FBF	-
, adaptive	D01FCF	-
trigonometric integral	D01ANF	-
tabular function	D01GAF	-
spline	E02BDF	-

D02 ORDINARY DIFFERENTIAL EQUATIONS

IV problem, range	D02BAF	DREBS
	D02CAF	DVERK
(stiff)	D02EAF	DGEAR
, range with output	D02BBF	-
	D02CBF	
(stiff)	D02EBF	
, range,err. est. stiff chk	D02BDF	-
, until soln comp. zero	D02BGF	-
	D02CGF	
(stiff)	D02EGF	
, until fn. of soln. zero	D02BHF	-
	D02CHF	
(stiff)	D02EHF	-
, comprehensive control	D02PAF	-
	D02QAF	
(stiff)	D02QBF	
interpolation for D02PAF, all comps.	D02XAF	-
, one comp.	D02XBF	-
D02QA/BF, all comp.	D02XGF	-
, one comp.	D02XHF	-

one step Runge Kutta	D02YAF	-
BV problem, 2 point	D02ADF	-
, 2-point, linear	D02GBF	-
	D02JAF	
	D02JBF	
, 2-point, non-linear	D02GAF	DTPTB
	D02HAF	
	D02HBF	
	D02RAF	
	D02SAF	
, generalized	D02AGF	-
, linear	D02TGF	-
collocation method	D02AFF	-
eigenvalues St-L.,reg.,finite range	D02KAF	-
,general	D02KDF	-
eigenfns. St-L,general	D02KEF	-

D03 PARTIAL DIFFERENTIAL EQUATIONS

elliptic, Laplace 2-d	D03EAF	-
, soln f.d. eqs. 5pt 2-d mol.	D03EBF	-
, Stone's strongly imp. 5pt	D03UAF	-
, soln f.d. eqs. 7pt 3-d mol.	D03ECF	-
, Stone's strongly imp. 7pt	D03UBF	-
triangulation	D03MAF	-
parabolic, 2 point BV, non linear	D03PBF	-
, (single eq.)	D03PAF	-
, (general sys.)	D03PGF	-

D04 NUMERICAL DIFFERENTIATION

fn of single real variable	D04AAF	DCSEVU
partial differentiation	-	DBCEVU

D05 INTEGRAL EQUATIONS

Fredholm, 2nd kind, split kernel	D05AAF	-
, smooth kernel	D05ABF	-

E01 INTERPOLATION

1 variable, equal spacing	E01ABF	-
, unequal spacing	E01AAF	ICSICU
, cubic spline	E01ADF	ICSCCU
	E01BAF	
, periodic cubic spline	-	ICSPLN
, Chebyshev polynomial	E01AEF	-
2 variables	E01ACF	IBCICU
		IBCIEU
		IQHSCV

E02 CURVE AND SURFACE FITTING

l-s curve, cubic splines	E02BAF	ICSFKU ICSVKU
, polynomial	E02ADF E02AFF	-
, Chebyshev series	E02AGF	-
, user supplied basis	-	IFLSQ
l-s surface fit	E02DAF E02CAF	-
minimax curve fit	E02ACF	-
minimax minimax soln. over-deter. lin. eq.	E02GCF	RLLMV
L1 approx. linear fn	E02GAF	RLLAV
constraints	E02GBF	-
rational approx.	-	IRATCU
Pade approximant	E02RAF	-
evaluate Chebyshev series	E02AEF	-
cubic spline	E02BBF	ICSEVU
ditto derivs.	E02BCF	DCSEVU
ditto, definite integral	E02BDF	-
poly in 2 vers, from E02CAF	E02CBF	-
bi-cubic spline	E02DBF	IBCEVU
rational fn. from E01RAF	E02RBF	-
poly. from Cheby. series	E02AKF	-
Chebyshev coeffs. of derivative	E02AHF	-
ditto of integral	E02AKF	-
sort 2-d data into panels	E02ZAF	-
generate points in n-dim space	-	ZSRCH
data smoothing, outlier detection	-	ICSMOU
, cubic spline	-	ICSSCU
(easy to use)	-	ICSSCV
, quasi Hermite	-	IQHSCU

E04 MINIMISING OR MAXIMISING A FUNCTION

1 variable, fn values	E04ABF	ZXGSN ZXGSP
, fn, 1 deriv.	E04BBF	-
1 var, easy use, fn values	E04CGF	-
, fn, 1 deriv.	E04DEF E04DFF	ZXMIN ZXCGR
, fn, 1 2 deriv.	E04EBF	-
1 var, comprehensive, fn values	E04CCF	-
, fn, 1 deriv.	E04DBF	-
bounded vars, easy use, fn values	E04JAF	-
, fn, 1 deriv.	E04KAF E04KCF	-
, fn,1 2 deriv.	E04LAF	-
bounded vars, comprehensive, fn values	E04JBF	-
, fn, 1 deriv.	E04KBF E04KDF	-
, fn, 1 2 deriv.	E04LBF	-
constrained, fn values	E04UAF	-

, fn, 1 deriv.	E04VAF E04VBF	-
, fn, 1 2 deriv.	E04WAF	-
, quadratic form	E04WAF	-
sum of squares, fn values	E04FDF	ZXSSQ
, fn, 1 deriv.	E04GCF E04GEF	-
, fn, 1 2 deriv.	E04HFF	-
sum sqs., comprehensive, fn vals.	E04FCF	-
, fn, 1 deriv.	E04GBF E04GDF	-
, fn, 1 2 deriv.	E04HEF	-
1st deriv estimation	E04HBF	-
check 1st deriv	E04HCF	
check 2nd deriv	E04HDF	-
check Jacobian	E04YAF	-
check Hessian	E04YBF	-
check 1st deriv fn constraints	E04ZAF	-
check 2nd deriv fn constraints	E04ZAF	-

F01 MATRIX OPERATIONS, INCLUDING INVERSION

invert real matrix, approximate	F01AAF	LINV1F
, accurate	-	LINV2F
invert real, sym, pos def, approx	F01ADF	LINV1P
, accurate	F01ACF	LINV2P
, simplified	F01ABF	-
, sym def band,approx	-	LIN1PB
, accurate	-	LIN2PB
, pos def	F01BPF	-
pseudo inverse and rank	F01BLF	LGINF
singular value decoomposition	F01BHF	LSVDF
, bidiagonal	F02SZF	LSVDB
QR decomposition, real, rank =n	F01AXF	-
, rank = n	F01BKF	-
LU decomposition, real	F01BTF	LUDATF
LU decomposition, real, banded	F01BMF F01LBF	-
LU decomposition, real, sparse	F01BRF F01BSF	-
LL' decomp.,real,sym,pos. def.	F01BXF	-
, band	F01MCF	-
, complex,. herm.,pos. def.	F01BNF	LUDECP
ULDL'U' decomp.,real,sym, def,band	F01BUF	-
LDL' of A E, A symm, E diag.	F01BQF	-
QU of m by n matrix	F01QAF	-
UQ of m by n matrix	F01QBF	-
reduction of real, sym Ax=kBx. B def	F01AEF	-
ABx=kx. B def	F01BDF	-
real,band,sym Ax=kBx B def	F01BVF	-
real,general Ax=kBx	-	EQZQF
complex,general Ax=kBx	-	ELZHC
balance complex matrix	F01AVF	EBALAC

balance real matrix	F01ATF	EBALAF
reduction, complex- u. Hessenberg	F01AMF	EHESSC
complex Herm- real tridiag	F01BCF	EHOUSH
real upp. Hessenberg	F01AKF	EHESSF
accumulate F01AKF products	F01APF	-
reduction real sym tridiag	F01AGF	EHOUSS
,accumulating product	F01AJF	-
,packed storage	F01AYF	-
,real sym band tridiag	F01BJF	-
(alt. storage)	F01BWF	-
,upper tri. bidiagonal	F01LZF	-
backtransformation of eigenvectors		
-complex, after balancing	F01AWF	EBBCKC
-complex, after Hessenberg reduction	F01ANF	-
-real, after balancing	F01AUF	EBBCKF
-real, after Hessenberg reductions	F01ALF	EHBCKF
-real sym., after reduction	F01AHF	-
-real sym., after reduction, packed	F01AZF	-
-Az=kBx or ABx=kx	F01AFF	-
-BAx=kx	F01BEF	-
Householder trans.,real	-	VHS12
,real, zero 1 el.	-	VHSH2R
,real, zero 2 els.	-	VHSH3R
,complex	-	VHSH2C
construct Givens rotation	-	D/SROTG
apply Givens rotation	-	D/SROT
construct modified Givens rotation	-	D/SROTMG
apply modified Givens rotation	-	D/SROTM

F02 EIGENVALUES AND EIGENVECTORS

blackbox, complex, all e-vals	F02AJF	EIGCC
, all e-vals -vecs	F02AKF	EIGCC
, selected e-vals -vecs	F02BDF	-
, complex Herm, all e-vals	F02AWF	EIGCH
, all e-vals -vecs	F02AXF	
, complex generalised Ax=kBx	F02GJF	EIGZC
A,B band, 1 e-vec	F02SDF	-
, real, all e-vals	F02AFF	EIGRF
, all e-vals -vecs	F02AGF	EIGRF
, selected e-vals -vecs	F02BCF	-
, real symm., all e-vals	F02AAF	EIGRS
, all e-vals -vecs	F02ABF	EIGRS
, selected e-vals -vecs	F02BBF	-
, band, e-vals e-vecs	-	EIGBS
, generalised Ax=kBx	F02BJF	EIGZF
, symmetric Ax=kBx, e-vals	F02ADF	-
,e-vals -vecs	F02AEF	-
complex Hessenberg,all e-vals	F02ANF	ELRH1C
,all e-vals -vecs	F02ARF	ELRH2C
,selected e-vecs	F02BLF	-
reduced complex, all e-vals -vecs	F02ARF	-
reduced complex Herm, e-vals -vecs	F02AYF	EHBCKH

reduced real, all e-vals -vecs	F02AQF	-
reduced general, complex	-	ELZVC
, real	-	EQZTF
		EQZVF
real Hessenberg, all e-vals	F02APF	EQRH3F
, selected e-vecs	F02BKF	EQRH1F
reduced real symm, all e-vals -vecs	F02AMF	EHOBKS
real symm. band, selected e-vals	F02BMF	-
real tri-diag, all e-vals	F02AVF	EQRT2S
, selected e-vals	F02BFF	EQRT1S
		EQRT3S
, selected e-vals -vecs	F02BEF	-
SVD, real upper bidiagonal	F02SZF	LSVDB
,sing.values rt-vecs. m by n(m =n)	F02WAF	-
(m= n)	F02WBF	-
,sing. values vectors m by n	F02WCF	-
QU-fact. part of SVD	F02WDF	-

F03 DETERMINANTS

black box, complex	F03ADF	-
, real	F03AAF	-
, real symm. pos. def.	F03ABF	-
, real symm. pos. def.band	F03ACF	-
lu det, complex	F03AHF	-
, real	F03AFF	-
ll' det, real, symm. pos. def.	F03AEF	-
, real symm. pos. def. band	F03AGF	LUDAPB
real banded	F03ALF	-
complex Hermitian pos. def.	F03AMF	-

F04 SIMULTANEOUS LINEAR EQUATIONS

black box, complex, approx.	F04ADF	LEQT1C
, accurate	-	LEQ2C
, real, accurate, 1 rhs	F04ATF	-
, 1 rhs	F04AEF	LEQT2F
, approx., 1 rhs	F04ARF	-
, 1 rhs	F04AAF	LEQT1F
, real sym. def,acc, 1rhs	F04ASF	-
, 1 rhs	F04ABF	LEQT2P
, approx.	-	LEQT1P
, band, approx.	F04LDF	LEQT1B
, accurate	-	LEQT2B
, real sym. def band, approx	F04ACF	LEQ1PB
(variable band)	F04MCF	-
, acc.	-	LEQ2PB
, real sym indef., approx.	-	LEQ1S
, accurate	-	LEQ2S
soln inverse, real	-	LINV3F
, real sym. def.	-	LINV3P
factorised, complex, approx	F04AKF	-

, complex Herm., approx	F04AWF	-
, real, accurate	F04AHF	LUREFF
, approx.	F04AJF	LUELMF
	F04AYF	
, real band	F04AVF	-
, real sparse, approx.	F04AXF	-
, real sym. def,accurate	F04AFF	LUREFP
, approx.	F04AGF	LUELMP
	F04AQF	
	F04AZF	
, real sym. def,band,app.	F04ALF	LUELPB
,acc.	-	LUREPB
least squares, rank n, accurate	F04AMF	LLBQF
, approx.	F04ANF	LLSQF
, rank n	F04AUF	-
, m by n m =n	F04JAF	-
m =n	F04JDF	-
,automatic treatement of rank.def.	F04JGF	-
L1, rank n	E02GAF	-

F05 ORTHOGONALIZATION

Schmidt orthogonalization	F05AAF	-
2-norm of vector	F05ABF	-

G01 SIMPLE STATISTICAL CALCULATIONS

produce a letter-value summary	-	BDLTV
descriptive, 1 variable, from data	G01AAF	BEIUGR
, from freq. table	G01ADF	BEIGRP
		BEGRPS
, 2 vars, from data	G01ABF	-
frequency table from raw data	G01AEF	BDCOU1
1-way analysis of variance	G01ACF	-
2-way conting.tab. reduction signif.	G01AFF	CTRBYC
		NHEXT
formation	-	BDCOU2
median polish of two-way table	-	BEMDP
compute exact probs. for conting table	-	CTPR
transgeneration matrix cols., in core	-	BDTRGI
, out of core	-	BDTRGO
var, co-var of linear fn., in core	-	BECVL
, out of core	-	BECVL1
plot of 2 vars (scatter plot)	G01AGF	-
plot vector against normal scores	G01AHF	-
print a box plot	-	USBOX
print stem and leaf display	-	USSLF
minimum and maximum in vector	-	USMNMX
calculation of normal scores	G01DAF	-
general cts. prob. dist. fn.	-	MDGC
ratio ordinate to normal upper tail	-	MSMRAT
distribution fn., Students t	G01BAF	MDTD

		MDTNF
, f	G01BBF	MDFD
, chi-square	G01BCF	MDCH
, beta 1st kind	G01BDF	MDBETA
, normal	S15ABF	MDNOR
, inverse Students t	G01CAF	MDSTI
, inverse f	G01CBF	MDFI
, inverse chi square	G01CCF	MDCHI
, inverse beta 1st kind	G01CDF	MDBETI
, inverse normal	G01CEF	MDNRIS
, binomial	-	MDBIN
, bivariate normal	-	MDBNOR
, non-central chi sq.	-	MDCHN
, f (real deg freedom)	-	MDFDRE
, gamma	-	MDGAM
, hypergeometric	-	MDHYP
, Kolmogorov-Smirnov asymp.	-	MDSMR
, non-central t	-	MDTN
, Poisson,terms cum. prob.	-	MDTPS
inverse of cont. pdf	-	MDGCI
generate order stats., normal dist	-	GGNO
, unif. dist	-	GGUO

G02 CORRELATION AND REGRESSION ANALYSIS

Pearson product-moment corr coeffs		
-all vars, no missing values	G02BAF	BECOVM BECORI
, casewise deletion	G02BBF	-
, pairwise deletion	G02BCF	-
-subset, no missing values	G02BGF	-
, casewise deletion	G02BHF	-
, pairwise deletion	G02BJF	-
"correlation-like" coeffs		
-all vars, no missing values	G02BDF	-
, casewise deletion	G02BEF	-
, pairwise deletion	G02BFF	-
-subset, no missing values	G02BKF	-
, casewise deletion	G02BLF	-
, pairwise deletion	G02BMF	-
Kendall Spearman coeffs		
-no missing vals, overwriting input	G02BNF	-
, preserving input	G02BQF	-
-casewise treatment, overwriting	G02BPF	-
, preserving	G02BRF	-
- pairwise treatment	G02BSF	-
linear regression, constant term,		
, no missing values	G02CAF	RLONE
, missing vals	G02CCF	RLONE
linear regression, no constant term,		
, no missing vals	G02CBF	-
, missing vals	G02CDF	-
mult.lin.reg,const term,from corrns.	G02CGF	RLMUL

,no const.,from corrns.	G02CHF	-
,from raw data	G02CJF	OFIMA3
select els. from vectors and matrices	G02CEF	RLSUBM
re-order vectors and matrix elements	G02CFF	RLSUM
means,st devs,corrns (out of score)	-	BECOR
tetrachoric correlations	-	BECTR
means,st devs,simple l.r. coeffs,st err		
(missing values, in core)	-	BEMIRI
, out of core)	-	BEMIRO
means, st devs, 3rd 4th moments		
(missing values,in core)	-	BEMMI
,out of core)	-	BEMMO
biserial/point biserial corrns	-	BESRB
biserial correlations	-	BESRN
bivariate normal corrn. est.		
from cont.table(ml method)	-	CBNRHO
generate orthog. central comp. design		RLCOMP
decode quadratic reg model	-	RLDCQM
var est for decoded orth poly coeffs	-	RLDCVA
coded	-	RLDCW
coeff decoder for orth. poly.model	-	RLDOPM
leaps and bounds - best subsets reg.	-	RLEAP
replication err d.f. s.s.(in core)	-	RLFITI
(out of core)	-	RLFITO
univ. Curvilinear fit,orth poly	-	RLFOTH
(easy use)	-	RLFOR
,with weigths	-	RLFOTW
mean correction corrected ssps,in core	-	RLGQMO
,out of core	-	RLGQMI
response control,simple lin reg model	-	RLINCF
inverse prediction	-	RLOPDC
generate orth polys	-	RLPOL
confidence int. for responses,in core	-	RLPRDI
,out of core	-	RLPRDO
residual anal for lin reg model	-	RLRES
forward stepwise regression	-	RLSTP
(easy to use)	-	RLSEP
fit y=a b*(c**x) by least squares	-	RSMITZ
log-linear fit of conting. table	-	CTLLF
inverse pred.,fitted lin. reg.model	-	RLINPF

G04 ANALYSIS OF VARIANCE

latin square design	G04ADF	ALSQAN
one-way classification	G04AEF	ACRDAN
two-way crossed classification	G04AFF	ARCBAN
two-way hierarchical classification	G04AGF	ANESTU
balanced incomplete block/lattice	-	ABIBN
contrast estimate and sums of sqs.	-	ACTRST
full factorial plan	-	AFACN
(easy to use)	-	AFACT
balanced complete design (b.c.d.)	-	AGBACP
general linear model	-	AGLMOD
interval est. variance component	-	AGVACL

expected ms. for b.c.d.	-	AGXPM
expected data by unweighted means	-	AMEANS
covariance anal. for 1-way classn.	-	ANCOV1
completely nested design,equal subcl.	-	ANESTE
,unequal subcl.	-	ANESTU
reordering data from a b.c.d.	-	AORDR
Student-Newman-Keuls test	-	ASNKMC

G05 RANDOM NUMBER GENERATORS

uniform over (0,1)	G05CAF	GGUBFS GGUBT
uniform over (a,b)	G05DAF	-
exponential	G05DBF	GGEXN
logistic	G05DGF	-
normal	G05DDF	GGNML GGNFM GGNQF
lognormal	G05DEF	GGNLG
Cauchy	G05DFF	GGCAY
gamma	G05DGF	GGAMR
chi-square	G05DHF	GGCHS
Student's t	G05DJF	GGAMR
Snedecor's f	G05DKF	GGAMR
beta, 1st kind	G05DLF	GGBTR
, 2nd kind	G05DMF	-
uniform integer	G05DYF	GGUD
pseudo-random logical	G05DZF	-
Weibull generator	G05DPF	GGWIB
unif devs. from sphere surf 3,4-space	-	GGSPH
vector of uniform (0,1) devs.	-	GGUBS
uniform (0,1) with shuffling	-	GGUW
geometric deviate	-	GGEOT
Poisson gen.-frequent param changes	-	GGPON
vector of normal deviates	-	GGNML
triangular distn generator	-	GGTRA
general continuous distn.	-	GGVCR
multinomial deviate generator	-	GGMTN
integer from reference vector	G05EYF	-
set up reference vector, uniform	G05EBF	-
, Poisson	G05ECF	GGPOS
, binomial	G05EDF	GGBN
, negative binomial	G05EEF	GGBNR
, hypergeometric	G05EFF	GGHPR
reference vector from pdf or cdf	G05EXF	-
m.v. normal gen. using ref. vec.	-	GGNSM
time series ref. vect. init.	G05EGF	-
time series gen. using ref. vect.	G05EWF	FTGEN
initialise generator, repeatable	G05CBF	-
, non-repeatable	G05CCF	-
save state of generator	G05CFF	-
restore state of generator	G05CGF	-
D-squared tally	-	GTDDU

D-squared test	-	GTD2D
moments of uniform random numbers	-	GTMNT
test for normality	-	GTNOR
prob. dist. into 2 equi-prob. states	-	GTPKP
poker test tally	-	GTPL
poker test	-	GTPOK
tally of co-ords. of pairs	-	GTPR
pairs (Goods serial) test	-	GTPST
runs test	-	GTRN
tally of no of runs and down	-	GTRTN
tally for triplets test	-	GTTRT
triplets test	-	GTTT
set of binomial random deviates	-	GGBN
set of discrete random devs., alias	-	GGDA
, table lookup	-	GGDT
set of deviates, mixture 2 exponent'ls	-	GGEXT
set of stable dist. random deviates	-	GGSTA
non. hom. Poisson process gen.	-	GGNPP
random par of 1,...,k	-	GGPER
simple random sample from finite pop.	-	GGSRS

G08 NONPARAMETRIC STATISTICS

sign test	G08AAF	NBSIGN
Wilcoxon test	G08ABF	NRWMD
median test	G08ACF	-
Mann-Whitney test	G08ADF	NRWRST
Friedman test	G08AEF	-
Kruskal-Wallis test	G08AFF	NAK1
Kolmogorov-Smirnov 1-sample test	G08CAF	NKS1
Kendall coeff of concordance	G08DAF	NMCC
Mood's and David's tests	G08BAF	-
Wilson's anova, 2,3 way, no reps	-	NAWNRP
, 1,2,3 way, equ.reps.	-	NAWRPE
, uneq.reps.	-	NAWRPU
Noether's test for cyclical trend	-	NBCYC
Cochran q test	-	NBQT
Cox Stuart sign test for trends	-	NBSDL
nonparametric pdf estimation	-	NDMPLE
includance test	-	NHINC
Kolmogorov-Smirnov 2-sample test	-	NKS2
Kendall's test for correlation	-	NMKEN
significance of Kendall correlation	-	NMKSF
k sample trends test	-	NMKTS
Bhapkar v test	-	NRBHA
ranking a vector	-	NMRANK
computing tie statistics	-	NMTIE
evaluate p.d.f.	-	NDEST
nonparametric p.d.f. estimation	-	NDKER

G09 PARAMETER ESTIMATION

interval est. of p (binomial)	-	BELBIN
lambda (poisson)	-	BELPOS
mean inf.,normal dist.,known var.	-	BEMNON
mean and var.inferences, normal	-	BEMSON
var inf.,normal sample,known mean	-	BENSON
mean and var inf.,2 norm.,uneq.var	-	BEPAT
, equ.var	-	BEPET
ml est norm.params from censored data	-	OTMLNR

G12 HYPOTHESIS TESTING

Chi-squared goodness of fit	-	GFIT
sample size/no class interval,chi-sq	-	GTCN

G13 TIME SERIES ANALYSIS

Box-Jenkins univariate modelling		
-mean,var,autocov.autocorr,par.cor.	-	FTAUTO
-AR params prelim. estimation	-	FTARPS
-MA params prelim. estimation	-	FTMPS
-transforms,diff,seasonal diff.	-	FTRDIF
-AR MA parameter estimation	-	FTMXL
-model analysis	-	FTCMP
-forecasting	-	FTCAST
trasfer functions		
-cross correlation of 2 series	-	FTCRXY
-prelim est fir transfer fn model	-	FTTRN
means,vars,cross-cov -cor:2 n-ch ser.	-	FTCROS
FFT power and cross spectra	-	FTFRS
single/multi chan tsa,time freq dom	-	FTFREQ
Kalman filtering	-	FTKALM
Wiener forecasting	-	FTWEIN
multichannel Wiener forecast	-	FTWENW
ML par est mult chan 1 o/p ts model	-	FTWENX

MULTIVARIATE TECHNIQUES

cluster analysis	-	OCLINK
discriminant anal,linear a la Fisher	-	ODFISH
,mv normal linear	-	ODNORM
factor/pca,score coeffs	-	OFCOEF
,unrot.factor loading	-	OFCOMM
,factor rot.,oblique axes	-	OFHARR
,unrot.fact.load (image)	-	OFIMAG
(princ.comp.mod.)	-	OFPPI
,oblique trans fact loading	-	OFPROT
,communalities norm.fact.res.cor mx	-	OFRESI
,orthog fact rot.(qu-,var-,equ-max)	-	OFROTA
(target matrix)	-	OFSCHN
pairwise dist. between cols of matrix	-	OCDIS
fact scores form fact coeffs	-	OFSCOR

principal component calculation	-	OFPRINC
Wilks test for m.v. norm indep.	-	OIND

SAMPLING

simple random sampling,prop. data	-	SSPAND
,cont. data	-	SSSAND
,cont. data,ratio/reg	-	SSRAND
strat. random sampling, prop. data	-	SSPBLK
, cont. data	-	SSSBLK
,cont. data,ratio/reg	-	SSRBLK
1-stage clust. sampling, cont. data	-	SSSCAN
2-stage sampling, cont. data	-	SSSEST

H OPERATIONS RESEARCH

lin. prog., simplex, 1 iteration	H01ABF	-
, revised simplex	H01ADF	ZXOLP
	H01BAF	ZX4LP
, contracted simplex	H01AEF	ZX3LP
, find pt. given lin. constraints	H01AFF	-
quadratic programming	H02AAF	-
integer linear programming	H02BAF	-
transportation problem	H03ABF	-

M01 SORTING

vector, real, ascending	M01ANF	VSRTA
, descending	M01APF	-
, absolute values	-	VSRTM
, integers, ascending	M01AQF	-
, integers, descending	M01ARF	-
, characters, alphanumeric	M01BBF	-
, reverse alphanumeric	M01BAF	-
vector index, real, ascending	M01AJF	VSRTP
, descending	M01AKF	-
, absolute values	-	VSRTR
, integer, ascending	M01ALF	-
, descending	M01AMF	-
index to sorted, real, ascending	M01AAF	-
, descending	M01ABF	-
, integers, ascending	M01ACF	-
, descending	M01ADF	-
matrix rows, real, ascending	M01AEF	-
, descending	M01AFF	-
, integers, ascending	M01AGF	-
, descending	M01AHF	-
matrix columns, character, alphanum	M01BDF	-
, reverse	M01BCF	-

P01 ERROR TRAPPING

value of error indication	P01AAF	-

S APPROXIMATIONS OF SPECIAL FUNCTIONS

tan	S07AAF	-
arcsin	S09AAF	-
arccos	S09ABF	-
tanh	S10AAF	-
sinh	S10ABF	-
cosh	S10ACF	-
arctanh	S11AAF	-
arcsinn	S11ABF	-
arccosh	S11ACF	-
exponential integer	S13AAF	MMDSI
sine integral	S13ADF	-
cosine integral	S13ACF	-
gamma	S14AAF	MGAMA
log gamma	S14ABF	MLGAMA
logaritmic deriv of gamma fn.	-	MMPSI
cumulative normal distribution	S15ABF	MDNOR
complement of cumulative normal dist	S15ACF	-
error function	S15AEF	MERF
complement of error fn	S15ADF	MERRC
inverse complement error fn.	-	MERFC1
inverse error function	-	MERFI
Dawson's integral	S15AFF	MMDAS
Bessel function j0	S17AEF	MMBSJ0
j1	S17AFF	MMBSJ1
y0	S17ACF	-
y1	S17ADF	-
y, general order	-	MMBSYN
Airy function ai	S17AGF	-
bi	S17AHF	-
deriv. of Airy function ai	S17AJF	-
bi	S17AKF	-
modified Bessel function i0	S18AEF	MMBSI0
i1	S18AFF	MMBSI1
k0	S18ACF	MMBSK0
k1	S18ADF	MMBSK1
Fresnel integrals s(x)	S20ACF	-
c(x)	S20ADF	-
complete elliptic integral, 1st kind	S21BBF	MMDELK
, 2nd kind	S21BCF	MMDELE
, 3rd kind	S21BDF	-
Kelvin fns., order zero	-	MMKEL0
, order one	-	MMKEL1
, derivs.	-	MMKELD
decompose integer into prime factors	-	VDCPS

X01 MATHEMATICAL CONSTANTS

pi	X01AAF	-
Eulers const	X01ABF	-

X02 MACHINE CONSTANTS

smallreal	X02AAF	-
smallest positive real	X02ABF	-
maxreal	X02ACF	-
X02ABF/X02AAF	X02ADF	-
largest neg. argument for exp	X02AEF	-
smallest x;x,-x,1.0/x,-1.0/x repres.	X02AGF	-
floating point base	X02BAF	-
maxint	X02BBF	-
max n for 2**n representable	X02BCF	-
min n for 2**n representable	X02BDF	-
max decimal digits	X02BEF	-
active set size in paged environment	X02CAF	-

X03 VECTOR/MATRIX ARITHMETIC

null vector	F01CQF	-
null matrix	F01CAF	-
unit matrix	F01CBF	-
copy vector, real	F01CPF	D/SCOPY
, complex	-	CCOPY
copy vector - matrix row	F01CNF	-
copy matrix	F01CMF	-
, partial	F01CFF	-
interchange vectors, real	-	D/SSWAP
, complex	-	CSWAP
interchange matrix row/column	-	VSRTU
const*vector, real	-	D/SSCAL
, complex	-	CSCAL
, real*complex	-	CSSCAL
const*vector vector, real	-	D/SAXPY
, complex	-	CAXPY
find. el. of max. magnitude, real	-	ID/SAMAX
, complex	-	ICAMAX
max. absolute value, vector	-	VABMXF
, matrix row/col.	-	VABMXS
sum of absolute values, real	-	D/SASUM VABSMF
, complex	-	SCASUM
, real matrix row/col	-	VABSMS
vector euclidean norm, real	-	D/SNRM2
, complex	-	SCNRM2
matrix 1-norm	-	VNRMF/S1
matrix euclidean norm	-	VNRMF/S2
matrix infinity norm	-	VNRMFI
matrix addition	F01CDF	VUA..

matrix subtraction	F01CEF	VUA..
partial matrix additions	F01CGF	-
partial matrix subtraction	F01CHF	-
matrix transposition	F01CRF	VTRAN
matrix multiplication	F01CKF	VMUL..
Matrix multiplication by transpose	F01CLF	VTPROF/S
mult vector by symm.matrix (packed)	F01CSF	-
matrix polynomial	-	VPOLYF
vector convolution	-	VCONVO
matrix storage mode conversion	-	VCVT
scalar product, real	F01DEF	(D)SDOT SDSDOT VIPRFF VIPRSS
, complex unconjugated	-	C(Z)DOTU
conjugated	-	C(Z)DOTC
scalar prod. const,real,basic prec.	F01DAF	-
,additional prec.	F01DBF X03AAF	SDSDOT
,complex,basic prec.	F01DCF	-
,additional prec.	F01DDF X03ABF	-
extended precision add	-	VXADD
multiply	-	VXMUL
store	-	VXSTO

X04 INPUT/OUTPUT UTILITIES

error message unit no.	X04AAF	-
advisory message unit no.	X04ABF	-
manipulate I/O unit numbers	-	UGETIO
set message level	-	UERSET
print error nessage	-	UERTST
plot cluster (from OCHIER)	-	USCLX
input of matrix	-	USCRDM USRDM
input of vector	-	USRDV
print histogram	-	USHIST USHIUT USHV1
print results of regression	-	USLEAP
print pdf information	-	USPC
plot 2 pdf's	-	USPDF
print binary tree	-	USTREE
"help" information	-	UHELP UHELP1 UHELP2 UHELP3 UHELP4
printer plot of functions	-	USPLT
print matrix	-	USWBM USMBS USWFM USWSM
print vector	-	USWFV

MANUAL REGISTRATION

If you wish to receive updates for this green book please complete the attached form and send it to:

Computing Support Library

Ms. A. Cambon
Building 36
Div. 1 Dept. A
JRC Euratom
21020 Ispra
Italy

I wish to receive future updates for "Using the IMSL & NAG Libraries".

Name ....

Address ....

....

....

Signature ....

COMMENT SHEET

Please use this sheet to send to us information about errors and omissions you have noticed in this document. Please note any apecific errors found giving page and paragraph reference.
If you wish to receive a reply to your comments please give your name and address.
Send your completed comment sheets to:

Computing Support Library

Ms. A. Cambon
Building 36
Div. 1 Dept. A
JRC Euratom
21020 Ispra
Italy

Comment sheet for green book: "Using the IMSL & NAG Libraries

From: - Name ....

Address ....

....

....

Tele: ....

Comments

....

....

....

....

....

....

....

....

....