Usage of NAG products at CERN

Version — September 1992

Application Software Group

Computing and Networks Division

CERN Geneva, Switzerland
Copyright Notice

Users of the NAG Library are reminded that its use is subject to a licence agreement. In particular, no copies of any of its subroutines may be taken to other installations without special permission. This includes the export of any executable program containing NAG subroutine(s). Access to the source code is limited, and if any user makes any alteration to a NAG subroutine (which needs permission) the user is obliged to change the name, but copyright of the modified subroutine belongs to NAG. Should any user be interested in performing such modifications, the procedure is to inquire the CERN Program Library Office about the possibility of accessing the source codes, defining the user’s objectives and purposes.

Requests for information should be addressed to:

CERN Program Library Office  
CERN-CN Division  
CH-1211 Geneva 23  
Switzerland  
Tel. +41 22 767 4951  
Fax. +41 22 767 7155  
Bitnet: CERNLIB@CERNVM  
DECnet: VXCERN::CERNLIB (node 22.190)  
Internet: CERNLIB@CERNVM.CERN.CH

Trademark notice: All trademarks appearing in this guide are acknowledged as such.

Contact Persons: Berenice Damgaard and Miguel Marquina /CN (CERNLIB@CERNVM.CERN.CH)
Technical Realization: Michel Goossens /CN (GOOSENS@CERNVM.CERN.CH)
Berenice Damgaard/CN (BERENICE@CERNVM.CERN.CH)

Edition – September 1992
Preliminary remarks

This manual, providing the necessary information on the usage of Numerical Algorithms Group Libraries at CERN, consists of:

– A description of the CERN implementation of various NAG products is listed in Appendix A.
– A user guide for the Fortran Library, describing how to access the library and run examples on the different systems.
– A user guide for the Graphics Library, describing how to access the library and run examples on the different systems.
– A tutorial example of usage of the Graphics Library. (The routines included in the Graphics Library are listed in Appendix C)
– A reprint of parts of the NAG Fortran Library Introductory Guide.
– Reproduction of eps files obtained when running examples of NAG Graphics usage.

The latest versions of the libraries are installed on VXCERN, CERNVM, CERNAPO and CSF services. Other installations are made upon request.

Throughout this manual, commands to be entered are underlined

Acknowledgements

B. Damgaard would like to thank G. Folger, M. Franceschi, M. Goossens, M. Marquina, I. McLaren, H.R. Renshall and J. Shiers for their help, and NAG Ltd. for kindly permitting us to reprint parts of their manual in Appendix B.

About the documentation

This document has been produced using \TeX with the cernman style developed at CERN. PostScript files, containing a printable version of each of the routines described in this manual, can be obtained from CERN by anonymous ftp as follows (commands to be typed by the user are underlined):

```plaintext
ftp asis01.cern.ch
Trying 128.141.201.136...
Connected to asis01.cern.ch.
220 asis01 FTP server (SunOS 4.1) ready.
Name (asis01:username): anonymous
Password: your_mailaddress
ftp> cd doc/cernlib
ftp> get naglib.ps
ftp> quit
```
# Table of Contents

1 Introduction .............................................. 1

2 Use of the NAG Fortran Library at CERN ................. 2
   2.1 Access to NAG Fortran Library programs at CERN ..... 2
   2.2 Overview of the Fortran Library ...................... 2
   2.3 Access to the Fortran Library examples: the NAGTEST utility . 4

3 Use of the NAG Graphics Library at CERN ............... 5
   3.1 Access to NAG Graphics Library programs at CERN .... 5
   3.2 Getting started with NAG graphics .................... 5
   3.3 Access to the Graphics Library examples: the NAGTEST utility .... 5

A Summary table of NAG products available at CERN ........ 10

B Essential introduction to the NAG Fortran library .......... 13

C NAG Graphics Library concise summary ................... 31

D Examples of NAGTEST runs of NAG graphics on CERNVM ... 35
Chapter 1: Introduction

This manual describes the CERN implementation of the Numerical Algorithms Group Program Library. This includes the NAG Fortran Library, the NAG Graphics Library, the Online Information Supplement, the PC/Workstation Library and the PC Graphics Library. More detailed information can be found in Appendix A.

This document is provided solely for internal use at CERN, and should be sufficient for most users of the library. Main parts of the introductory guide to the mathematical part of the library are reproduced in Appendix B, by kind permission of NAG Ltd. Reference copies of the complete manuals are available in the Computer Science Library and the Theory Division Library for inspection. Additional sets have been distributed to several groups. Type `XFIND NAG MANUALS` on CERNVM or VXCERN for the precise location of all sets. Users interested in buying their own copies are referred to contact NAG directly:

NAG Response Center
Telephone: +44 865 311744
Telefax: +44 865 311755

The NAG organization has granted CERN a site-wide licence for all systems, for which there is an application available, as they are listed in Appendix A. Users wishing to install the NAG library on their local machine should contact the CERN Program Library Office.

An interactive help facility called NAGHELP, which includes access to the subroutine calling sequence descriptions (for both the graphical and the mathematical parts of the NAG library) is also provided by NAG. This facility has been installed on CERNVM, VXCERN, CERNAPO and CSF services. In addition, the same documentation is available on VXCERN in the usual VMS style.

The way to access the online help documents on the different CERN services is described below. We recommend the use of NAGHELP on those systems where it is implemented, since it provides access to the full description of the routines.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>NAG Tool</th>
<th>Further Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERNVM</td>
<td>NAGHELP</td>
<td>XFIND NAG</td>
</tr>
<tr>
<td>VXCERN</td>
<td>NAGHELP</td>
<td>HELP NAG or XFIND NAG</td>
</tr>
<tr>
<td></td>
<td>HELP @NAGLIB</td>
<td>and the .DOC files in the NAG directories</td>
</tr>
<tr>
<td>UNIX</td>
<td>NAGHELP</td>
<td>files in the NAG directory</td>
</tr>
</tbody>
</table>

There are 932 examples on the NAG directories which show the use of the Fortran Library and of the Graphics Library. In the case of the mathematical part, the expected results of the given tests are also available for comparison. For the graphics examples, the 74 example results are shown in Appendix D. How to access and run these examples is described in Chapters 1.3 and 2.3.
Chapter 2: Use of the NAG Fortran Library at CERN

2.1 Access to NAG Fortran Library programs at CERN

Access to the NAG Fortran Library is granted on all systems via the CERNLIB command:

```
CERNLIB NAGLIB
```

Then the standard loading procedures must be followed to generate an executable:

```
SYSTEM  COMMAND
CERNVM
  CERNLIB NAGLIB
  LOAD myobj (NOAUTO)
VXCERN
  CERNLIB NAGLIB
  LINK myobj,'LIB$
UNIX - Bourne/Korn shell
  CERNLIB='cernlib naglib'
  load_command myobj.o $CERNLIB
UNIX - C shell
  set CERNLIB='cernlib naglib'
  load_command myobj.o $CERNLIB
```

Remember that all floating point arguments to the NAG Library subroutines must be of type DOUBLE PRECISION. Not using double precision may lead to access or memory violation errors which abort your program, or just give wrong results.

2.2 Overview of the Fortran Library

Below we give a short overview of the chapters included in the NAG Fortran Library. For the complete list of all the routines included in each chapter, please refer to the NAG reference manuals or the online summaries.

Contents of NAG Fortran Library

- A02 = Complex Arithmetic
- C02 = Zeros of Polynomials
- C05 = Roots of One or More Transcendental Equations
- C06 = Summation of Series
- D01 = Quadrature
- D02 = Ordinary Differential Equations
- D02M - D02N = Integrators for Stiff Ordinary Differential Equations
- D03 = Partial Differential Equations
- D04 = Numerical Differentiation
- D05 = Integral Equations
- E01 = Interpolation
- E02 = Curve and Surface Fitting
2.2. Overview of the Fortran Library

E04 = Minimizing or Maximizing a Function
F    = Linear Algebra
  F01 = Matrix Operations, Including Inversion
  F02 = Eigenvalues and Eigenvectors
  F03 = Determinants
  F04 = Simultaneous Linear Equations
  F05 = Orthogonalisation
  F06 = Linear Algebra Support Routines
  F07 = Linear Equations (LAPACK)
G01 = Simple Calculations on Statistical Data
G02 = Correlation and Regression Analysis
G03 = Multivariate Methods
G04 = Analysis of Variance
G05 = Random Number Generators
G07 = Univariate Estimation
G08 = Nonparametric Statistics
G12 = Survival Analysis
G13 = Time Series Analysis
H    = Operations Research
M01 = Sorting
P01 = Error Trapping
S    = Approximations of Special Functions
X01 = Mathematical Constants
X02 = Machine Constants
X03 = Innerproducts
X04 = Input/Output Utilities
X05 = Date and Time Utilities

The routines of each of the above mentioned chapters have names like CCCxyz, where the first three characters are the chapter name and the last three specify the given routine.

The F07 chapter consists of the LAPACK package, which was introduced at Mark 15 of the NAG Fortran Library. The chapter contains subroutines written in Fortran for solving the most common problems in numerical linear algebra: systems of linear equations, linear least squares problems, eigenvalue problems, and singular value problems. The algorithms and software are structured to achieve high efficiency on vector processors, high-performance "superscalar" workstations, and shared-memory multi-processors. While the LAPACK project has been concerned with high performance computers, the routines do not compromise efficiency on conventional machines.

NAG recommends that the users read the following minimum reference material before calling any library routine:

(a) Essential Introduction
(b) Chapter Introduction
(c) Routine Introduction
(d) Implementation-specific Users’ Note

All the items are included in the NAG Fortran Library Manuals. Besides, online access to (a) and (d) is provided as described in the introductory section of this manual.
2.3 Access to the Fortran Library examples: the NAGTEST utility.

NAGTEST is an application developed by CERN to manipulate the example programs offered by NAG. NAGTEST

- extracts the source codes of the chosen example program;
- extracts the corresponding input data, when necessary;
- compiles and runs the example;
- optionally extracts the program results as supplied by NAG, and compares them with the ones just obtained;
- makes all the relevant files accessible to the user, facilitating adaptations of the examples to specific problems.

Online information about the NAGTEST utility can be obtained via the local HELP facility. The syntax is described in the following table.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERNVM</td>
<td>NAGTEST CCCxyz [ (COMPARE )]</td>
</tr>
<tr>
<td>VXCERN</td>
<td>NAGTEST[/COMPARE] CCCxyz</td>
</tr>
<tr>
<td>UNIX</td>
<td>nagtest [-compare] cccxyz</td>
</tr>
</tbody>
</table>

where CCCxyz is the name of the routine for which an example is wanted, following the NAG names scheme explained in section 1.2.
Chapter 3: Use of the NAG Graphics Library at CERN

3.1 Access to NAG Graphics Library programs at CERN

This implementation of the NAG Graphics Library uses the Graphics Kernel System (GKS) available at CERN. The high-level routines as well as the utility and interface routines are in chapter J in the NAG classification scheme.

The NAG programs themselves are accessed as follows:

To make the library (and the required GKS library) available for loading type:

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERNVM</td>
<td>CERNLIB NAGLIB(GKS)</td>
</tr>
<tr>
<td>VXCERN</td>
<td>CERNLIB NAGLIB/GKS</td>
</tr>
</tbody>
</table>

Remember that all floating point arguments to the NAG Library subroutines must be of type DOUBLE PRECISION. The loading procedure is similar to the one described in the previous chapter, ie. with the commands:

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERNVM</td>
<td>CERNLIB NAGLIB/GKS</td>
</tr>
<tr>
<td>VXCERN</td>
<td>CERNLIB NAGLIB(GKS)</td>
</tr>
<tr>
<td></td>
<td>LINK myobj,'LIB'$</td>
</tr>
</tbody>
</table>

On UNIX systems the Graphics Library may be installed separately should this be necessary. A GKS library or equivalent would be necessary.

3.2 Getting started with NAG graphics

This section should be read in conjunction with the ”GENERAL INTRODUCTION - Using the NAG Graphics Library” available at the Computer Science Library (Bld. 513) for reference.

An overview of the routines included in the NAG Graphics Library can be found in Appendix C of this manual.

There are 74 examples on the NAGLIB directories, which show the use of the high level NAG graphical subroutines with names like J06xyf. The procedure to run these examples is explained in section 2.3 and is very similar to the one followed with the Fortran Library.

3.3 Access to the Graphics Library examples: the NAGTEST utility

The NAGTEST utility provides in this case the following functionality:

- It extracts the source codes of the chosen example program;
- It extracts the data for an example program, when necessary;
- It compiles and runs the example;
- It gives the user the opportunity of previewing the graphical results of the example on the terminal.
Chapter 3. Use of the NAG Graphics Library at CERN

- It creates a GKS metafile and an EPS output of the corresponding picture(s) which may be further manipulated with convenient utilities.

- It makes all the relevant files accessible to the users, facilitating adaptations of the examples to specific problems.

Online information about the NAGTEST utility can be obtained via the local HELP facility. The syntax is described in the table below.

<table>
<thead>
<tr>
<th>SYSTEM</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERNVM</td>
<td>NAGTEST J06xyz</td>
</tr>
<tr>
<td>VXERN</td>
<td>NAGTEST J06xyz</td>
</tr>
</tbody>
</table>

where J06xyz is the name of the routine for which an example is wished, following the NAG names scheme explained before, with all routines of the Graphics Library beginning with J06.

For most of the high level graphical subroutines there is an example with its corresponding data file on the NAGLIB directories.

Until you get more experience, use the following model, substituting different NAG routines to work on your own data as described below:

NAGTEST J06xyz

The user is prompted for the GKS workstation type. GKS workstation types for the terminals at CERN are to be found using "XFIND GKS".

For example, to look at the demonstration of the routine J06HGF, type:

NAGTEST J06HGF

If the correct terminal type is given, you will get the graphics output of the example program for routine J06HGF at the screen. Besides, the following Program Text and Program Data are automatically accessed:

*AJ06HGF
C J06HGF EXAMPLE PROGRAM TEXT.
C .. Parameters ..
INTEGER NIN, NOUT
PARAMETER (NIN=5,NOUT=6)
INTEGER MDIM, NMAX, NRWS
PARAMETER (MDIM=7,NMAX=15,NRWS=1500)
C .. Local Scalars ..
DOUBLE PRECISION PHI, R, THETA, UMAX, UMIN, VMAX, VMIN, ZMAX, ZMIN
INTEGER I, IFAIL, J, K, LENGTH, M, N
C .. Local Arrays ..
DOUBLE PRECISION HTS(MDIM,NMAX), RWS(NRWS)
CHARACTER*4 LABELX(MDIM), LABELY(NMAX)
CHARACTER*17 LABX(MDIM), LAYY(NMAX)
3.3. Access to the Graphics Library examples: the NAGTEST utility

.C .. External Subroutines ..
EXTERNAL J06AHF, J06HGF, J06VAF, J06WAF, J06WCF, J06WZF,
       XXXXXX

.C .. Executable Statements ..

Select output channels for error messages

CALL J06VAF(1,NOUT)

Initialise plotting device

CALL XXXXXX

Read viewport co-ordinates

READ (NIN,FMT=*)
READ (NIN,FMT=*) UMIN, UMAX, VMIN, VMAX

Call NAG Graphical Interface routines to initialise
the NAG Graphics and set the viewport.
The data region is set by J06HGF

CALL J06WAF
CALL J06WCF(UMIN,UMAX,VMIN,VMAX)

Read in M and N, the number of data points
on the X and Y axes respectively

READ (NIN,FMT=*) M, N
IF (M.LE.0 .OR. M.GT.MDIM) THEN
   WRITE (NOUT,FMT=99999)
ELSE IF (N.LE.0 .OR. N.GT.NMAX) THEN
   WRITE (NOUT,FMT=99998)
ELSE

Read in array of heights

READ (NIN,FMT=*) ((HTS(I,J),J=1,N),I=1,M)

Read in the Z data limits

READ (NIN,FMT=*) ZMIN, ZMAX

Read in the axes annotation

READ (NIN,FMT=*) LENGTH
Chapter 3. Use of the NAG Graphics Library at CERN

```fortran
IF (LENGTH.GT.0) THEN
    READ (NIN,FMT=*) (LABELX(K),K=1,M)
    READ (NIN,FMT=*) (LABELY(K),K=1,N)
ELSE
    READ (NIN,FMT=*) LABX(1)
    READ (NIN,FMT=*) LABY(1)
END IF

C
C Read in viewpoint definition
C
READ (NIN,FMT=*) R, THETA, PHI

C
IFAIL = 0
IF (LENGTH.GT.0) THEN
    CALL J06HGF(HTS,MDIM,M,N,R,THETA,PHI,RWS,NRWS,ZMIN,ZMAX,
                  * LABELX,LABELY,LENGTH,IFAIL)
ELSE
    CALL J06HGF(HTS,MDIM,M,N,R,THETA,PHI,RWS,NRWS,ZMIN,ZMAX,
                  * LABX,LABY,LENGTH,IFAIL)
END IF

C
C Draw title
C
CALL J06AHF('J06HGF EXAMPLE PLOT')
END IF

C
C Terminate plotting
C
CALL GPRMPT(1,'READY?',LSTRI,REPLY)
CALL GCSTOP
STOP

C 99999 FORMAT (' M is out of range')
99998 FORMAT (' N is out of range')
END
```
3.3. Access to the Graphics Library examples: the NAGTEST utility

This and other examples are provided by NAG to demonstrate the features of their Graphics Library, and were tailored for the CERN environment. In particular, by means of the calls to routines GPRMPT and GCST0P, they terminate more properly than the original versions. Another common feature of the NAG example programs is the initialisation of plotting device by means of the routine named XXXXXX (yes: it is really called this!) by NAG. The examples call the routine XXXXXX to initialise the GKS system, and a version of this is provided to enable the NAG examples to run. We keep the name XXXXXX, but we have modified this routine in order to make a more proper initialisation. XXXXXX will direct all input/output to GKS workstation 1, which is the terminal type provided by the user. Simultaneously (or if carriage return is issued instead of providing the workstation type) all graphics output is written to a metafile (GKS workstation 2) with logical unit 25 on disk (with the names J06xyf .METAFILE on VM and J06xyf.MET on VAX). This metafile can than be played back using GRVIEW or plotted using GRPLOT. In XXXXXX we chose the logical unit for the GKS error file to be the same as for the NAG error log, namely unit 09. In this way, any GKS errors will be output to the same file as the NAG error log.

The result of this example, as well as the results of all the 74 examples provided with NAGTEST as run on CERNVM are shown in Appendix D.
## Appendix A: Summary table of NAG products available at CERN

<table>
<thead>
<tr>
<th>PRODUCT CODE</th>
<th>PRODUCTION DATE</th>
<th>CONTENTS</th>
</tr>
</thead>
</table>
| FLAPT14D     | Apr-1990        | NAG Fortran Library, Mark 14  
              |                 | Apollo Domain DN10000         |
| FLIB615D     | Apr-1992        | NAG Fortran Library, Mark 15  
              |                 | IBM RISC System/6000 Double Precision |
| FLIBM15DB/CMS| Mar-1992        | NAG Fortran Library, Mark 15  
              |                 | IBM VS Version 2 Scalar Double Precision |
| FLCRX15SE    | Mar-1992        | NAG Fortran Library, Mark 15  
              |                 | Cray X-MP UNICOS Single Precision "E" |
| FLCRX14S     | Dec-1990        | NAG Fortran Library, Mark 14  
              |                 | Cray X-MP UNICOS Single Precision |
| FLDVV15D     | Aug-1991        | NAG Fortran Library, Mark 15  
              |                 | DEC VAX/VMS Double Precision    |
| FLSU315D     | Oct-1991        | NAG Fortran Library, Mark 15  
              |                 | Sun 3 Double Precision          |
| FLSU415D     | Oct-1991        | NAG Fortran Library, Mark 15  
              |                 | Sun 4 Double Precision          |
| FLD3115D     | Oct-1991        | NAG Fortran Library, Mark 15  
              |                 | DECstation Double Precision     |
| FLAPT15DA    | Jan-1992        | NAG Fortran Library, Mark 15  
              |                 | Apollo Domain Double Precision  |
| FLH9715D     | Feb-1992        | NAG Fortran Library, Mark 15  
              |                 | HP 9000 Series 700 Double Precision |
| FLH9315D     | Mar-1992        | NAG Fortran Library, Mark 15  
              |                 | HP 9000 Series 300/400 Double Precision |
| FLSG415D     | Mar-1992        | NAG Fortran Library, Mark 15  
              |                 | Silicon Graphics 4D Double Precision |
| FLIB614D     | Jul-1990        | NAG Fortran Library, Mark 14  
<pre><code>          |                 | IBM RISC System/6000 Double Precision |
</code></pre>
<table>
<thead>
<tr>
<th>Product Code</th>
<th>Date</th>
<th>Version/Platform Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLIBM14DB/CMS</td>
<td>Oct-90</td>
<td>NAG Fortran Library, Mark 14 IBM VS Version 2 Scalar Double Precision</td>
</tr>
<tr>
<td>FLCRX14SE</td>
<td>Dec-90</td>
<td>NAG Fortran Library, Mark 14 Cray X-MP UNICOS Single Precision &quot;E&quot;</td>
</tr>
<tr>
<td>GLDVV03D</td>
<td>Jun-90</td>
<td>NAG Graphics Library, Mark 3 DEC VAX/VMS Double Precision</td>
</tr>
<tr>
<td>GLIBM03DB</td>
<td>Jul-90</td>
<td>NAG Graphics Library, Mark 3 IBM VS Version 2 Double Precision</td>
</tr>
<tr>
<td>GLUNIX03DAA</td>
<td>Dec-90</td>
<td>NAG Graphics Library, Mark 3 UNIX F77</td>
</tr>
<tr>
<td>GLH9303DAA</td>
<td>Dec-90</td>
<td>NAG Graphics Library, Mark 3 HP9000 Ser 300 HP-UX F/9000</td>
</tr>
<tr>
<td>OLIBM14B/CMS</td>
<td>Apr-91</td>
<td>NAG On-line Information Supplement, Mark 14.0G3.0 IBM VM/CMS</td>
</tr>
<tr>
<td>OLDVV15B</td>
<td>Nov-91</td>
<td>NAG On-line Information Supplement, Mark 15.0G3.0 DEC VAX/VMS</td>
</tr>
<tr>
<td>OLUNX15B</td>
<td>Dec-91</td>
<td>NAG On-line Information Supplement, Mark 15.0G3.0 Generic UNIX</td>
</tr>
<tr>
<td>WLAM201DAB</td>
<td>May-92</td>
<td>NAG Fortran Workstation Library 1 Apple Macintosh II</td>
</tr>
<tr>
<td>WLIBP01DGB</td>
<td>May-92</td>
<td>NAG Fortran Workstation Library 1 IBM PC PCDOS/MSDOS - Msoft 4.10</td>
</tr>
<tr>
<td>WLIBP01DGB</td>
<td>May-92</td>
<td>NAG Fortran Workstation Library 1 IBM PC PCDOS/MSDOS - Lahey 3.00</td>
</tr>
<tr>
<td>WLIBP01DEA</td>
<td>May-92</td>
<td>NAG Fortran Workstation Library 1 IBM PC PCDOS/MSDOS - R-M 2.42</td>
</tr>
<tr>
<td>GLIBP03DI</td>
<td>Feb-92</td>
<td>NAG Graphics Library Mark 3</td>
</tr>
</tbody>
</table>
Appendix A. Summary table of NAG products available at CERN

<table>
<thead>
<tr>
<th>Product</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOS Microsoft Fortran Double Precision</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GLIBP03DE</td>
<td>Apr-1992</td>
<td>NAG Graphics Library Mark 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DOS Ryan-McFarland Fortran Double Precision</td>
</tr>
</tbody>
</table>

Appendix B: Essential introduction to the NAG Fortran library

(verbatim copy taken from NAG Introductory Guide - Mark 15)

ESSENTIAL INTRODUCTION TO THE NAG FORTRAN LIBRARY

This document is essential reading for any prospective user of the Library.

Contents:

1. The Library and its Documentation
   1.1. Structure of the Library
   1.2. Structure of the Manual
   1.3. Supplementary Documentation
   1.4. Marks of the Library
   1.5. Implementations of the Library
   1.6. Precision of the Library
   1.7. Library Identification
   1.8. Fortran Language Standards

2. Using the Library
   2.1. General Advice
   2.2. Programming Advice
   2.3. Error-handling and the Parameter IFAIL
   2.4. Input/output in the Library
   2.5. Auxiliary Routines

3. Using the Documentation
   3.1. Using the Manual
   3.2. Structure of Routine Documents
   3.3. Specifications of Parameters
      3.3.1. Classification of parameters
      3.3.2. Constraints and suggested values
      3.3.3. Array parameters
   3.4. Implementation-dependent Information
   3.5. Example Programs and Results
   3.6. Summary for New Users
   3.7. Supplementary Documentation
   3.8. Pre-Mark 14 Routine Documents

4. Contact between Users and NAG

5. Further Information

6. References
1. The Library and its Documentation

1.1. Structure of the Library

The NAG Fortran Library is a comprehensive collection of Fortran 77 routines for the solution of numerical and statistical problems. The word 'routine' is used to denote 'subroutine' or 'function'.

The Library is divided into chapters, each devoted to a branch of numerical analysis or statistics. Each chapter has a three-character name and a title, e.g.

D01 - Quadrature

Exceptionally two chapters (H and S) have one-character names. (The chapters and their names are based on the ACM modified SHARE classification index [1].)

All documented routines in the Library have six-character names, beginning with the characters of the chapter name, e.g. D01AJF Note that the second and third characters are digits, not letters; e.g. 0 is the digit zero, not the letter O. The last letter of each routine name always appears as 'F' in the documentation, but may be changed to 'E' in some single-precision implementations (see Section 1.6).

1.2. Structure of the Manual

The NAG Fortran Library Manual is the principal documentation for the NAG Fortran Library. It has the same chapter structure as the Library: each chapter of routines in the Library has a corresponding chapter (of the same name) in the Manual. The chapters occur in alphanumeric order. General introductory documents and indexes are placed at the beginning of the Manual.

Each chapter consists of the following documents:

- Chapter Introduction, e.g. Introduction - D01;
- Chapter Contents, e.g. Contents - D01;

routine documents, one for each documented routine in the chapter. A routine document has the same name as the routine which it describes. Within each chapter, routine documents occur in alphanumeric order. Exceptionally, some chapters (F06, X01, X02), which contain simple support routines, do not have individual routine documents; instead, all the routines are described together in the Chapter Introduction.
1.3. Supplementary Documentation

In addition to the full Manual, NAG provides the following alternative forms of documentation, which may be more convenient to use, but do not contain all the information and advice which is provided in the full Manual:

- the Introductory Guide
- the Concise Reference
- the On-line Information Supplement

All these forms of documentation follow the same basic structure (ordering, division into chapters) as the Manual. Further details of their contents are given in Section 3.7.

1.4. Marks of the Library

Periodically a new Mark of the NAG Fortran Library is released: new routines are added, corrections or improvements are made to existing routines; occasionally routines are withdrawn if they have been superseded by improved routines.

At each Mark, the documentation of the Library is updated. You must make sure that your documentation has been updated to the same Mark as the Library software that you are using.

Marks are numbered, e.g. 12, 13, 14. The current Mark is 15.

The Library software may be updated between Marks to an intermediate maintenance level, in order to incorporate corrections. Maintenance levels are indicated by a letter following the Mark number, e.g. 15A, 15B, and so on (Mark 15 documentation supports all these maintenance levels).

1.5. Implementations of the Library

The NAG Fortran Library is available on many different computer systems. For each distinct system, an implementation of the Library is prepared by NAG, e.g. the Cray XMP Unicos implementation. The implementation is distributed to sites as a tested compiled library.

An implementation is usually specific to a range of machines (e.g. the DEC VAX range); it may also be specific to a particular operating
system, Fortran compiler, or compiler option (e.g. scalar or vector mode).

Essentially the same facilities are provided in all implementations of the Library, but, because of differences in arithmetic behaviour and in the compilation system, routines cannot be expected to give identical results on different systems, especially for sensitive numerical problems.

The documentation supports all implementations of the Library, with the help of a few simple conventions, and a small amount of implementation-dependent information, which is published in a separate Users’ Note for each implementation (see Section 3.4).

1.6. Precision of the Library

The NAG Fortran Library is developed in both single precision and double precision versions. REAL variables and arrays in the single precision version are replaced by DOUBLE PRECISION variables and arrays in the double precision version.

On most systems only one precision of the Library is available; the precision chosen is that which is considered most suitable in general for numerical computation (double precision on most systems).

On some systems both precisions are provided: in this case, the double precision routines have names ending in ‘F’ (as in the documentation), and the single precision routines have names ending in ‘E’. Thus in DEC VAX/VMS implementations:

- D01AJF is a routine in the double precision implementation;
- D01AJE is the corresponding routine in the single precision implementation.

1.7. Library Identification

You must know which implementation, which precision and which Mark 3 of the Library you are using or intend to use. To find out which implementation, precision and Mark of the Library is available at your site, you can run a program which calls the NAG Library routine A00AAF (or A00AAE in some single precision implementations). This routine has no parameters; it simply outputs text to the NAG Library advisory message unit (see Section 2.4). An example of the output is:
1.8. Fortran Language Standards

All routines in the Library conform to ANSI Standard Fortran 77 [8], except for the use of a double precision complex data type (usually COMPLEX*16) in some routines in double precision implementations of the Library - there is no provision for this data type in the standard.

Many of the routines in the Library were originally written to conform to the earlier Fortran 66 standard [7], and their calling sequences contain a few parameters which are not strictly necessary in Fortran 77.

2. Using the Library

2.1. General Advice

A NAG Fortran Library routine cannot be guaranteed to return meaningful results, irrespective of the data supplied to it. Care and thought must be exercised in:

(a) formulating the problem;
(b) programming the use of library routines;
(c) assessing the significance of the results.

The Foreword to the Manual provides some further discussion of points (a) and (c); Sections 2.2 to 2.5 are concerned with (b).

2.2. Programming Advice

The NAG Fortran Library and its documentation are designed on the assumption that users know how to write a calling program in Fortran.

When programming a call to a routine, read the routine documentation carefully, especially the description of the Parameters. This states clearly which parameters must have values assigned to them on entry to
the routine, and which return useful values on exit. See Section 3.3 for further guidance.

The most common types of programming error in using the Library are:

- incorrect parameters in a call to a Library routine;
- calling a double precision implementation of the Library from a single precision program, or vice versa.

Therefore if a call to a Library routine results in an unexpected error message from the system (or possibly from within the Library), check the following:

Has the NAG routine been called with the correct number of parameters?

Do the parameters all have the correct type?

Have all array parameters been dimensioned correctly?

Is your program in the same precision as the NAG Library routines to which your program is being linked?

Have NAG routine names have been modified - if necessary - as described in Sections 1.6 and 2.5?

Avoid the use of NAG-type names for your own program units or COMMON blocks: in general, do not use names which contain a three-character NAG chapter name embedded in them; they may clash with the names of an auxiliary routine or COMMON block used by the NAG Library.

2.3. Error handling and the Parameter IFAIL

NAG Fortran Library routines may detect various kinds of error, failure or warning conditions. Such conditions are handled in a systematic way by the Library. They fall roughly into three classes:

(i) an invalid value of a parameter on entry to a routine;
(ii) a numerical failure during computation (e.g. approximate singularity of a matrix, failure of an iteration to converge);
(iii) a warning that although the computation has been completed, the results cannot be guaranteed to be completely reliable.

All three classes are handled in the same way by the Library, and are all referred to here simply as errors.
The error-handling mechanism uses the parameter IFAIL, which occurs in the calling sequence of most NAG Library routines (almost always it is the last parameter). IFAIL serves two purposes:

(i) it allows users to specify what action a Library routine should take if it detects an error;
(ii) it reports the outcome of a call to a Library routine, either success (IFAIL = 0) or failure (IFAIL ≠ 0, with different values indicating different reasons for the failure, as explained in Section 6 of the routine document).

For the first purpose IFAIL must be assigned a value before calling the routine; since IFAIL is reset by the routine, it must be passed as a variable, not as an integer constant. Allowed values on entry are:

IFAIL = 0: an error message is output, and execution is terminated (hard failure);

IFAIL = +1: execution continues without any error message;

IFAIL = -1: an error message is output, and execution continues. The settings IFAIL = 1 are referred to as ‘soft failure’.

The safest choice is to set IFAIL to 0, but this is not always convenient: some routines return useful results even though a failure (in some cases merely a warning) is indicated. However, if IFAIL is set to 1 on entry, it is essential for the program to test its value on exit from the routine, and to take appropriate action.

The specification of IFAIL in Section 5 of a routine document suggests a suitable setting of IFAIL for that routine.

For a full description of the error-handling mechanism, see Chapter P01.

2.4. Input/output in the Library

Most NAG Library routines perform no output to an external file, except possibly to output an error message. All error messages are written to a logical error message unit. This unit number (which is set by default to 6 in most implementations) can be changed by calling the Library routine X04AAF.

Some NAG Library routines may optionally output their final results, or intermediate results to monitor the course of computation. All output other than error messages is written to a logical advisory
message unit. This unit number (which is also set by default to 6 in most implementations) can be changed by calling the Library routine X04ABF. Although it is logically distinct from the error message unit, in practice the two unit numbers may be the same.

All output from the Library is formatted.

The only Library routines which perform input from an external file are a few option-setting routines in Chapter E04: the unit number is a parameter to the routine, and all input is formatted.

You must ensure that the relevant Fortran unit numbers are associated with the desired external files, either by an OPEN statement in your calling program, or by operating system commands.

2.5. Auxiliary Routines

In addition to those Library routines which are documented and are intended to be called by users, the Library also contains many auxiliary routines. Details of all the auxiliary routines which are called directly or indirectly by any documented NAG Library routine, are supplied to sites in machine-readable form with the Library software.

In general, you need not be concerned with them at all, although you may be made aware of their existence if, for example, you examine a memory map of an executable program which calls NAG routines. The only exception is that when calling some NAG Library routines, you may be required or allowed to supply the name of an auxiliary routine from the NAG Library as an external procedure parameter. The routine documents give the necessary details. In such cases, you only need to supply the name of the routine; you never need to know details of its parameter-list.

NAG auxiliary routines have names which are similar to the name of the documented routine(s) to which they are related, but with last letter 'Z', 'Y', and so on, e.g. D01BAZ is an auxiliary routine called by D01BAF. In a single precision implementation in which the names of documented routines end in 'E', the names of auxiliary routines have their first three and last three characters interchanged, e.g. BAZD01 is an auxiliary routine (corresponding to D01BAZ) called by D01BAE.

3. Using the Documentation

3.1. Using the Manual
The Manual is designed to serve the following functions:

- to give background information about different areas of numerical and statistical computation;
- to advise on the choice of the most suitable NAG Library routine or routines to solve a particular problem;
- to give all the information needed to call a NAG Library routine correctly from a Fortran program, and to assess the results.

At the beginning of the Manual are some general introductory documents. The following may help you to find the chapter, and possibly the routine, which you need to solve your problem:

Contents Summary Mark 15 - a structured list of routines in the Library, by chapter;
KWIC Index - a keyword index to chapters and routines;
GAMS Index - a list of NAG routines classified according to the GAMS scheme.

Having found a likely chapter or routine, you should read the corresponding Chapter Introduction, which gives background information about that area of numerical computation, and recommendations on the choice of a routine, including indices, tables or decision trees.

When you have chosen a routine, you must consult the routine document. Each routine document is essentially self-contained (it may contain references to related documents). It includes a description of the method, detailed specifications of each parameter, explanations of each error exit, remarks on accuracy, and an example program to illustrate the use of the routine.

3.2. Structure of Routine Documents

Note: at Mark 14 some changes were made to the style and appearance of routine documents. If you have a Manual which contains pre-mark 14 routine documents, you will find that it contains older documents which differ in style, although they contain essentially the same information. Sections 3.2, 3.3 and 3.5 of this Essential Introduction describe the new-style routine documents. Section 3.8 gives some details about the old-style documents.

All routine documents have the same structure, consisting of nine numbered sections:

1. Purpose
Appendix B. Essential introduction to the NAG Fortran library

2. Specification
3. Description
4. References
5. Parameters (see Section 3.3 below)
6. Error Indicators
7. Accuracy
8. Further Comments
9. Example (see Section 3.5 below)

In a few documents, Section 5 also includes a description of printed output which may optionally be produced by the routine.

3.3. Specifications of Parameters

Section 5 of each routine document contains the specification of the parameters, in the order of their appearance in the parameter list.

3.3.1. Classification of parameters

Parameters are classified as follows:

Input: you must assign values to these parameters on or before entry to the routine, and these values are unchanged on exit from the routine.

Output: you need not assign values to these parameters on or before entry to the routine; the routine may assign values to them.

Input/Output: you must assign values to these parameters on or before entry to the routine, and the routine may then change these values.

Workspace: array parameters which are used as workspace by the routine. You must supply arrays of the correct type and dimension, but you need not be concerned with their contents.

External Procedure: a subroutine or function which must be supplied (e.g. to evaluate an integrand or to print intermediate output). Usually it must be supplied as part of your calling program, in which case its specification includes full details of its parameter-list and specifications of its parameters (all enclosed in a box). Its parameters are classified in the same way as those of the Library routine, but because you must write the procedure rather than call it, the significance of the classification is different:

Input: values may be supplied on entry, which your procedure must not change.
Output: you may or must assign values to these parameters before exit from your procedure.

Input/Output: values may be supplied on entry, and you may or must assign values to them before exit from your procedure.

Occasionally, as mentioned in Section 2.5, the procedure can be supplied from the NAG Library, and then you only need to know its name.

User Workspace: array parameters which are passed by the Library routine to an external procedure parameter. They are not used by the routine, but you may use them to pass information between your calling program and the external procedure.

Dummy: a simple variable which is not used by the routine. A variable or constant of the correct type must be supplied, but its value need not be set. (A dummy parameter is usually a parameter which was required by an earlier version of the routine and is retained in the parameter-list for compatibility.)

3.3.2. Constraints and suggested values

The word ‘Constraint:’ or ‘Constraints:’ in the specification of an Input parameter introduces a statement of the range of valid values for that parameter, e.g.

Constraint: N > 0.

If the routine is called with an invalid value for the parameter (e.g. N = 0), the routine will usually take an error exit, returning a non-zero value of IFAIL (see Section 2.3).

In newer documents constraints on parameters of type CHARACTER only list uppercase alphabetic characters, e.g.

Constraint: STRING = 'A' or 'B'.

In practice all routines with CHARACTER parameters will permit the use of lower case characters.

The phrase ‘Suggested Value:’ introduces a suggestion for a reasonable initial setting for an Input parameter (e.g. accuracy or maximum number of iterations) in case you are unsure what value to use; you should be prepared to use a different setting if the suggested value turns out to be unsuitable for your problem.
3.3.3. Array parameters

Most array parameters have dimensions which depend on the size of the problem. In Fortran terminology they have ‘adjustable dimensions’: the dimensions occurring in their declarations are integer variables which are also parameters of the Library routine.

For example, a Library routine might have the specification:

```fortran
SUBROUTINE <name> (M, N, A, B, LDB)
INTEGER M, N, A(N), B(LDB,N), LDB
```

For a one-dimensional array parameter, such as A in this example, the specification would begin:

```fortran
A(N) - INTEGER array.
```

You must ensure that the dimension of the array, as declared in your calling (sub)program, is at least as large as the value you supply for N. It may be larger; but the routine uses only the first N elements.

For a two-dimensional array parameter, such as B in the example, the specification might be:

```fortran
B(LDB,N) - INTEGER array.
```

On entry: the m by n matrix B.

and the parameter LDB might be described as follows:

```fortran
LDB - INTEGER. Input
On entry: the first dimension of the array B as declared in the (sub)program from which <name> is called.
Constraint: LDB >= M.
```

You must supply the first dimension of the array B, as declared in your calling (sub)program, through the parameter LDB, even though the number of rows actually used by the routine is determined by the parameter M. You must ensure that the first dimension of the array is at least as large as the value you supply for M. The extra parameter LDB is needed because Fortran does not allow information about the dimensions of array parameters to be passed automatically to a routine.
You must also ensure that the second dimension of the array, as
declared in your calling (sub)program, is at least as large as the
value you supply for N. It may be larger, but the routine only uses
the first N columns.

A program to call the hypothetical routine used as an example in this
section might include the statements:

```
INTEGER AA(100), BB(100,50)
LDB = 100
.
.
M = 80
N = 20
CALL <name>(M,N,AA,BB,LDB)
```

Fortran requires that the dimensions which occur in array
declarations, must be greater than zero. Many NAG routines are
designed so that they can be called with a parameter like N in the
above example set to 0 (in which case they would usually exit
immediately without doing anything). If so, the declarations in the
Library routine would use the 'assumed size' array dimension, and
would be given as:

```
INTEGER M, N, A(*), B(LDB,*), LDB
```

However, the original declaration of an array in your calling program
must always have constant dimensions, greater than or equal to 1.

Consult an expert or a textbook on Fortran, if you have difficulty in
calling NAG routines with array parameters.

3.4. Implementation-dependent Information

In order to support all implementations of the Library, the Manual has
adopted a convention of using bold italics to distinguish terms which
have different interpretations in different implementations.

The most important bold italicised terms are the following; their
interpretation depends on whether the implementation is in single
precision or double precision:

<table>
<thead>
<tr>
<th>Term</th>
<th>Single Precision</th>
<th>Double Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>real</td>
<td>REAL</td>
<td>DOUBLE PRECISION</td>
</tr>
<tr>
<td>complex</td>
<td>COMPLEX</td>
<td>COMPLEX*16</td>
</tr>
</tbody>
</table>

(or equivalent)
Appendix B. Essential introduction to the NAG Fortran library

basic precision means single precision or double precision
additional precision means double precision or quadruple precision

Another important bold italicised term is which denotes the relative precision to which real floating-point numbers are stored in the computer, e.g. in an implementation with approximately 16 decimal digits of precision, has a value of approximately $10^{-16}$.

The precise value of is given by the function X02AJF. Other functions in Chapter X02 return the values of other implementation-dependent constants, such as the overflow threshold, or the largest representable integer. Refer to the X02 Chapter Introduction for more details.

For each implementation of the Library, a separate Users’ Note is published. This is a short document, revised at each Mark. At most installations it is available in machine-readable form. It gives any necessary additional information which applies specifically to that implementation, in particular:

- the interpretation of bold italicised terms;
- the values returned by X02 routines;
- the default unit numbers for output (see Section 2.4);
- details of name changes for Library routines (see Sections 1.6 and 2.5).

3.5. Example Programs and Results

The example program in the last section of each routine document illustrates a simple call of the routine. The programs are designed so that they can fairly easily be modified, and so serve as the basis for a simple program to solve a user’s own problem.

Bold italicised terms are used in the printed text of the example program, to denote precision-dependent features in the code. The correct Fortran code must be substituted before the program can be run. In addition to the terms real and complex which were explained in Section 3.4, the following are used in the example programs:

Intrinsic Functions: real means REAL or DBLE (see Note below)
imag means AIMAG or DIMAG
cmplx means CMPLX or DCMPLX
conjg means CONJG or DCONJG

Edit Descriptor: e means E or D (in FORMAT statements)
Exponent Letter: e means E or D (in constants)

Note: in some implementations, the intrinsic function real with a
complex argument must be interpreted as DREAL rather than DBLE.

For each implementation of the Library, NAG distributes the example programs in machine-readable form, with all necessary modifications already applied. Many sites make the programs accessible in this form to users.

Note that the results from running the example programs may not be identical in all implementations, and may not agree exactly with the results which are printed in the Manual and which were obtained from an Apollo DN3000 double precision implementation (with approximately 16 digits of precision).

The Users' Note for your implementation will mention any special changes which need to be made to the example programs, and any significant differences in the results.

3.6. Summary for New Users

If you are unfamiliar with the NAG Library and are thinking of using a routine from it, please follow these instructions:

(a) read the whole of the Essential Introduction;
(b) consult the Contents Summary or KWIC Index to choose an appropriate chapter or routine;
(c) read the relevant Chapter Introduction;
(d) choose a routine, and read the routine document. If the routine does not after all meet your needs, return to steps (b) or (c);
(e) read the Users' Note for your implementation;
(f) consult local documentation, which should be provided by your local support staff, about access to the NAG Library on your computing system.

You should now be in a position to include a call to the routine in a program, and to attempt to run it. You may of course need to refer back to the relevant documentation in the case of difficulties, for advice on assessment of results, and so on.

As you become familiar with the Library, some of steps (a) to (f) can be omitted, but it is always essential to:

- be familiar with the Chapter Introduction;
- read the routine document;
- be aware of the Users' Note for your implementation.

3.7. Supplementary Documentation
The Introductory Guide contains all the general introductory documents, indexes, chapter introductions and chapter contents, from the full Manual. It thus gives background information, and advice on choosing the most suitable NAG Library routine; but it does not contain any detailed specifications of the routines.

The Concise Reference contains details of the parameter-lists of all routines in the Library, and very terse (usually one-line) summaries of the specification of each parameter, and of the meaning of each error-exit. It is not an adequate substitute for the documentation in the full Manual, especially if you are trying to use a routine for the first time, but it is intended to be a compact and convenient memory-aid for users who have gained some familiarity with the Library.

The On-line Information Supplement is a machine-based 'Help' system, which describes the subject areas covered by the Library, advises on the choice of routines, and gives essential programming details for each documented routine. It contains a machine-readable version of Sections 1, 2, 5 and 6 of each routine document.

The On-Line Information Supplement is a separate product from the Library: consult local documentation to see if it is available at your site.

3.8. Pre-Mark 14 Routine Documents

You need only read this section if you have an updated Manual, which contains pre-mark 14 documents.

You will find that older routine documents appear in a somewhat different style, or even several styles if your Manual dates back to Mark 7, say. The following are the most important differences between the earlier styles and the new style introduced at Mark 14:

- before Mark 12, routine documents had 13 sections: the extra sections have either been dropped or merged with the present Section 8 (Further Comments);

- in Section 5, parameters were not classified as Input, Output and so on; the phrase 'Unchanged on exit' was used to indicate an input parameter;

- example programs were revised at Mark 12 and again at Mark 14, to take advantage of features of Fortran 77: the programs printed in older documents do not correspond exactly with those which are now
distributed to sites in machine-readable form;

- before Mark 12, the printed example programs did not use bold italicised terms; they were written in standard single precision Fortran;

- before Mark 9, the printed example results were generated on an ICL 1906A (with approximately 11 digits of precision), and between Marks 9 and 12 they were generated on an ICL 2900 (with approximately 16 digits of precision);

- before Mark 13, documents referred to 'the appropriate implementation document'; this means the same as the 'Users’ Note for your implementation'.

4. Contact between Users and NAG

For further advice or communication about the NAG Library, you should first turn to the staff of your local computer installation. This covers such matters as:

- obtaining a copy of the Users’ Note for your implementation;
- obtaining information about local access to the Library;
- seeking advice about using the Library;
- reporting suspected errors in routines or documents;
- making suggestions for new routines or features;
- purchasing NAG documentation.

Your installation may have advisory and/or information services to handle such enquiries. In addition NAG asks each installation mounting the Library to nominate a NAG site representative, who may be approached directly in the absence of an advisory service. Site representatives receive information from NAG about confirmed errors, the imminence of updates, and so on, and will forward users’ enquiries to the appropriate person in the NAG organisation if they cannot be dealt with locally. If you are unable to make contact with your local site representative, you should write to the address given in the Users’ Note or to the address given at the head of the Library Manual.
Appendix B. Essential introduction to the NAG Fortran library

5. Further Information

In the NAG Fortran Library Manual, the document Development of NAG gives general information about the NAG project, while Summary of Services gives details of other NAG products and services, including numerical subroutine libraries in Ada, Pascal and Algol 68.

In addition, references [2], [3], [4], and [5] discuss various aspects of the design and development of the NAG Library, and NAG's technical policies and organisation.

6. References

[1] Collected Algorithms from ACM,  
    Index by subject to algorithms, 1960-1976.

[2] FORD, B.  
    Transportable Numerical Software.  
    Lecture Notes in Computer Science, 142, pp. 128-140, 1982.

    The NAG Library machine.  
    Software Practice and Experience, 9, 1, pp. 65-72, 1979.

    The Evolving NAG Library Service.  
    In: 'Sources and Development of Mathematical Software',  
    W. Cowell (Ed.).  

    Computer-based Documentation for the NAG Library.  
    Lecture Notes in Computer Science, 142, pp. 91-127, 1982.

[6] HOPPER, M.J.  
    TSSD, a Typesetting System for Scientific Documents.  
    United Kingdom Atomic Energy Authority, Harwell,  

    American National Standards Institute, Publication X3.9, 1966.

    American National Standards Institute, Publication X3.9, 1978.
Appendix C: NAG Graphics Library concise summary

J06A - AXES, GRIDS, BORDERS and TITLES

J06AAF - Pair of axes for current data region, automatic annotation
J06ABF - Pair of axes for current data region, user-specified annotation
J06ACF - Grid for the current data region, automatic annotation
J06ADF - Grid for the current data region, user-specifiable annotation
J06AEF - Scaled border for current data region, automatic annotation
J06AFF - Scaled border for current data region, user-specified annotation
J06AGF - Single axis under user control
J06AHF - Plot title, centred at the top of the current data region
J06AJF - Axis title, centred at side or bottom of current data region
J06AKF - Pair of logarithmic axes for current data region, automatic annotation
J06ALF - Logarithmic grid for current data region, automatic annotation
J06AMF - Logarithmic border for current data region, automatic annotation
J06ANF - Single logarithmic axis under user control
J06APF - Single axis with user-specified position of tick marks and annotation
J06AQF - Single axis with user-supplied annotation, and control over position of tick marks and annotation

J06B - POINT PLOTTING and STRAIGHT LINE DRAWING

J06BAF - Plot data points with optional straight lines and markers
J06BBF - Linear regression line using output from G02CAF, G02CBF, G02CCF or G02CDF
J06BCF - Plots a series of data points with optional error bars
J06BYF - Key to current NAG pen styles
J06BZF - Key to current line styles and/or markers

J06C - CURVE DRAWING

J06CAF - Plot single-valued curve through data points
J06CBF - Plot single-valued curve through data points, called point-wise
J06CCF - Plot possibly multi-valued curve through data points
J06CDF - Plot possibly multi-valued curve through data points, called point-wise

J06D - ODE GRAPHICS

J06DAF - Plots a graph of components of the solution of a system of ODEs, comprehensive version, using option setting
Appendix C. NAG Graphics Library concise summary

J06E - GENERAL FUNCTION DRAWING

J06EAF - Plot user-supplied function over specified range
J06EBF - Plot user-supplied function over specified range, called point-wise
J06ECF - Plots a parametric curve

J06F - SPECIAL FUNCTION DRAWING

J06FAF - Plot cubic spline in an interval, from its B-spline representation
J06FBF - Plot polynomial represented in Chebyshev form, using output from E02ADF or E02AFF

J06G - CONTOURING

J06GAF - Easy-to-use contour map, data on regular rectangular grid
J06GBF - Comprehensive contour map, data on regular rectangular grid
J06GCF - Easy-to-use contour map, data on irregular rectangular grid
J06GDF - Comprehensive contour map, data on irregular rectangular grid
J06GEF - Easy-to-use contour map, user-supplied function
J06GFF - Comprehensive contour map, user-supplied function
J06GGF - Comprehensive contour map, using option setting, data scattered
J06GYF - Key to contour plot produced with J06GGF
J06GZF - Key to contour indices

J06H - SURFACE VIEWING

J06HAF - Easy-to-use isometric surface view, data on regular rectangular grid
J06HBF - Comprehensive isometric surface view, data on regular rectangular grid
J06HCF - Perspective surface view, data on regular rectangular grid
J06HDF - Easy-to-use perspective surface view, data on irregular rectangular grid
J06HEF - Comprehensive perspective surface view, data on irregular rectangular grid
J06HFF - Perspective surface view, sections parallel to either horizontal axis
J06HGF - Perspective view of a three-dimensional histogram
J06HHF - Annotate data points on a perspective surface view
J06HJF - Annotate individual data blocks on a 3D histogram
J06HKF - Isometric surface view of a two-dimensional function f(x,y)
J06HLF - Perspective surface view of a two-dimensional function f(x,y)
J06J - DATA PRESENTATION

J06JAF - Shaded bar chart, vertical, various styles (easy-to-use)
J06JBF - Shaded bar chart, vertical, various styles (comprehensive)
J06JCF - Shaded bar chart, variable width bars
J06JDF - Shaded bar chart, horizontal, various styles (easy-to-use)
J06JEF - Shaded bar chart, horizontal, various styles (comprehensive)
J06JFF - Shaded block chart, variable width and height blocks
J06JGF - Marker diagram, sorts data into bins
J06JHF - Marker diagram, subdivision of bins
J06JKF - Shaded pie chart, (easy-to-use)
J06JLF - Shaded pie chart, (comprehensive)
J06JMF - Shaded bar chart, vertical or horizontal, no subdivision of bins (easy-to-use)
J06JNF - Frequency distribution diagram, various styles, integer grid input
J06JPF - Frequency distribution diagram and scatter plot, various styles, x,y input data
J06JYF - Key to numeric values
J06JZF - Key to area fill styles

J06K - VECTOR FIELD PLOTTING

J06KAF - Plots a two-dimensional vector field diagram
J06KBF - Plots a three-dimensional vector field diagram
J06KCF - Plots a two-dimensional vector field diagram from a user-supplied function
J06KDF - Plots a three-dimensional vector field diagram from a user-supplied function

J06S - STATISTICAL GRAPHICS

J06SAF - Plots a time-series, and optionally user-supplied forecasts and standard error limits
J06SBF - Plots autorecorrelation or partial autorecorrelation function of a time series
J06SCF - Plots a cumulative normal probability graph. Data may be grouped together and a histogram plotted
J06SDF - Plots a linear regression line and optional confidence limits for data and line
J06SEF - Draws box and whisker plots

J06V - UTILITY ROUTINES

J06VAF - Returns or defines the unit number for error messages
J06VBF - Returns or defines the unit number for advisory messages
Appendix C. NAG Graphics Library concise summary

J06VCF - Returns or defines the unit number for command sequence output
J06VDF - Returns or defines the unit number from which Hershey fonts will be read (used only by PC Graphics implementations)
J06VEF - Returns or defines the unit number to which SAVE options should be directed
J06VFF - Reads a set of optional parameters from an external file
J06VGF - Supplies individual optional parameter for a specified routine
Appendix D: Examples of NAGTEST runs of NAG graphics on CERNVM
Appendix D. Examples of NAGTEST runs of NAG graphics on CERNVM
Appendix D. Examples of NAGTEST runs of NAG graphics on CERNVM
Appendix D. Examples of NAGTEST runs of NAG graphics on CERNVM
Appendix D. Examples of NAGTEST runs of NAG graphics on CERNVM

---

**JOSHLF EXAMPLE PLOT**

![Plot 1](image1.png)

**JOBJLF EXAMPLE PLOT**

![Plot 2](image2.png)

**JOBJLF EXAMPLE PLOT 1**

![Plot 3](image3.png)

**JOBJLF EXAMPLE PLOT 2**

![Plot 4](image4.png)

**JOBJLF EXAMPLE PLOT**

![Plot 5](image5.png)
Appendix D. Examples of NAGTEST runs of NAG graphics on CERNVM