NAG Library Routine Document

F12FGF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

Note: *this routine uses* **optional parameters** *to define choices in the problem specification. If you wish to use* default *settings for all of the optional parameters, then the option setting routine F12FDF need not be called. If, however, you wish to reset some or all of the settings please refer to Section 11 in F12FDF for a detailed description of the specification of the optional parameters.*

1 Purpose

F12FGF is the main solver routine in a suite of routines which includes F12FDF and F12FFF. F12FGF must be called following an initial call to F12FFF and following any calls to F12FDF.

F12FGF returns approximations to selected eigenvalues, and (optionally) the corresponding eigenvectors, of a standard or generalized eigenvalue problem defined by real banded symmetric matrices. The banded matrix must be stored using the LAPACK storage format for real banded nonsymmetric matrices.

2 Specification

```
SUBROUTINE F12FGF (KL, KU, AB, LDAB, MB, LDMB, SIGMA, NCONV, D, Z, LDZ,    &
                   RESID, V, LDV, COMM, ICOMM, IFAIL)

INTEGER            KL, KU, LDAB, LDMB, NCONV, LDZ, LDV, ICOMM(*), IFAIL
REAL (KIND=nag_wp) AB(LDAB,*), MB(LDMB,*), SIGMA, D(*), Z(LDZ,*),          &
                   RESID(*), V(LDV,*), COMM(*)
```

3 Description

The suite of routines is designed to calculate some of the eigenvalues, λ, (and optionally the corresponding eigenvectors, x) of a standard eigenvalue problem $Ax = \lambda x$, or of a generalized eigenvalue problem $Ax = \lambda Bx$ of order n, where n is large and the coefficient matrices A and B are banded, real and symmetric.

Following a call to the initialization routine F12FFF, F12FGF returns the converged approximations to eigenvalues and (optionally) the corresponding approximate eigenvectors and/or an orthonormal basis for the associated approximate invariant subspace. The eigenvalues (and eigenvectors) are selected from those of a standard or generalized eigenvalue problem defined by real banded symmetric matrices. There is negligible additional computational cost to obtain eigenvectors; an orthonormal basis is always computed, but there is an additional storage cost if both are requested.

The banded matrices A and B must be stored using the LAPACK storage format for banded nonsymmetric matrices; please refer to Section 3.3.2 in the F07 Chapter Introduction for details on this storage format.

F12FGF is based on the banded driver routines **dsbdr1** to **dsbdr6** from the ARPACK package, which uses the Implicitly Restarted Lanczos iteration method. The method is described in Lehoucq and Sorensen (1996) and Lehoucq (2001) while its use within the ARPACK software is described in great detail in Lehoucq *et al.* (1998). This suite of routines offers the same functionality as the ARPACK banded driver software for real symmetric problems, but the interface design is quite different in order to make the option setting clearer and to combine the different drivers into a general purpose routine.

F12FGF, is a general purpose direct communication routine that must be called following initialization by F12FFF. F12FGF uses options, set either by default or explicitly by calling F12FDF, to return the converged approximations to selected eigenvalues and (optionally):

– the corresponding approximate eigenvectors;

– an orthonormal basis for the associated approximate invariant subspace;

– both.

4 References

Lehoucq R B (2001) Implicitly restarted Arnoldi methods and subspace iteration *SIAM Journal on Matrix Analysis and Applications* **23** 551–562

Lehoucq R B and Scott J A (1996) An evaluation of software for computing eigenvalues of sparse nonsymmetric matrices *Preprint MCS-P547-1195* Argonne National Laboratory

Lehoucq R B and Sorensen D C (1996) Deflation techniques for an implicitly restarted Arnoldi iteration *SIAM Journal on Matrix Analysis and Applications* **17** 789–821

Lehoucq R B, Sorensen D C and Yang C (1998) *ARPACK Users' Guide: Solution of Large-scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods* SIAM, Philidelphia

5 Arguments

1: KL – INTEGER *Input*

On entry: the number of subdiagonals of the matrices A and B.

Constraint: KL ≥ 0.

2: KU – INTEGER *Input*

On entry: the number of superdiagonals of the matrices A and B. Since A and B are symmetric, the normal case is KU = KL.

Constraint: KU ≥ 0.

3: AB(LDAB, $*$) – REAL (KIND=nag_wp) array *Input*

Note: the second dimension of the array AB must be at least $\max(1, \mathrm{N})$ (see F12FFF).

On entry: must contain the matrix A in LAPACK banded storage format for nonsymmetric matrices (see Section 3.3.4 in the F07 Chapter Introduction).

4: LDAB – INTEGER *Input*

On entry: the first dimension of the array AB as declared in the (sub)program from which F12FGF is called.

Constraint: LDAB $\geq 2 \times \mathrm{KL} + \mathrm{KU} + 1$.

5: MB(LDMB, $*$) – REAL (KIND=nag_wp) array *Input*

Note: the second dimension of the array MB must be at least $\max(1, \mathrm{N})$ (see F12FFF).

On entry: must contain the matrix B in LAPACK banded storage format for nonsymmetric matrices (see Section 3.3.4 in the F07 Chapter Introduction).

6: LDMB – INTEGER *Input*

On entry: the first dimension of the array MB as declared in the (sub)program from which F12FGF is called.

Constraint: LDMB $\geq 2 \times \mathrm{KL} + \mathrm{KU} + 1$.

7: SIGMA – REAL (KIND=nag_wp) *Input*

On entry: if one of the **Shifted Inverse** (see F12FDF) modes has been selected then SIGMA contains the real shift used; otherwise SIGMA is not referenced.

8: NCONV – INTEGER *Output*

On exit: the number of converged eigenvalues.

9: D(∗) – REAL (KIND=nag_wp) array *Output*

Note: the dimension of the array D must be at least NCV (see F12FFF).

On exit: the first NCONV locations of the array D contain the converged approximate eigenvalues.

10: Z(LDZ, ∗) – REAL (KIND=nag_wp) array *Output*

Note: the second dimension of the array Z must be at least NCV + 1 if the default option **Vectors** = RITZ has been selected and at least 1 if the option **Vectors** = NONE or SCHUR has been selected (see F12FFF).

On exit: if the default option **Vectors** = RITZ (see F12FDF) has been selected then Z contains the final set of eigenvectors corresponding to the eigenvalues held in D. The real eigenvector associated with eigenvalue $i-1$, for $i = 1, 2, \ldots,$ NCONV, is stored in the ith column of Z.

11: LDZ – INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which F12FGF is called.

Constraints:

if the default option **Vectors** = Ritz has been selected, LDZ ≥ N;
if the option **Vectors** = None or Schur has been selected, LDZ ≥ 1.

12: RESID(∗) – REAL (KIND=nag_wp) array *Input/Output*

Note: the dimension of the array RESID must be at least N (see F12FFF).

On entry: need not be set unless the option **Initial Residual** has been set in a prior call to F12FDF in which case RESID must contain an initial residual vector.

On exit: contains the final residual vector.

13: V(LDV, ∗) – REAL (KIND=nag_wp) array *Output*

Note: the second dimension of the array V must be at least max(1, NCV) (see F12FFF).

On exit: if the option **Vectors** (see F12FDF) has been set to Schur or Ritz and a separate array Z has been passed then the first NCONV × n elements of V will contain approximate Schur vectors that span the desired invariant subspace.

The jth Schur vector is stored in the ith column of V.

14: LDV – INTEGER *Input*

On entry: the first dimension of the array V as declared in the (sub)program from which F12FGF is called.

Constraint: LDV ≥ n.

15: COMM(∗) – REAL (KIND=nag_wp) array *Communication Array*

Note: the dimension of the array COMM must be at least max(1, LCOMM) (see F12FFF).

On initial entry: must remain unchanged from the prior call to F12FDF and F12FFF.

On exit: contains no useful information.

16: ICOMM(∗) – INTEGER array *Communication Array*

Note: the dimension of the array ICOMM must be at least max(1, LICOMM) (see F12FFF).

On initial entry: must remain unchanged from the prior call to F12FBF and F12FDF.

On exit: contains no useful information.

17: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value −1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this argument, the recommended value is 0. **When the value −1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: IFAIL = 0 unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

IFAIL = 1

On entry, KL < 0.

IFAIL = 2

On entry, KU < 0.

IFAIL = 3

On entry, LDAB < 2 × KL + KU + 1.

IFAIL = 4

Iteration Limit < 0.

IFAIL = 5

The options **Generalized** and **Regular** are incompatible.

IFAIL = 6

Eigenvalues from **Both Ends** of the spectrum were requested, but only one eigenvalue (NEV) is requested.

IFAIL = 7

The **Initial Residual** was selected but the starting vector held in RESID is zero.

IFAIL = 8

On entry, LDZ < max(1, N) or LDZ < 1 when no vectors are required.

IFAIL = 9

On entry, the option **Vectors** = Select was selected, but this is not yet implemented.

IFAIL = 10

The number of eigenvalues found to sufficient accuracy is zero.

IFAIL = 11

Could not build a Lanczos factorization. Consider changing NCV or NEV in the initialization routine (see Section 5 in F12FAF for details of these arguments).

IFAIL = 12

Unexpected error in internal call to compute eigenvalues and corresponding error bounds of the current symmetric tridiagonal matrix. Please contact NAG.

IFAIL = 13

Unexpected error during calculation of a real Schur form: there was a failure to compute all the converged eigenvalues. Please contact NAG.

IFAIL = 14

Failure during internal factorization of real banded matrix. Please contact NAG.

IFAIL = 15

Failure during internal solution of real banded system. Please contact NAG.

IFAIL = 16

The maximum number of iterations has been reached. Some Ritz values may have converged; NCONV returns the number of converged values.

IFAIL = 17

No shifts could be applied during a cycle of the implicitly restarted Lanczos iteration. One possibility is to increase the size of NCV relative to NEV (see Section 5 in F12FFF for details of these arguments).

IFAIL = 18

Either an initial call to the setup routine has not been made
or the communication arrays have become corrupted.

IFAIL = 19

The routine was unable to dynamically allocate sufficient internal workspace. Please contact NAG.

IFAIL = −99

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL = −399

Your licence key may have expired or may not have been installed correctly.

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The relative accuracy of a Ritz value, λ, is considered acceptable if its Ritz estimate $\leq$ **Tolerance** $\times |\lambda|$. The default **Tolerance** used is the ***machine precision*** given by X02AJF.

8 Parallelism and Performance

F12FGF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F12FGF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

None.

10 Example

This example solves $Ax = \lambda x$ in regular mode, where A is obtained from the standard central difference discretization of the two-dimensional convection-diffusion operator $\frac{d^2u}{dx^2} + \frac{d^2u}{dy^2} = \rho\frac{du}{dx}$ on the unit square with zero Dirichlet boundary conditions. A is stored in LAPACK banded storage format.

10.1 Program Text

```
    Program f12fgfe

!     F12FGF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: daxpy, dgbmv, dnrm2, f12fff, f12fgf, nag_wp,      &
                             x04abf, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: one = 1.0_nag_wp
      Real (Kind=nag_wp), Parameter    :: zero = 0.0_nag_wp
      Integer, Parameter               :: inc1 = 1, iset = 1, nin = 5,         &
                                          nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: h2, sigma
      Integer                          :: i, idiag, ifail, isub, isup, j, kl,  &
                                          ku, lcomm, ldab, ldmb, ldv, licomm,  &
                                          lo, n, nconv, ncv, nev, nx, outchn
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: ab(:,:), ax(:), comm(:), d(:),       &
                                          d_print(:,:), mb(:,:), resid(:),     &
                                          v(:,:)
      Integer, Allocatable             :: icomm(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: abs, int, max, real
```

```
!     .. Executable Statements ..
      Write (nout,*) 'F12FGF Example Program Results'
      Write (nout,*)

!     Skip heading in data file
      Read (nin,*)

      Read (nin,*) nx, nev, ncv
      n = nx*nx

!     Initialize communication arrays.
!     Query the required sizes of the communication arrays.

      licomm = -1
      lcomm = -1
      Allocate (icomm(max(1,licomm)),comm(max(1,lcomm)))

      ifail = 0
      Call f12fff(n,nev,ncv,icomm,licomm,comm,lcomm,ifail)

      licomm = icomm(1)
      lcomm = int(comm(1))
      Deallocate (icomm,comm)
      Allocate (icomm(max(1,licomm)),comm(max(1,lcomm)))

      ifail = 0
      Call f12fff(n,nev,ncv,icomm,licomm,comm,lcomm,ifail)

!     Construct the matrix A in banded form and store in AB.
!     KU, KL are number of superdiagonals and subdiagonals within
!     the band of matrices A and M.

      kl = nx
      ku = nx
      ldab = 2*kl + ku + 1
      Allocate (ab(ldab,n))

!     Zero out AB.

      ab(1:ldab,1:n) = 0.0_nag_wp

!     Main diagonal of A.

      h2 = one/real((nx+1)*(nx+1),kind=nag_wp)
      idiag = kl + ku + 1
      ab(idiag,1:n) = 4.0_nag_wp/h2

!     First subdiagonal and superdiagonal of A.

      isup = kl + ku
      isub = kl + ku + 2

      Do i = 1, nx
        lo = (i-1)*nx

        Do j = lo + 1, lo + nx - 1
          ab(isup,j+1) = -one/h2
          ab(isub,j) = -one/h2
        End Do

      End Do

!     KL-th subdiagonal and KU-th superdiagonal.

      isup = kl + 1
      isub = 2*kl + ku + 1

      Do i = 1, nx - 1
        lo = (i-1)*nx

        Do j = lo + 1, lo + nx
```

```
          ab(isup,nx+j) = -one/h2
          ab(isub,j) = -one/h2
        End Do

      End Do

!     Find eigenvalues of largest magnitude and the corresponding
!     eigenvectors.

      ldmb = 2*kl + ku + 1
      ldv = n
      Allocate (mb(ldmb,n),d(ncv),v(ldv,ncv+1),resid(n))

      ifail = -1
      Call f12fgf(kl,ku,ab,ldab,mb,ldmb,sigma,nconv,d,v,ldv,resid,v,ldv,comm,  &
        icomm,ifail)

      If (ifail/=0) Then
        Go To 100
      End If

!     Compute the residual norm  ||A*x - lambda*x||.

      Allocate (d_print(nconv,2),ax(n))
      d_print(1:nconv,1) = d(1:nconv)

      Do j = 1, nconv

!       The NAG name equivalent of dgbmv is f06pbf
        Call dgbmv('N',n,n,kl,ku,one,ab(kl+1,1),ldab,v(1,j),inc1,zero,ax,inc1)

!       The NAG name equivalent of daxpy is f06ecf
        Call daxpy(n,-d_print(j,1),v(1,j),inc1,ax,inc1)

!       The NAG name equivalent of dnrm2 is f06ejf
        d_print(j,2) = dnrm2(n,ax,1)
      End Do

      d_print(1:nconv,2) = d_print(1:nconv,2)/abs(d_print(1:nconv,1))

      Write (nout,*)
      Flush (nout)

      outchn = nout
      Call x04abf(iset,outchn)

      ifail = 0
      Call x04caf('G','N',nconv,2,d_print,nconv,' Ritz values and residuals',  &
        ifail)

100   Continue
    End Program f12fgfe
```

10.2 Program Data

```
F12FGF Example Program Data
 10 4 10 : Values for NX NEV and NCV
```

10.3 Program Results

```
 F12FGF Example Program Results


  Ritz values and residuals
                1           2
1    8.9117E+02   1.7472E-15
2    9.1978E+02   7.7418E-16
3    9.1978E+02   5.7476E-16
4    9.4839E+02   1.0840E-15
```