NAG Fortran Library Routine Document

F12FAF

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.

1 Purpose

F12FAF is a setup routine in a suite of routines consisting of F12FAF, F12FBF, F12FCF, F12FDF and F12FEF. It is used to find some of the eigenvalues (and optionally the corresponding eigenvectors) of a standard or generalized eigenvalue problem defined by real symmetric matrices.

The suite of routines is suitable for the solution of large sparse, standard or generalized, symmetric eigenproblems where only a few eigenvalues from a selected range of the spectrum are required.

2 Specification

```
SUBROUTINE F12FAF (N, NEV, NCV, ICOMM, LICOMM, COMM, LCOMM, IFAIL)
INTEGER              N, NEV, NCV, ICOMM(*), LICOMM, LCOMM, IFAIL
double precision     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 sparse, real and symmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, real and symmetric problems.

F12FAF is a setup routine which must be called before F12FBF, the reverse communication iterative solver, and before F12FDF, the options setting routine. F12FCF, is a post-processing routine that must be called following a successful final exit from F12FBF, while F12FEF can be used to return additional monitoring information during the computation.

This setup routine initializes the communication arrays, sets (to their default values) all options that can be set by the user via the option setting routine F12FDF, and checks that the lengths of the communication arrays as passed by the user are of sufficient length. For details of the options available and how to set them see Section 10.2 of the document for F12FDF.

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 Parameters

1: N – INTEGER *Input*

On entry: the order of the matrix A (and the order of the matrix B for the generalized problem) that defines the eigenvalue problem.

Constraint: $\mathrm{N} > 0$.

2: NEV – INTEGER *Input*

On entry: the number of eigenvalues to be computed.

Constraint: $0 < \mathrm{NEV} < \mathrm{N} - 1$.

3: NCV – INTEGER *Input*

On entry: the number of Lanczos basis vectors to use during the computation.

At present there is no a-priori analysis to guide the selection of NCV relative to NEV. However, it is recommended that $\mathrm{NCV} \geq 2 \times \mathrm{NEV} + 1$. If many problems of the same type are to be solved, you should experiment with increasing NCV while keeping NEV fixed for a given test problem. This will usually decrease the required number of matrix-vector operations but it also increases the work and storage required to maintain the orthogonal basis vectors. The optimal 'cross-over' with respect to CPU time is problem dependent and must be determined empirically.

Constraint: $\mathrm{NEV} < \mathrm{NCV} \leq \mathrm{N}$.

4: ICOMM($*$) – INTEGER array *Communication Array*
5: LICOMM – INTEGER *Input*

On entry: the dimension of the array ICOMM as declared in the (sub)program from which F12FAF is called.

If $\mathrm{LICOMM} = -1$, a workspace query is assumed and the routine only calculates the required dimension of ICOMM, which it returns in $\mathrm{ICOMM}(1)$.

Constraint: $\mathrm{LICOMM} \geq 140$.

6: COMM($*$) – ***double precision*** array *Communication Array*
7: LCOMM – INTEGER *Input*

On entry: the dimension of the array COMM as declared in the (sub)program from which F12FAF is called.

If $\mathrm{LCOMM} = -1$, a workspace query is assumed and the routine only calculates the required dimension of COMM, which it returns in $\mathrm{COMM}(1)$.

Constraint: $\mathrm{LCOMM} \geq 3 \times \mathrm{N} + \mathrm{NCV} \times \mathrm{NCV} + 8 \times \mathrm{NCV} + 60$.

8: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

On exit: $\mathrm{IFAIL} = 0$ unless the routine detects an error (see Section 6).

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, for users not familiar with this parameter the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

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, N ≤ 0.

IFAIL $= 2$

On entry, NEV ≤ 0.

IFAIL $= 3$

On entry, NCV $\leq$ NEV or NCV $>$ N.

IFAIL $= 4$

On entry, LICOMM < 140 and LICOMM $\neq -1$.

IFAIL $= 5$

On entry, LCOMM $< 3 \times$ N $+$ NCV $\times$ NCV $+ 8 \times$ NCV $+ 60$ and LCOMM $\neq -1$.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

The example solves $Ax = \lambda x$ in regular mode, where A is obtained from the standard central difference discretization of the Laplacian operator $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$ on the unit square, with zero Dirichlet boundary conditions. Eigenvalues of smallest magnitude are selected.

9.1 Program Text

```
*     F12FAF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          IMON, IPOINT, LICOMM, NIN, NOUT
      PARAMETER        (IMON=0,IPOINT=0,LICOMM=134,NIN=5,NOUT=6)
      INTEGER          MAXN, MAXNCV, LDV
      PARAMETER        (MAXN=256,MAXNCV=30,LDV=MAXN)
      INTEGER          LCOMM
      PARAMETER        (LCOMM=3*MAXN+MAXNCV*MAXNCV+8*MAXNCV+60)
*     .. Local Scalars ..
      DOUBLE PRECISION SIGMA
      INTEGER          IFAIL, IREVCM, J, N, NCONV, NCV, NEV, NITER,
     +                 NSHIFT, NX
*     .. Local Arrays ..
      DOUBLE PRECISION AX(MAXN), COMM(LCOMM), D(MAXNCV,2), MX(MAXN),
     +                 RESID(MAXN), V(LDV,MAXNCV), X(MAXN)
      INTEGER          ICOMM(LICOMM)
*     .. External Functions ..
      DOUBLE PRECISION DNRM2
      EXTERNAL         DNRM2
*     .. External Subroutines ..
```

```
      EXTERNAL         AV, DCOPY, F12FAF, F12FBF, F12FCF, F12FDF, F12FEF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F12FAF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) NX, NEV, NCV
      N = NX*NX
      IF (N.LT.1 .OR. N.GT.MAXN) THEN
         WRITE (NOUT,99999) 'N is out of range: N = ', N
      ELSE IF (NCV.GT.MAXNCV) THEN
         WRITE (NOUT,99999) 'NCV is out of range: NCV = ', NCV
      ELSE
         IFAIL = 0
         CALL F12FAF(N,NEV,NCV,ICOMM,LICOMM,COMM,LCOMM,IFAIL)
*     Set the region of the spectrum that is required.
         CALL F12FDF('SMALLEST MAGNITUDE',ICOMM,COMM,IFAIL)
*     Increase the iteration limit if required.
         CALL F12FDF('ITERATION LIMIT=500',ICOMM,COMM,IFAIL)
         IF (IPOINT.EQ.1) THEN
*        Use pointers to Workspace in calculating matrix vector products
*        rather than interfacing through the array X
            CALL F12FDF('POINTERS=YES',ICOMM,COMM,IFAIL)
         END IF
         IREVCM = 0
         IFAIL = -1
   20    CONTINUE
         CALL F12FBF(IREVCM,RESID,V,LDV,X,MX,NSHIFT,COMM,ICOMM,IFAIL)
         IF (IREVCM.NE.5) THEN
            IF (IREVCM.EQ.-1 .OR. IREVCM.EQ.1) THEN
*           Perform matrix vector multiplication y <--- Op*x
               IF (IPOINT.EQ.0) THEN
                  CALL AV(NX,X,AX)
                  CALL DCOPY(N,AX,1,X,1)
               ELSE
                  CALL AV(NX,COMM(ICOMM(1)),COMM(ICOMM(2)))
               END IF
            ELSE IF (IREVCM.EQ.4 .AND. IMON.NE.0) THEN
*           Output monitoring information
               CALL F12FEF(NITER,NCONV,D,D(1,2),ICOMM,COMM)
               WRITE (6,99998) NITER, NCONV, DNRM2(NEV,D(1,2),1)
            END IF
            GO TO 20
         END IF
         IF (IFAIL.EQ.0) THEN
*        Post-Process using F12FCF to compute eigenvalue/vectors.
            CALL F12FCF(NCONV,D,V,LDV,SIGMA,RESID,V,LDV,COMM,ICOMM,
     +                  IFAIL)
            WRITE (NOUT,99996) NCONV
            DO 40 J = 1, NCONV
               WRITE (NOUT,99995) J, D(J,1)
   40       CONTINUE
         ELSE
            WRITE (NOUT,99997) IFAIL
         END IF
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,'Iteration',1X,I3,', No. converged =',1X,I3,', norm o',
     +       'f estimates =',E16.8)
99997 FORMAT (1X,' NAG Routine F12FBF Returned with IFAIL = ',I6)
99996 FORMAT (1X,/' The ',I4,' Ritz values of smallest magnitude are:',
     +       /)
99995 FORMAT (1X,I8,5X,F12.4)
      END
*
      SUBROUTINE AV(NX,V,W)
*     .. Parameters ..
      DOUBLE PRECISION ONE
      PARAMETER        (ONE=1.0D+0)
```

```
*     .. Scalar Arguments ..
      INTEGER       NX
*     .. Array Arguments ..
      DOUBLE PRECISION V(NX*NX), W(NX*NX)
*     .. Local Scalars ..
      DOUBLE PRECISION H2
      INTEGER       J, LO
*     .. External Subroutines ..
      EXTERNAL      DAXPY, DSCAL, TV
*     .. Intrinsic Functions ..
      INTRINSIC     DBLE
*     .. Executable Statements ..
      H2 = ONE/DBLE((NX+1)*(NX+1))
      CALL TV(NX,V(1),W(1))
      CALL DAXPY(NX,-ONE,V(NX+1),1,W(1),1)
      DO 20 J = 2, NX - 1
         LO = (J-1)*NX
         CALL TV(NX,V(LO+1),W(LO+1))
         CALL DAXPY(NX,-ONE,V(LO-NX+1),1,W(LO+1),1)
         CALL DAXPY(NX,-ONE,V(LO+NX+1),1,W(LO+1),1)
   20 CONTINUE
      LO = (NX-1)*NX
      CALL TV(NX,V(LO+1),W(LO+1))
      CALL DAXPY(NX,-ONE,V(LO-NX+1),1,W(LO+1),1)
      CALL DSCAL(NX*NX,ONE/H2,W,1)
      RETURN
      END
*
      SUBROUTINE TV(NX,X,Y)
*     Compute the matrix vector multiplication y<---T*x where T is a nx
*     by nx tridiagonal matrix.
*     .. Parameters ..
      DOUBLE PRECISION ONE
      PARAMETER     (ONE=1.0D+0)
*     .. Scalar Arguments ..
      INTEGER       NX
*     .. Array Arguments ..
      DOUBLE PRECISION X(NX), Y(NX)
*     .. Local Scalars ..
      DOUBLE PRECISION DD, DL, DU
      INTEGER       J
*     .. Executable Statements ..
      DD = 4.0D+0
      DL = -ONE
      DU = -ONE
      Y(1) = DD*X(1) + DU*X(2)
      DO 20 J = 2, NX - 1
         Y(J) = DL*X(J-1) + DD*X(J) + DU*X(J+1)
   20 CONTINUE
      Y(NX) = DL*X(NX-1) + DD*X(NX)
      RETURN
      END
```

9.2 Program Data

```
F12FAF Example Program Data
 10 4 10 : Values for NX NEV and NCV
```

9.3 Program Results

```
 F12FAF Example Program Results

 The    4 Ritz values of smallest magnitude are:

        1          19.6054
        2          48.2193
        3          48.2193
        4          76.8333
```