

NAG Fortran Library Routine Document

F12FEF

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

F12FEF can be used to return additional monitoring information during computation. It is in a suite of routines consisting of F12FEF, F12FAF, F12FBF, F12FCF and F12FDF.

2 Specification

```
SUBROUTINE F12FEF (NITER, NCONV, RITZ, RZEST, ICOMM, COMM)
INTEGER          NITER, NCONV, ICOMM(134)
double precision RITZ(*), RZEST(*), 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.

On an intermediate exit from F12FBF with IREVCM = 4, F12FEF may be called to return monitoring information on the progress of the Arnoldi iterative process. The information returned by F12FEF is:

- the number of the current Arnoldi iteration;
- the number of converged eigenvalues at this point;
- the real and imaginary parts of the converged eigenvalues;
- the error bounds on the converged eigenvalues.

F12FEF does not have an equivalent routine from the ARPACK package which prints various levels of detail of monitoring information through an output channel controlled via a parameter value (see Lehoucq *et al.* (1998) for details of ARPACK routines). F12FEF should not be called at any time other than immediately following a IREVCM = 4 return from F12FBF.

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, Philadelphia

5 Parameters

- 1: NITER – INTEGER *Output*
On exit: the number of the current Arnoldi iteration.

- 2: NCONV – INTEGER Output
On exit: the number of converged eigenvalues so far.
- 3: RITZ(*) – *double precision* array Output
Note: the dimension of the array RITZ must be at least NEV.
On exit: the first NCONV locations of the array RITZ contain the real converged approximate eigenvalues.
- 4: RZEST(*) – *double precision* array Output
Note: the dimension of the array RZEST must be at least NEV.
On exit: the first NCONV locations of the array RZEST contain the Ritz estimates (error bounds) on the real NCONV converged approximate eigenvalues.
- 5: ICOMM(134) – INTEGER array Communication Array
 ICOMM, on initial entry, must remain unchanged between calls to F12FBE.
- 6: COMM(*) – *double precision* array Communication Array
 COMM, on initial entry, must remain unchanged between calls to F12FBE.

6 Error Indicators and Warnings

None.

7 Accuracy

A Ritz value, λ , is deemed to have converged if its Ritz estimate $\leq \text{Tolerance} \times |\lambda|$. The default **Tolerance** used is the *machine precision* given by X02AJF.

8 Further Comments

None.

9 Example

The example solves $Kx = \lambda KGx$ using the **Buckling** option (see F12FDF, where K and KG are obtained by the finite element method applied to the one-dimensional discrete Laplacian operator $\frac{\partial^2 u}{\partial x^2}$ on $[0, 1]$, with zero Dirichlet boundary conditions using piecewise linear elements. The shift, σ , is a real number, and the operator used in the buckling iterative process is $OP = \text{inv}(K - \sigma KG) \times K$ and $B = K$.

9.1 Program Text

```
*      F12FEF Example Program Text
*      Mark 21 Release. NAG Copyright 2004.
*      .. Parameters ..
INTEGER          LICOMM, NIN, NOUT
PARAMETER       (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)
INTEGER          IMON, IPOINT
PARAMETER       (IMON=0, IPOINT=0)
DOUBLE PRECISION FOUR, ONE, SIX, TWO
PARAMETER       (FOUR=4.0D+0, ONE=1.0D+0, SIX=6.0D+0, TWO=2.0D+0)
*      .. Local Scalars ..
DOUBLE PRECISION H, R1, R2, SIGMA
INTEGER          IFAIL, INFO, IREVCN, J, N, NCONV, NCV, NEV,
```

```

+           NITER, NSHIFT
*   .. Local Arrays ..
  DOUBLE PRECISION AD(MAXN), ADL(MAXN), ADU(MAXN), ADU2(MAXN),
+           COMM(LCOMM), D(MAXNCV,2), MX(MAXN), RESID(MAXN),
+           V(LDV,MAXNCV), X(MAXN)
  INTEGER      ICOMM(LICOMM), IPIV(MAXN)
*   .. External Functions ..
  DOUBLE PRECISION DNRM2
  EXTERNAL      DNRM2
*   .. External Subroutines ..
  EXTERNAL      AV, DCOPY, DGTTRF, DGTTRS, F12FAF, F12FBF,
+           F12FCF, F12FDF, F12FEF
*   .. Intrinsic Functions ..
  INTRINSIC     DBLE
*   .. Executable Statements ..
  WRITE (NOUT,*) 'F12FEF Example Program Results'
  WRITE (NOUT,*)
*   Skip heading in data file
  READ (NIN,*)
  READ (NIN,*) N, NEV, NCV
  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)
*   We are solving a generalized problem
    CALL F12FDF('GENERALIZED',ICOMM,COMM,IFAIL)
*   Indicate that we are using the buckling mode.
    CALL F12FDF('BUCKLING',ICOMM,COMM,IFAIL)
    IF (IPOINT.EQ.1) THEN
      CALL F12FDF('POINTERS=YES',ICOMM,COMM,IFAIL)
    END IF
*
    H = ONE/DBLE(N+1)
    R1 = (FOUR/SIX)*H
    R2 = (ONE/SIX)*H
    SIGMA = ONE
    DO 20 J = 1, N
      AD(J) = TWO/H - SIGMA*R1
      ADL(J) = -ONE/H - SIGMA*R2
20  CONTINUE
    CALL DCOPY(N,ADL,1,ADU,1)
    CALL DGTTRF(N,ADL,AD,ADU,ADU2,IPIV,INFO)
*
    IREVCM = 0
    IFAIL = -1
40  CONTINUE
    CALL F12FBF(IREVCM,RESID,V,LDV,X,MX,NSHIFT,COMM,ICOMM,IFAIL)
    IF (IREVCM.NE.5) THEN
      IF (IREVCM.EQ.-1) THEN
*   Perform  $y \leftarrow OP*x = inv[K-SIGMA*KG]*K*x$ .
        IF (IPOINT.EQ.0) THEN
          CALL AV(N,X,MX)
          CALL DCOPY(N,MX,1,X,1)
          CALL DGTTRS('N',N,1,ADL,AD,ADU,ADU2,IPIV,X,N,INFO)
        ELSE
          CALL AV(N,COMM(ICOMM(1)),COMM(ICOMM(2)))
          CALL DGTTRS('N',N,1,ADL,AD,ADU,ADU2,IPIV,COMM(ICOMM(2)),N,INFO)
+
        END IF
      ELSE IF (IREVCM.EQ.1) THEN
*   Perform  $y \leftarrow OP*x = inv[K-sigma*KG]*K*x$ .
        IF (IPOINT.EQ.0) THEN
          CALL DCOPY(N,MX,1,X,1)
          CALL DGTTRS('N',N,1,ADL,AD,ADU,ADU2,IPIV,X,N,INFO)
        ELSE
          CALL DCOPY(N,COMM(ICOMM(3)),1,COMM(ICOMM(2)),1)
          CALL DGTTRS('N',N,1,ADL,AD,ADU,ADU2,IPIV,COMM(ICOMM(2)),N,INFO)
+

```

```

        END IF
    ELSE IF (IREVCM.EQ.2) THEN
*       Perform y <--- M*x.
        IF (IPOINT.EQ.0) THEN
            CALL AV(N,X,MX)
        ELSE
            CALL AV(N,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 40
END IF
IF (IFAIL.EQ.0) THEN
*       Post-Process using F12FCF to compute eigenvalues/vectors.
    CALL F12FCF(NCONV,D,V,LDV,SIGMA,RESID,V,LDV,COMM,ICOMM,
+           IFAIL)
    WRITE (NOUT,99996) NCONV, SIGMA
    DO 60 J = 1, NCONV
        WRITE (NOUT,99995) J, D(J,1)
60    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,' generalized Ritz values closest to ',
+           F8.4,' are:',/)
99995 FORMAT (1X,I8,5X,F12.4)
END
*
SUBROUTINE AV(N,V,W)
*   .. Parameters ..
DOUBLE PRECISION ONE, TWO
PARAMETER (ONE=1.0D+0,TWO=2.0D+0)
*   .. Scalar Arguments ..
INTEGER N
*   .. Array Arguments ..
DOUBLE PRECISION V(N), W(N)
*   .. Local Scalars ..
DOUBLE PRECISION H
INTEGER J
*   .. External Subroutines ..
EXTERNAL DSCAL
*   .. Intrinsic Functions ..
INTRINSIC DBLE
*   .. Executable Statements ..
H = ONE/DBLE(N+1)
W(1) = TWO*V(1) - V(2)
DO 20 J = 2, N - 1
    W(J) = -V(J-1) + TWO*V(J) - V(J+1)
20 CONTINUE
J = N
W(J) = -V(J-1) + TWO*V(J)
CALL DSCAL(N,ONE/H,W,1)
RETURN
END

```

9.2 Program Data

F12FEF Example Program Data
 100 4 10 : Values for N NEV and NCV

9.3 Program Results

F12FEF Example Program Results

The 4 generalized Ritz values closest to 1.0000 are:

1	9.8704
2	39.4912
3	88.8909
4	158.1175
