NAG Fortran Library Routine Document

F12AEF

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 F12ADF need not be called. If, however, you wish to reset some or all of the settings please refer to Section 10 of the document for F12ADF for a detailed description of the specification of the optional parameters*.

1 Purpose

F12AEF can be used to return additional monitoring information during computation. It is in a suite of routines consisting of F12AEF, F12AAF, F12ABF, F12ACF and F12ADF.

2 Specification

```
SUBROUTINE F12AEF (NITER, NCONV, RITZR, RITZI, RZEST, ICOMM, COMM)
INTEGER              NITER, NCONV, ICOMM(*)
double precision     RITZR(*), RITZI(*), 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 non-symmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, real and non-symmetric problems.

On an intermediate exit from F12ABF with IREVCM = 4, F12AEF may be called to return monitoring information on the progress of the Arnoldi iterative process. The information returned by F12AEF 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.

F12AEF 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). F12AEF should not be called at any time other than immediately following an IREVCM = 4 return from F12ABF.

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: 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: RITZR($*$) – ***double precision*** array *Output*

Note: the dimension of the array RITZR must be at least NCV (see F12AAF).

On exit: the first NCONV locations of the array RITZR contain the real parts of the converged approximate eigenvalues.

4: RITZI($*$) – ***double precision*** array *Output*

Note: the dimension of the array RITZI must be at least NCV (see F12AAF).

On exit: the first NCONV locations of the array RITZI contain the imaginary parts of the converged approximate eigenvalues.

5: RZEST($*$) – ***double precision*** array *Output*

Note: the dimension of the array RZEST must be at least NCV (see F12AAF).

On exit: the first NCONV locations of the array RZEST contain the Ritz estimates (error bounds) on the converged approximate eigenvalues.

6: ICOMM($*$) – INTEGER array *Communication Array*

Note: the dimension of the array ICOMM must be at least $\max(1, \mathrm{LICOMM})$, where LICOMM is passed to the setup routine F12AAF (see F12AAF).

ICOMM must remain unchanged.

7: COMM($*$) – ***double precision*** array *Communication Array*

Note: the dimension of the array COMM must be at least $\max(1, \mathrm{LCOMM})$, where LCOMM is passed to the setup routine F12AAF (see F12AAF).

COMM must remain unchanged.

6 Error Indicators and Warnings

None.

7 Accuracy

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

8 Further Comments

None.

9 Example

This example solves $Ax = \lambda Bx$ in shifted-real mode, where A is the tridiagonal matrix with 2 on the diagonal, -2 on the subdiagonal and 3 on the superdiagonal. The matrix B is the tridiagonal matrix with 4 on the diagonal and 1 on the off-diagonals. The shift sigma, σ, is a complex number, and the operator used in the shifted-real iterative process is $\mathrm{OP} = \mathrm{real}\big((A - \sigma B)_{-1} B\big)$.

9.1 Program Text

```
*     F12AEF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          LICOMM, NIN, NOUT
      PARAMETER        (LICOMM=140,NIN=5,NOUT=6)
      INTEGER          MAXN, MAXNCV, LDV
      PARAMETER        (MAXN=256,MAXNCV=30,LDV=MAXN)
      INTEGER          LCOMM
      PARAMETER        (LCOMM=3*MAXN+3*MAXNCV*MAXNCV+6*MAXNCV+60)
      DOUBLE PRECISION ZERO
      PARAMETER        (ZERO=0.0D+0)
*     .. Local Scalars ..
      COMPLEX *16      C1, C2, C3
      DOUBLE PRECISION DENI, DENR, NUMI, NUMR, SIGMAI, SIGMAR
      INTEGER          IFAIL, IFAIL1, INFO, IREVCM, J, N, NCONV, NCV,
     +                 NEV, NITER, NSHIFT
      LOGICAL          FIRST
*     .. Local Arrays ..
      COMPLEX *16      CDD(MAXN), CDL(MAXN), CDU(MAXN), CDU2(MAXN),
     +                 CTEMP(MAXN)
      DOUBLE PRECISION AX(MAXN), COMM(LCOMM), D(MAXNCV,3), MX(MAXN),
     +                 RESID(MAXN), V(LDV,MAXNCV), X(MAXN)
      INTEGER          ICOMM(LICOMM), IPIV(MAXN)
*     .. External Functions ..
      DOUBLE PRECISION DDOT, DNRM2, F06BNF
      EXTERNAL         DDOT, DNRM2, F06BNF
*     .. External Subroutines ..
      EXTERNAL         AV, DCOPY, F12AAF, F12ABF, F12ACF, F12ADF,
     +                 F12AEF, MV, ZGTTRF, ZGTTRS
*     .. Intrinsic Functions ..
*
      INTRINSIC        DBLE, CMPLX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F12AEF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NEV, NCV, SIGMAR, SIGMAI
      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 F12AAF(N,NEV,NCV,ICOMM,LICOMM,COMM,LCOMM,IFAIL)
*        Set the mode.
         CALL F12ADF('SHIFTED REAL',ICOMM,COMM,IFAIL)
*        Set problem type
         CALL F12ADF('GENERALIZED',ICOMM,COMM,IFAIL)
*        Solve A*x = lambda*B*x in shift-invert mode.
*        The shift, sigma, is a complex number (sigmar, sigmai).
*        OP = Real_Part{inv[A-(SIGMAR,SIGMAI)*M]*M and  B = M.
         C1 = CMPLX(-2.0D+0-SIGMAR,-SIGMAI,KIND=KIND(DENI))
         C2 = CMPLX(2.0D+0-4.0D+0*SIGMAR,-4.0D+0*SIGMAI,KIND=KIND(DENI))
         C3 = CMPLX(3.0D+0-SIGMAR,-SIGMAI,KIND=KIND(DENI))
*
         DO 20 J = 1, N - 1
            CDL(J) = C1
            CDD(J) = C2
            CDU(J) = C3
   20    CONTINUE
         CDD(N) = C2
*
         CALL ZGTTRF(N,CDL,CDD,CDU,CDU2,IPIV,INFO)
*
         IREVCM = 0
         IFAIL = -1
   40    CONTINUE
         CALL F12ABF(IREVCM,RESID,V,LDV,X,MX,NSHIFT,COMM,ICOMM,IFAIL)
```

```
         IF (IREVCM.NE.5) THEN
            IF (IREVCM.EQ.-1) THEN
*              Perform  x <--- OP*x = inv[A-SIGMA*M]*M*x
               CALL MV(N,X)
               DO 60 J = 1, N
                  CTEMP(J) = CMPLX(X(J),KIND=KIND(DENI))
   60          CONTINUE
               CALL ZGTTRS('N',N,1,CDL,CDD,CDU,CDU2,IPIV,CTEMP,N,INFO)
               DO 80 J = 1, N
                  X(J) = DBLE(CTEMP(J))
   80          CONTINUE
            ELSE IF (IREVCM.EQ.1) THEN
*              Perform  x <--- OP*x = inv[A-SIGMA*M]*M*x,
*              M*X stored in MX.
               DO 100 J = 1, N
                  CTEMP(J) = CMPLX(MX(J),KIND=KIND(DENI))
  100          CONTINUE
               CALL ZGTTRS('N',N,1,CDL,CDD,CDU,CDU2,IPIV,CTEMP,N,INFO)
               DO 120 J = 1, N
                  X(J) = DBLE(CTEMP(J))
  120          CONTINUE
            ELSE IF (IREVCM.EQ.2) THEN
*              Perform  y <--- M*x
               CALL MV(N,X)
            ELSE IF (IREVCM.EQ.4) THEN
*              Output monitoring information
               CALL F12AEF(NITER,NCONV,D,D(1,2),D(1,3),ICOMM,COMM)
               WRITE (6,99998) NITER, NCONV, DNRM2(NEV,D(1,3),1)
            END IF
            GO TO 40
         END IF
         IF (IFAIL.EQ.0) THEN
*           Post-Process using F12ACF to compute eigenvalues/vectors.
            IFAIL1 = 0
            CALL F12ACF(NCONV,D,D(1,2),V,LDV,SIGMAR,SIGMAI,RESID,V,LDV,
     +                  COMM,ICOMM,IFAIL1)
            FIRST = .TRUE.
            DO 140 J = 1, NCONV
*              Use Rayleigh Quotient to recover eigenvalues of the original
*              problem.
               IF (D(J,2).EQ.ZERO) THEN
*                 Ritz value is real.
                  CALL AV(N,V(1,J),AX)
                  NUMR = DDOT(N,V(1,J),1,AX,1)
                  CALL DCOPY(N,V(1,J),1,MX,1)
                  CALL MV(N,MX)
                  DENR = DDOT(N,V(1,J),1,MX,1)
                  D(J,1) = NUMR/DENR
               ELSE IF (FIRST) THEN
*                 Ritz value is complex.
*                 Compute x'(Ax)
                  CALL AV(N,V(1,J),AX)
                  NUMR = DDOT(N,V(1,J),1,AX,1)
                  NUMI = DDOT(N,V(1,J+1),1,AX,1)
                  CALL AV(N,V(1,J+1),AX)
                  NUMR = NUMR + DDOT(N,V(1,J+1),1,AX,1)
                  NUMI = -NUMI + DDOT(N,V(1,J),1,AX,1)
*                 Compute x'(Mx)
                  CALL DCOPY(N,V(1,J),1,MX,1)
                  CALL MV(N,MX)
                  DENR = DDOT(N,V(1,J),1,MX,1)
                  DENI = DDOT(N,V(1,J+1),1,MX,1)
                  CALL DCOPY(N,V(1,J+1),1,MX,1)
                  CALL MV(N,MX)
                  DENR = DENR + DDOT(N,V(1,J+1),1,MX,1)
                  DENI = -DENI + DDOT(N,V(1,J),1,MX,1)
*                 d=x'(Ax)/x'(Mx)
                  D(J,1) = (NUMR*DENR+NUMI*DENI)/F06BNF(DENR,DENI)
                  D(J,2) = (NUMI*DENR-NUMR*DENI)/F06BNF(DENR,DENI)
                  FIRST = .FALSE.
               ELSE
```

```
*                 Second of complex conjugate pair.
                  D(J,1) = D(J-1,1)
                  D(J,2) = -D(J-1,2)
                  FIRST = .TRUE.
               END IF
  140       CONTINUE
*           Print computed eigenvalues.
            WRITE (NOUT,99996) NCONV, SIGMAR, SIGMAI
            DO 160 J = 1, NCONV
               WRITE (NOUT,99995) J, D(J,1), D(J,2)
  160       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 =',E12.4)
99997 FORMAT (1X,' NAG Routine F12ABF Returned with IFAIL = ',I6)
99996 FORMAT (1X,/' The ',I4,' generalized Ritz values closest to ','(',
     +       F8.4,', ',F8.4,')',' are:',/)
99995 FORMAT (1X,I8,5X,'(',F7.4,',',F7.4,')')
      END
*
      SUBROUTINE MV(N,V)
*     Compute the in-place matrix vector multiplication X<---M*X,
*     where M is mass matrix formed by using piecewise linear elements
*     on [0,1].
*
*     .. Parameters ..
      DOUBLE PRECISION FOUR
      PARAMETER        (FOUR=4.0D+0)
*     .. Scalar Arguments ..
      INTEGER          N
*     .. Array Arguments ..
      DOUBLE PRECISION V(N)
*     .. Local Scalars ..
      DOUBLE PRECISION VM1, VV
      INTEGER          J
*     .. Executable Statements ..
      VM1 = V(1)
      V(1) = FOUR*V(1) + V(2)
      DO 20 J = 2, N - 1
         VV = V(J)
         V(J) = VM1 + FOUR*VV + V(J+1)
         VM1 = VV
   20 CONTINUE
      V(N) = VM1 + FOUR*V(N)
      RETURN
      END
*
      SUBROUTINE AV(N,V,W)
*     .. Parameters ..
      DOUBLE PRECISION THREE, TWO
      PARAMETER        (THREE=3.0D+0,TWO=2.0D+0)
*     .. Scalar Arguments ..
      INTEGER          N
*     .. Array Arguments ..
      DOUBLE PRECISION V(N), W(N)
*     .. Local Scalars ..
      INTEGER          J
*     .. Executable Statements ..
      W(1) = TWO*V(1) + THREE*V(2)
      DO 20 J = 2, N - 1
         W(J) = -TWO*V(J-1) + TWO*V(J) + THREE*V(J+1)
   20 CONTINUE
      W(N) = -TWO*V(N-1) + TWO*V(N)
      RETURN
      END
```

9.2 Program Data

```
F12AEF Example Program Data
 100 4 20 4.0D-1 6.0D-1 : Values for NX NEV NCV SIGMAR SIGMAI
```

9.3 Program Results

```
 F12AEF Example Program Results

 Iteration   1, No. converged =   0, norm of estimates =  0.1052E+00
 Iteration   2, No. converged =   0, norm of estimates =  0.1188E-02
 Iteration   3, No. converged =   0, norm of estimates =  0.1389E-05
 Iteration   4, No. converged =   0, norm of estimates =  0.3939E-08
 Iteration   5, No. converged =   0, norm of estimates =  0.1158E-10
 Iteration   6, No. converged =   0, norm of estimates =  0.5222E-13

 The    4 generalized Ritz values closest to (  0.4000,   0.6000) are:

        1     ( 0.5000,-0.5958)
        2     ( 0.5000, 0.5958)
        3     ( 0.5000,-0.6331)
        4     ( 0.5000, 0.6331)
```