NAG Fortran Library Routine Document

F12ASF

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

F12ASF can be used to return additional monitoring information during computation. It is in a suite of routines consisting of F12ASF, F12ANF, F12APF, F12AQF and F12ARF.

2 Specification

```
SUBROUTINE F12ASF (NITER, NCONV, RITZ, RZEST, ICOMM, COMM)
INTEGER            NITER, NCONV, ICOMM(134)
complex*16         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 complex eigenvalue problem $Ax = \lambda x$, or of a generalized complex eigenvalue problem $Ax = \lambda Bx$ of order n, where n is large and the coefficient matrices A and B are sparse and complex. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense complex problems.

On an intermediate exit from F12APF with $\mathrm{IREVCM} = 4$, F12ASF may be called to return monitoring information on the progress of the Arnoldi iterative process. The information returned by F12ASF is:

the number of the current Arnoldi iteration;

the number of converged eigenvalues at this point;

the converged eigenvalues;

the error bounds on the converged eigenvalues.

F12ASF 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). F12ASF should not be called at any time other than immediately following a $\mathrm{IREVCM} = 4$ return from F12APF.

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: RITZ($*$) – ***complex*16*** 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 converged approximate eigenvalues.

4: RZEST($*$) – ***complex*16*** 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 complex Ritz esimates on the converged approximate eigenvalues.

5: ICOMM(134) – INTEGER array *Communication Array*

ICOMM, on initial entry, must remain unchanged between calls to F12APF.

6: COMM($*$) – ***complex*16*** array *Communication Array*

COMM, on initial entry, must remain unchanged between calls to F12APF.

6 Error Indicators and Warnings

None.

7 Accuracy

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

8 Further Comments

None.

9 Example

The example solves $Ax = \lambda Bx$ in shifted-inverse mode, where A and B are obtained from the standard central difference discretization of the one-dimensional convection-diffusion operator $\frac{d^2u}{dx^2} + \rho\frac{du}{dx}$ on $[0, 1]$, with zero Dirichlet boundary conditions. The shift, σ, is a complex number, and the operator used in the shifted-inverse iterative process is OP $= \mathrm{inv}(A - \sigma B) \times B$.

9.1 Program Text

```
*     F12ASF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          IMON, LICOMM, NERR, NIN, NOUT
      PARAMETER        (IMON=1,LICOMM=140,NERR=6,NIN=5,NOUT=6)
      INTEGER          MAXN, MAXNCV, LDV
      PARAMETER        (MAXN=256,MAXNCV=30,LDV=MAXN)
      INTEGER          LCOMM
      PARAMETER        (LCOMM=3*MAXN+3*MAXNCV*MAXNCV+5*MAXNCV+60)
      COMPLEX *16      ONE, TWO, FOUR, SIX
      PARAMETER        (ONE=(1.0D+0,0.0D+0),TWO=(2.0D+0,0.0D+0),
     +                 FOUR=(4.0D+0,0.0D+0),SIX=(6.0D+0,0.0D+0))
*     .. Local Scalars ..
      COMPLEX *16      H, RHO, S, S1, S2, S3, SIGMA
      INTEGER          IFAIL, IFAIL1, INFO, IREVCM, J, N, NCONV, NCV,
     +                 NEV, NITER, NSHIFT, NX
```

```
*     .. Local Arrays ..
      COMPLEX *16       AX(MAXN), COMM(LCOMM), D(MAXNCV,2), DD(MAXN),
     +                  DL(MAXN), DU(MAXN), DU2(MAXN), MX(MAXN),
     +                  RESID(MAXN), V(LDV,MAXNCV), X(MAXN)
      INTEGER           ICOMM(LICOMM), IPIV(MAXN)
*     .. External Functions ..
      DOUBLE PRECISION DZNRM2
      EXTERNAL          DZNRM2
*     .. External Subroutines ..
      EXTERNAL          F12ANF, F12APF, F12AQF, F12ARF, F12ASF, MV,
     +                  ZCOPY, ZGTTRF, ZGTTRS
*     .. Intrinsic Functions ..
      INTRINSIC         DCMPLX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F12ASF 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 F12ANF(N,NEV,NCV,ICOMM,LICOMM,COMM,LCOMM,IFAIL)
*     Set the mode.
         CALL F12ARF('SHIFTED INVERSE',ICOMM,COMM,IFAIL)
*     Set problem type.
         CALL F12ARF('GENERALIZED',ICOMM,COMM,IFAIL)
*      SIGMA = ONE
         SIGMA = (5.0D+3,0.0D0)
         RHO = (1.0D+1,0.0D0)
         H = ONE/DCMPLX(N+1)
         S = RHO/TWO
         S1 = -ONE/H - S - SIGMA*H/SIX
         S2 = TWO/H - FOUR*SIGMA*H/SIX
         S3 = -ONE/H + S - SIGMA*H/SIX
*
         DO 20 J = 1, N - 1
            DL(J) = S1
            DD(J) = S2
            DU(J) = S3
   20    CONTINUE
         DD(N) = S2
*
         CALL ZGTTRF(N,DL,DD,DU,DU2,IPIV,INFO)
         IF (INFO.NE.0) THEN
            WRITE (NERR,99998) INFO
            GO TO 80
         END IF
*
         IREVCM = 0
         IFAIL = -1
   40    CONTINUE
         CALL F12APF(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(NX,X,AX)
               CALL ZCOPY(N,AX,1,X,1)
               CALL ZGTTRS('N',N,1,DL,DD,DU,DU2,IPIV,X,N,INFO)
               IF (INFO.NE.0) THEN
                  WRITE (NERR,99997) INFO
                  GO TO 80
               END IF
            ELSE IF (IREVCM.EQ.1) THEN
*           Perform  x <--- OP*x = inv[A-SIGMA*M]*M*x,
*           MX stored in COMM from location IPNTR(3)
               CALL ZGTTRS('N',N,1,DL,DD,DU,DU2,IPIV,MX,N,INFO)
```

```
               CALL ZCOPY(N,MX,1,X,1)
               IF (INFO.NE.0) THEN
                  WRITE (NERR,99997) INFO
                  GO TO 80
               END IF
            ELSE IF (IREVCM.EQ.2) THEN
*           Perform  y <--- M*x
               CALL MV(NX,X,AX)
               CALL ZCOPY(N,AX,1,X,1)
            ELSE IF (IREVCM.EQ.4 .AND. IMON.NE.0) THEN
*           Output monitoring information
               CALL F12ASF(NITER,NCONV,D,D(1,2),ICOMM,COMM)
               WRITE (6,99996) NITER, NCONV, DZNRM2(NEV,D(1,2),1)
            END IF
            GO TO 40
         END IF
         IF (IFAIL.EQ.0) THEN
*        Post-Process using F12AQF to compute eigenvalues/vectors.
            IFAIL1 = 0
            CALL F12AQF(NCONV,D,V,LDV,SIGMA,RESID,V,LDV,COMM,ICOMM,
     +                  IFAIL1)
            WRITE (NOUT,99994) NCONV, SIGMA
            DO 60 J = 1, NCONV
               WRITE (NOUT,99993) J, D(J,1)
   60       CONTINUE
         ELSE
            WRITE (NOUT,99995) IFAIL
         END IF
   80    CONTINUE
      END IF
      STOP
*
99999 FORMAT (1X,A,I5)
99998 FORMAT (1X,'** Error status returned by ZGTTRF, INFO =',I12)
99997 FORMAT (1X,'** Error status returned by ZGTTRS, INFO =',I12)
99996 FORMAT (1X,'Iteration',1X,I3,', No. converged =',1X,I3,', norm o',
     +       'f estimates =',E12.4)
99995 FORMAT (1X,' NAG Routine F12APF Returned with IFAIL = ',I6)
99994 FORMAT (1X,/' The ',I4,' generalized Ritz values closest to',' (',
     +       F8.3,',',F8.3,') are:',/)
99993 FORMAT (1X,I8,5X,'( ',F10.4,' , ',F10.4,' )')
      END
*
      SUBROUTINE MV(NX,V,W)
*     Compute the out-of--place matrix vector multiplication Y<---M*X,
*     where M is mass matrix formed by using piecewise linear elements
*     on [0,1].
*
*     .. Parameters ..
      COMPLEX *16    ONE, FOUR, SIX
      PARAMETER      (ONE=(1.0D+0,0.0D+0),FOUR=(4.0D+0,0.0D+0),
     +               SIX=(6.0D+0,0.0D+0))
*     .. Scalar Arguments ..
      INTEGER        NX
*     .. Array Arguments ..
      COMPLEX *16    V(NX*NX), W(NX*NX)
*     .. Local Scalars ..
      COMPLEX *16    H
      INTEGER        J, N
*     .. External Subroutines ..
      EXTERNAL       ZSCAL
*     .. Intrinsic Functions ..
      INTRINSIC      DCMPLX
*     .. Executable Statements ..
      N = NX*NX
      W(1) = (FOUR*V(1)+V(2))/SIX
      DO 20 J = 2, N - 1
         W(J) = (V(J-1)+FOUR*V(J)+V(J+1))/SIX
   20 CONTINUE
      W(N) = (V(N-1)+FOUR*V(N))/SIX
*
```

```
      H = ONE/DCMPLX(N+1)
      CALL ZSCAL(N,H,W,1)
      RETURN
      END
```

9.2 Program Data

```
F12ASF Example Program Data
 16 4 10 : Vaues for NX NEV and NCV
```

9.3 Program Results

```
 F12ASF Example Program Results

 Iteration   1, No. converged =   0, norm of estimates =  0.7246E-06
 Iteration   2, No. converged =   0, norm of estimates =  0.2545E-08
 Iteration   3, No. converged =   2, norm of estimates =  0.8628E-11
 Iteration   4, No. converged =   2, norm of estimates =  0.2611E-13
 Iteration   5, No. converged =   2, norm of estimates =  0.1989E-15
 Iteration   6, No. converged =   3, norm of estimates =  0.2204E-17

 The    4 generalized Ritz values closest to (5000.000,   0.000) are:

        1     (  4829.8497 ,    -0.0000 )
        2     (  5279.5223 ,     0.0000 )
        3     (  4400.6310 ,    -0.0000 )
        4     (  5749.7160 ,     0.0000 )
```