NAG Fortran Library Routine Document

F12AQF

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

F12AQF is a post-processing routine in a suite of routines consisting of F12AQF, F12ANF, F12APF, F12ARF and F12ASF, that must be called following a final exit from F12AQF.

2 Specification

```
SUBROUTINE F12AQF (NCONV, D, Z, LDZ, SIGMA, RESID, V, LDV, COMM, ICOMM,
1                  IFAIL)
INTEGER            NCONV, LDZ, LDV, ICOMM(*), IFAIL
complex*16         D(*), Z(LDZ,*), SIGMA, 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 sparse, complex and nonsymmetric. The suite can also be used to find selected eigenvalues/eigenvectors of smaller scale dense, complex and nonsymmetric problems.

Following a call to F12APF, F12AQF 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 complex nonsymmetric matrices. There is negligible additional cost to obtain eigenvectors; an orthonormal basis is always computed, but there is an additional storage cost if both are requested.

F12AQF is based on the routine **zneupd** from the ARPACK package, which uses the Implicitly Restarted Arnoldi 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). An evaluation of software for computing eigenvalues of sparse nonsymmetric matrices is provided in Lehoucq and Scott (1996). This suite of routines offers the same functionality as the ARPACK software for complex nonsymmetric problems, but the interface design is quite different in order to make the option setting clearer to the user and to simplify some of the interfaces.

F12AQF, is a post-processing routine that must be called following a successful final exit from F12APF. F12AQF uses data returned from F12APF and options, set either by default or explicitly by calling F12ARF, 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 Parameters

1: NCONV – INTEGER *Output*

On exit: the number of converged eigenvalues as found by F12ARF.

2: D(∗) – ***complex*16*** array *Output*

Note: the dimension of the array D must be at least NEV.

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

3: Z(LDZ,∗) – ***complex*16*** array *Output*

Note: the second dimension of the array Z must be at least NEV if the default option **Vectors** = Ritz has been selected and at least 1 if the option **Vectors** = None or Schur has been selected.

On exit: if the default option **Vectors** = Ritz has been selected then Z contains the final set of eigenvectors corresponding to the eigenvalues held in D. The complex eigenvector associated with an eigenvalue is stored in the corresponding column of Z.

4: LDZ – INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which F12AQF is called.

Constraints:

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

5: SIGMA – ***complex*16*** *Input*

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

6: RESID(∗) – ***complex*16*** array *Input*

Note: the dimension of the array RESID must be at least N.

On entry: RESID must not be modified following a call to F12APF since it contains data required by F12AQF.

7: V(LDV,∗) – ***complex*16*** array *Input/Output*

Note: the second dimension of the array V must be at least $\max(1, \text{NCV})$.

On entry: the NCV columns of V contain the Arnoldi basis vectors for OP as constructed by F12APF.

On exit: if the option **Vectors** = Schur or Ritz has been set a separate array Z has been passed, then the first NCONV columns of V will contain approximate Schur vectors that span the desired invariant subspace.

8: LDV – INTEGER *Input*

On entry: the first dimension of the array V as declared in the (sub)program from which F12AQF is called.

Constraint: LDV $\geq$ N.

9: COMM(∗) – ***complex*16*** array *Communication Array*

COMM must remain unchanged from the prior call to F12APF.

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

ICOMM must remain unchanged from the prior call to F12APF.

11: 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: 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, LDZ < max(1, N) or LDZ < 1 when no vectors are required.

IFAIL = 2

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

IFAIL = 3

The number of eigenvalues found to sufficient accuracy prior to calling F12AQF, as communicated through the parameter ICOMM, is zero.

IFAIL = 4

The number of converged eigenvalues as calculated by F12APF differ from the value passed to it through the parameter ICOMM.

IFAIL = 5

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

IFAIL = 6

Unexpected error: the computed Schur form could not be reordered by an internal call. Please contact NAG.

IFAIL = 7

Unexpected error in internal call while calculating eigenvectors. Please contact NAG.

IFAIL = 8

Either the solver routine F12APF has not been called prior to the call of this routine or a communication array has become corrupted.

IFAIL = 9

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

IFAIL = 10

An unexpected error has occurred. Please contact NAG.

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 Further Comments

None.

9 Example

The example solves $Ax = \lambda Bx$ in regular-invert mode, where A and B are derived 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.

9.1 Program Text

```
*     F12AQF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          IMON, LICOMM, NERR, NIN, NOUT
      PARAMETER        (IMON=0,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
      PARAMETER        (ONE=(1.0D+0,0.0D+0))
*     .. Local Scalars ..
      COMPLEX *16      H, SIGMA
      INTEGER          IFAIL, IFAIL1, INFO, IREVCM, J, N, NCONV, NCV,
     +                 NEV, NITER, NSHIFT, NX
*     .. Local Arrays ..
      COMPLEX *16      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         AV, F12ANF, F12APF, F12AQF, F12ARF, F12ASF, MV,
     +                 ZGTTRF, ZGTTRS
*     .. Intrinsic Functions ..
*
      INTRINSIC        DCMPLX
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F12AQF 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('REGULAR INVERSE',ICOMM,COMM,IFAIL)
*     Set problem type.
         CALL F12ARF('GENERALIZED',ICOMM,COMM,IFAIL)
*     Use pointers to Workspace than interfacing through the array X.
         CALL F12ARF('POINTERS=YES',ICOMM,COMM,IFAIL)
         H = ONE/DCMPLX(N+1)
*
         DO 20 J = 1, N - 1
            DL(J) = H
            DD(J) = (4.0D+0,0.0D+0)*H
            DU(J) = H
   20    CONTINUE
         DD(N) = (4.0D+0,0.0D+0)*H
*
         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 .OR. IREVCM.EQ.1) THEN
*           Perform  y <--- OP*x = inv[M]*A*x      |
               CALL AV(NX,COMM(ICOMM(1)),COMM(ICOMM(2)))
               CALL ZGTTRS('N',N,1,DL,DD,DU,DU2,IPIV,COMM(ICOMM(2)),N,
     +                     INFO)
               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,COMM(ICOMM(1)),COMM(ICOMM(2)))
            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
            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 =',E16.8)
99995 FORMAT (1X,' NAG Routine F12APF Returned with IFAIL = ',I6)
99994 FORMAT (1X,/' The ',I4,' Ritz values of largest magnitude are:',/)
```

```
99993 FORMAT (1X,I8,5X,'( ',F12.4,' , ',F12.4,' )')
      END
*
      SUBROUTINE AV(NX,V,W)
*     .. Parameters ..
      COMPLEX *16   ONE, TWO, RHO
      PARAMETER     (ONE=(1.0D+0,0.0D+0),TWO=(2.0D+0,0.0D+0),
     +              RHO=(1.0D+1,0.0D+0))
*     .. Scalar Arguments ..
      INTEGER       NX
*     .. Array Arguments ..
      COMPLEX *16   V(NX*NX), W(NX*NX)
*     .. Local Scalars ..
      COMPLEX *16   DD, DL, DU, H, S
      INTEGER       J, N
*     .. Intrinsic Functions ..
      INTRINSIC     DCMPLX
*     .. Executable Statements ..
      N = NX*NX
      H = ONE/DCMPLX(N+1)
      S = RHO/TWO
      DD = TWO/H
      DL = -ONE/H - S
      DU = -ONE/H + S
      W(1) = DD*V(1) + DU*V(2)
      DO 20 J = 2, N - 1
         W(J) = DL*V(J-1) + DD*V(J) + DU*V(J+1)
   20 CONTINUE
      W(N) = DL*V(N-1) + DD*V(N)
      RETURN
      END
*
      SUBROUTINE MV(NX,V,W)
*     .. Parameters ..
      COMPLEX *16   ONE, FOUR
      PARAMETER     (ONE=(1.0D+0,0.0D+0),FOUR=(4.0D+0,00D+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) + ONE*V(2)
      DO 20 J = 2, N - 1
         W(J) = ONE*V(J-1) + FOUR*V(J) + ONE*V(J+1)
   20 CONTINUE
      W(N) = ONE*V(N-1) + FOUR*V(N)
      H = ONE/DCMPLX(N+1)
      CALL ZSCAL(N,H,W,1)
      RETURN
      END
```

9.2 Program Data

```
F12AQF Example Program Data
 10 4 20 : Vaues for NX NEV and NCV
```

9.3 Program Results

```
 F12AQF Example Program Results

 The    4 Ritz values of largest magnitude are:

        1     (   20383.0384 ,      -0.0000 )
        2     (   20338.7563 ,      -0.0000 )
        3     (   20265.2844 ,      -0.0000 )
        4     (   20163.1142 ,       0.0000 )
```