F02BJF – NAG Fortran Library Routine Document

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

F02BJF calculates all the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax = \lambda Bx$ where A and B are real, square matrices, using the QZ algorithm.

2 Specification

```
 SUBROUTINE F02BJF(N, A, IA, B, IB, EPS1, ALFR, ALFI, BETA, MATV,
1                  V, IV, ITER, IFAIL)
 INTEGER           N, IA, IB, IV, ITER(N), IFAIL
 real              A(IA,N), B(IB,N), EPS1, ALFR(N), ALFI(N),
1                  BETA(N), V(IV,N)
 LOGICAL           MATV
```

3 Description

All the eigenvalues and, if required, all the eigenvectors of the generalized eigenproblem $Ax = \lambda Bx$ where A and B are real, square matrices, are determined using the QZ algorithm. The QZ algorithm consists of 4 stages:

(a) A is reduced to upper Hessenberg form and at the same time B is reduced to upper triangular form.
(b) A is further reduced to quasi-triangular form while the triangular form of B is maintained.
(c) The quasi-triangular form of A is reduced to triangular form and the eigenvalues extracted. This routine does not actually produce the eigenvalues λ_j, but instead returns α_j and β_j such that

$$\lambda_j = \alpha_j/\beta_j, \quad j = 1, 2, \ldots, n.$$

The division by β_j becomes the responsibility of the user's program, since β_j may be zero indicating an infinite eigenvalue. Pairs of complex eigenvalues occur with α_j/β_j and α_{j+1}/β_{j+1} complex conjugates, even though α_j and α_{j+1} are not conjugate.
(d) If the eigenvectors are required (MATV = .TRUE.), they are obtained from the triangular matrices and then transformed back into the original co-ordinate system.

4 References

[1] Moler C B and Stewart G W (1973) An algorithm for generalized matrix eigenproblems *SIAM J. Numer. Anal.* **10** 241–256

[2] Ward R C (1975) The combination shift QZ algorithm *SIAM J. Numer. Anal.* **12** 835–853

[3] Wilkinson J H (1979) Kronecker's canonical form and the QZ algorithm *Linear Algebra Appl.* **28** 285–303

5 Parameters

1: N — INTEGER *Input*

On entry: n, the order of the matrices A and B.

2: A(IA,N) — ***real*** array *Input/Output*

On entry: the n by n matrix A.

On exit: the array is overwritten.

3: IA — INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F02BJF is called.

Constraint: IA $\geq$ N.

4: B(IB,N) — ***real*** array *Input/Output*

On entry: the n by n matrix B.

On exit: the array is overwritten.

5: IB — INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F02BJF is called.

Constraint: IB $\geq$ N.

6: EPS1 — ***real*** *Input*

On entry: the tolerance used to determine negligible elements. If EPS1 $>$ 0.0, an element will be considered negligible if it is less than EPS1 times the norm of its matrix. If EPS1 $\leq$ 0.0, ***machine precision*** is used in place of EPS1. A positive value of EPS1 may result in faster execution but less accurate results.

7: ALFR(N) — ***real*** array *Output*
8: ALFI(N) — ***real*** array *Output*

On exit: the real and imaginary parts of α_j, for $j = 1, 2, \ldots, n$.

9: BETA(N) — ***real*** array *Output*

On exit: β_j, for $j = 1, 2, \ldots, n$.

10: MATV — LOGICAL *Input*

On entry: MATV must be set .TRUE. if the eigenvectors are required, otherwise .FALSE..

11: V(IV,N) — ***real*** array *Output*

On exit: if MATV = .TRUE., then

(i) if the jth eigenvalue is real, the jth column of V contains its eigenvector;
(ii) if the jth and $(j+1)$th eigenvalues form a complex pair, the jth and $(j+1)$th columns of V contain the real and imaginary parts of the eigenvector associated with the first eigenvalue of the pair. The conjugate of this vector is the eigenvector for the conjugate eigenvalue.

Each eigenvector is normalised so that the component of largest modulus is real and the sum of squares of the moduli equal one.

If MATV = .FALSE., V is not used.

12: IV — INTEGER *Input*

On entry: the first dimension of the array V as declared in the (sub)program from which F02BJF is called.

Constraint: IV $\geq$ N.

13: ITER(N) — INTEGER array *Output*

On exit: ITER(j) contains the number of iterations needed to obtain the jth eigenvalue. Note that the eigenvalues are obtained in reverse order, starting with the nth.

14: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors detected by the routine:

IFAIL $= i$

More than $30 \times$ N iterations are required to determine all the diagonal 1 by 1 or 2 by 2 blocks of the quasi-triangular form in the second step of the QZ algorithm. IFAIL is set to the index i of the eigenvalue at which this failure occurs. If the soft failure option is used, α_j and β_j are correct for $j = i+1, i+2, \ldots, n$, but V does not contain any correct eigenvectors.

7 Accuracy

The computed eigenvalues are always exact for a problem $(A+E)x = \lambda(B+F)x$ where $\|E\|/\|A\|$ and $\|F\|/\|B\|$ are both of the order of $\max(\text{EPS1}, \epsilon)$, EPS1 being defined as in Section 5 and ϵ being the ***machine precision***.

Note. Interpretation of results obtained with the QZ algorithm often requires a clear understanding of the effects of small changes in the original data. These effects are reviewed in Wilkinson [3], in relation to the significance of small values of α_j and β_j. It should be noted that if α_j and β_j are **both** small for any j, it may be that no reliance can be placed on **any** of the computed eigenvalues $\lambda_i = \alpha_i/\beta_i$. The user is recommended to study [3] and, if in difficulty, to seek expert advice on determining the sensitivity of the eigenvalues to perturbations in the data.

8 Further Comments

The time taken by the routine is approximately proportional to n^3 and also depends on the value chosen for parameter EPS1.

9 Example

To find all the eigenvalues and eigenvectors of $Ax = \lambda Bx$ where

$$A = \begin{pmatrix} 3.9 & 12.5 & -34.5 & -0.5 \\ 4.3 & 21.5 & -47.5 & 7.5 \\ 4.3 & 21.5 & -43.5 & 3.5 \\ 4.4 & 26.0 & -46.0 & 6.0 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 1 & 2 & -3 & 1 \\ 1 & 3 & -5 & 4 \\ 1 & 3 & -4 & 3 \\ 1 & 3 & -4 & 4 \end{pmatrix}.$$

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     F02BJF Example Program Text
*     Mark 14 Revised.  NAG Copyright 1989.
*     .. Parameters ..
      INTEGER          NMAX, IA, IB, IZ
      PARAMETER        (NMAX=8,IA=NMAX,IB=NMAX,IZ=NMAX)
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
*     .. Local Scalars ..
      real             EPS1
      INTEGER          I, IFAIL, IP, J, K, N
      LOGICAL          MATZ
*     .. Local Arrays ..
      real             A(IA,NMAX), ALFI(NMAX), ALFR(NMAX), B(IB,NMAX),
     +                 BETA(NMAX), Z(IZ,NMAX)
      INTEGER          ITER(NMAX)
*     .. External Functions ..
      real             X02AJF
      EXTERNAL         X02AJF
*     .. External Subroutines ..
      EXTERNAL         F02BJF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F02BJF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.GT.0 .AND. N.LE.NMAX) THEN
         READ (NIN,*) ((A(I,J),J=1,N),I=1,N)
         READ (NIN,*) ((B(I,J),J=1,N),I=1,N)
         MATZ = .TRUE.
         EPS1 = X02AJF()
         IFAIL = 1
*
         CALL F02BJF(N,A,IA,B,IB,EPS1,ALFR,ALFI,BETA,MATZ,Z,IZ,ITER,
     +               IFAIL)
*
         IF (IFAIL.NE.0) THEN
            WRITE (NOUT,*)
            WRITE (NOUT,99999) 'Error in F02BJF. IFAIL =', IFAIL
         ELSE
            IP = 0
            DO 40 I = 1, N
               WRITE (NOUT,*)
               WRITE (NOUT,99999) 'Eigensolution', I
               WRITE (NOUT,*)
               WRITE (NOUT,99998) 'ALFR(', I, ')', ALFR(I), '   ALFI(',
     +           I, ')', ALFI(I), '  BETA(', I, ')', BETA(I)
               IF (BETA(I).EQ.0.0e0) THEN
                  WRITE (NOUT,*) 'LAMBDA is infinite'
               ELSE
                  IF (ALFI(I).EQ.0.0e0) THEN
                     WRITE (NOUT,*)
                     WRITE (NOUT,99997) 'LAMBDA    ', ALFR(I)/BETA(I)
                     WRITE (NOUT,*)
                     WRITE (NOUT,*) 'Eigenvector'
```

```
                  WRITE (NOUT,99996) (Z(J,I),J=1,N)
               ELSE
                  WRITE (NOUT,*)
                  WRITE (NOUT,99997) 'LAMBDA    ', ALFR(I)/BETA(I),
     +              ALFI(I)/BETA(I)
                  WRITE (NOUT,*)
                  WRITE (NOUT,*) 'Eigenvector'
                  K = (-1)**(IP+2)
                  DO 20 J = 1, N
                     WRITE (NOUT,99995) Z(J,I-IP), K*Z(J,I-IP+1)
   20             CONTINUE
                  IP = 1 - IP
               END IF
            END IF
   40    CONTINUE
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Number of iterations (machine-dependent)'
         WRITE (NOUT,99994) (ITER(I),I=1,N)
      END IF
      ELSE
         WRITE (NOUT,*)
         WRITE (NOUT,99999) 'N is out of range: N = ', N
      END IF
      STOP
*
99999 FORMAT (1X,A,I4)
99998 FORMAT (1X,A,I1,A,F7.3,A,I1,A,F7.3,A,I1,A,F7.3)
99997 FORMAT (1X,A,2F7.3)
99996 FORMAT (1X,F7.3)
99995 FORMAT (1X,2F7.3)
99994 FORMAT (1X,8I4)
      END
```

9.2 Program Data

```
F02BJF Example Program Data
  4
   3.9  12.5 -34.5  -0.5
   4.3  21.5 -47.5   7.5
   4.3  21.5 -43.5   3.5
   4.4  26.0 -46.0   6.0
   1.0   2.0  -3.0   1.0
   1.0   3.0  -5.0   4.0
   1.0   3.0  -4.0   3.0
   1.0   3.0  -4.0   4.0
```

9.3 Program Results

```
 F02BJF Example Program Results

 Eigensolution   1

 ALFR(1)  3.801   ALFI(1)  0.000  BETA(1)  1.900

 LAMBDA      2.000

 Eigenvector
```

```
  0.996
  0.006
  0.063
  0.063

Eigensolution   2

ALFR(2)  1.563   ALFI(2)  2.084  BETA(2)  0.521

LAMBDA      3.000  4.000

Eigenvector
  0.945  0.000
  0.189  0.000
  0.113 -0.151
  0.113 -0.151

Eigensolution   3

ALFR(3)  3.030   ALFI(3) -4.040  BETA(3)  1.010

LAMBDA      3.000 -4.000

Eigenvector
  0.945  0.000
  0.189  0.000
  0.113  0.151
  0.113  0.151

Eigensolution   4

ALFR(4)  4.000   ALFI(4)  0.000  BETA(4)  1.000

LAMBDA      4.000

Eigenvector
  0.988
  0.011
 -0.033
  0.154

Number of iterations (machine-dependent)
   0   0   5   0
```
