F02EBF – 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

F02EBF computes all the eigenvalues, and optionally all the eigenvectors, of a real general matrix.

2 Specification

```
      SUBROUTINE F02EBF(JOB, N, A, LDA, WR, WI, VR, LDVR, VI, LDVI,
     1                  WORK, LWORK, IFAIL)
      INTEGER           N, LDA, LDVR, LDVI, LWORK, IFAIL
      real              A(LDA,*), WR(*), WI(*), VR(LDVR,*), VI(LDVI,*),
     1                  WORK(LWORK)
      CHARACTER*1       JOB
```

3 Description

This routine computes all the eigenvalues, and optionally all the right eigenvectors, of a real general matrix A:

$$Ax_i = \lambda_i x_i \text{ for } i = 1,2,\ldots,n.$$

Note that even though A is real, λ_i and x_i may be complex. If x_i is an eigenvector corresponding to a complex eigenvalue λ_i, then the complex conjugate vector $\bar{x}_i$ is the eigenvector corresponding to the complex conjugate eigenvalue $\bar{\lambda}_i$.

4 References

[1] Golub G H and van Loan C F (1996) *Matrix Computations* Johns Hopkins University Press (3rd Edition), Baltimore

5 Parameters

1: JOB — CHARACTER*1 *Input*

On entry: indicates whether eigenvectors are to be computed as follows:

if JOB = 'N', then only eigenvalues are computed;
if JOB = 'V', then eigenvalues and eigenvectors are computed.

Constraint: JOB = 'N' or 'V'.

2: N — INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: N $\geq$ 0.

3: A(LDA,*) — ***real*** array *Input/Output*

Note: the second dimension of the array A must be at least max(1,N).

On entry: the n by n general matrix A.

On exit: if JOB = 'V', A contains the Schur form of the balanced input matrix A' (see Section 8); if JOB = 'N', the contents of A are overwritten.

4: LDA — INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F02EBF is called.

Constraint: LDA $\geq$ max(1,N).

5: WR($*$) — ***real*** array *Output*

Note: the dimension of the array WR must be at least max(1,N).

6: WI($*$) — ***real*** array *Output*

Note: the dimension of the array WI must be at least max(1,N).

On exit: the real and imaginary parts, respectively, of the computed eigenvalues. Complex conjugate pairs of eigenvalues are stored in consecutive elements of the arrays, with the eigenvalue having positive imaginary part first.

7: VR(LDVR,$*$) — ***real*** array *Output*

Note: the second dimension of the array VR must be at least max(1,N) if JOB = 'V' and at least 1 otherwise.

On exit: If JOB = 'V', VR contains the real parts of the eigenvectors, with the ith column holding the real part of the eigenvector associated with the eigenvalue λ_i (stored in WR(i) and WI(i)).

VR is not referenced if JOB = 'N'.

8: LDVR — INTEGER *Input*

On entry: the first dimension of the array VR as declared in the (sub)program from which F02EBF is called.

Constraints:

LDVR $\geq$ 1 if JOB = 'N',
LDVR $\geq$ max(1,N) if JOB = 'V'.

9: VI(LDVI,$*$) — ***real*** array *Output*

Note: the second dimension of the array VI must be at least max(1,N) if JOB = V and at least 1 otherwise.

On exit: If JOB = 'V', VI contains the imaginary parts of the eigenvectors, with the ith column holding the imaginary part of the eigenvector associated with the eigenvalue λ_i (stored in WR(i) and WI(i)).

VI is not referenced if JOB = 'N'.

10: LDVI — INTEGER *Input*

On entry: the first dimension of the array VI as declared in the (sub)program from which F02EBF is called.

Constraints:

LDVI $\geq$ 1 if JOB = 'N',
LDVI $\geq$ max(1,N) if JOB = 'V'.

11: WORK(LWORK) — ***real*** array *Workspace*
12: LWORK — INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F02EBF is called. On some high-performance computers, increasing the dimension of WORK will enable the routine to run faster; a value of 64 $\times$ N should allow near-optimal performance on almost all machines.

Constraint: LWORK $\geq$ max(1,4$\times$N).

13: 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

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

On entry, JOB ≠ 'N' or 'V',
or N < 0,
or LDA < max(1,N),
or LDVR < 1, or LDVR < N and JOB = 'V',
or LDVI < 1, or LDVI < N and JOB = 'V',
or LWORK < max(1,4×N).

IFAIL = 2

The QR algorithm failed to compute all the eigenvalues.

7 Accuracy

If λ_i is an exact eigenvalue, and $\tilde{\lambda}_i$ is the corresponding computed value, then

$$|\tilde{\lambda}_i - \lambda_i| \leq \frac{c(n)\epsilon \| A' \|_2}{s_i},$$

where $c(n)$ is a modestly increasing function of n, ϵ is the ***machine precision***, and s_i is the reciprocal condition number of λ_i; A' is the balanced form of the original matrix A (see Section 8), and $\| A' \| \leq \| A \|$.

If x_i is the corresponding exact eigenvector, and $\tilde{x}_i$ is the corresponding computed eigenvector, then the angle $\theta(x_i, x_i)$ between them is bounded as follows:

$$\theta(x_i, x_i) \leq \frac{c(n)\epsilon \| A' \|_2}{sep_i}$$

where sep_i is the reciprocal condition number of x_i.

The condition numbers s_i and sep_i may be computed by calling F08QLF (STRSNA/DTRSNA), using the Schur form of the balanced matrix A' which is returned in the array A when JOB = 'V'.

8 Further Comments

The routine calls routines from LAPACK in the F08 Chapter Introduction. It first balances the matrix, using a diagonal similarity transformation to reduce its norm; and then reduces the balanced matrix A' to upper Hessenberg form H, using an orthogonal similarity transformation: $A' = QHQ^T$. If only eigenvalues are required, the routine uses the Hessenberg QR algorithm to compute the eigenvalues. If the eigenvectors are required, the routine first forms the orthogonal matrix Q that was used in the reduction to Hessenberg form; it then uses the Hessenberg QR algorithm to compute the Schur factorization of A' as: $A' = ZTZ^T$. It computes the right eigenvectors of T by backward substitution, pre-multiplies them by Z to form the eigenvectors of A'; and finally transforms the eigenvectors to those of the original matrix A.

Each eigenvector x (real or complex) is normalized so that $\| x \|_2 = 1$, and the element of largest absolute value is real and positive.

The time taken by the routine is approximately proportional to n^3.

9 Example

To compute all the eigenvalues and eigenvectors of the matrix A, where

$$A = \begin{pmatrix} 0.35 & 0.45 & -0.14 & -0.17 \\ 0.09 & 0.07 & -0.54 & 0.35 \\ -0.44 & -0.33 & -0.03 & 0.17 \\ 0.25 & -0.32 & -0.13 & 0.11 \end{pmatrix}.$$

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.

```
*     F02EBF Example Program Text
*     Mark 16 Release. NAG Copyright 1992.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, LDA, LDV, LDVI, LDVR, LWORK
      PARAMETER        (NMAX=8,LDA=NMAX,LDV=NMAX,LDVI=NMAX,LDVR=NMAX,
     +                 LWORK=64*NMAX)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, J, N
*     .. Local Arrays ..
      complex          V(LDV,NMAX)
      real             A(LDA,NMAX), VI(LDVI,NMAX), VR(LDVR,NMAX),
     +                 WI(NMAX), WORK(LWORK), WR(NMAX)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F02EBF, X04DBF
*     .. Intrinsic Functions ..
      INTRINSIC        cmplx
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F02EBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N
      IF (N.LE.NMAX) THEN
*
*        Read A from data file
*
         READ (NIN,*) ((A(I,J),J=1,N),I=1,N)
*
*        Compute eigenvalues and eigenvectors of A
*
         IFAIL = 0
*
         CALL F02EBF('Vectors',N,A,LDA,WR,WI,VR,LDVR,VI,LDVI,WORK,LWORK,
     +               IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,*) 'Eigenvalues'
         WRITE (NOUT,99999) (' (',WR(I),',',WI(I),')',I=1,N)
         WRITE (NOUT,*)
         DO 40 J = 1, N
            DO 20 I = 1, N
               V(I,J) = cmplx(VR(I,J),VI(I,J))
   20       CONTINUE
   40    CONTINUE
*
         CALL X04DBF('General',' ',N,N,V,LDV,'Bracketed','F7.4',
     +               'Eigenvectors','Integer',RLABS,'Integer',CLABS,80,
     +               0,IFAIL)
*
      END IF
      STOP
*
99999 FORMAT ((3X,4(A,F7.4,A,F7.4,A,:)))
      END
```

9.2 Program Data

```
F02EBF Example Program Data
  4                            :Value of N
  0.35   0.45  -0.14  -0.17
  0.09   0.07  -0.54   0.35
 -0.44  -0.33  -0.03   0.17
  0.25  -0.32  -0.13   0.11   :End of matrix A
```

9.3 Program Results

```
 F02EBF Example Program Results

 Eigenvalues
    ( 0.7995, 0.0000) (-0.0994, 0.4008) (-0.0994,-0.4008) (-0.1007, 0.0000)

 Eigenvectors
                    1                  2                  3                  4
 1  ( 0.6551, 0.0000) (-0.1933, 0.2546) (-0.1933,-0.2546) ( 0.1253, 0.0000)
 2  ( 0.5236, 0.0000) ( 0.2519,-0.5224) ( 0.2519, 0.5224) ( 0.3320, 0.0000)
 3  (-0.5362, 0.0000) ( 0.0972,-0.3084) ( 0.0972, 0.3084) ( 0.5938, 0.0000)
 4  ( 0.0956, 0.0000) ( 0.6760, 0.0000) ( 0.6760, 0.0000) ( 0.7221, 0.0000)
```