NAG Toolbox for MATLAB

f02eb

1 Purpose

f02eb computes all the eigenvalues, and optionally all the eigenvectors, of a real general matrix.

2 Syntax

```
[a, wr, wi, vr, vi, ifail] = f02eb(job, a, 'n', n)
```

3 Description

f02eb computes all the eigenvalues, and optionally all the right eigenvectors, of a real general matrix A:

$$Ax_i = \lambda_i x_i, \qquad i = 1, 2, \ldots, n.$$

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

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

5 Parameters

5.1 Compulsory Input Parameters

1: **job – string**

Indicates whether eigenvectors are to be computed.

job = 'N'

Only eigenvalues are computed.

job = 'V'

Eigenvalues and eigenvectors are computed.

Constraint: **job** = 'N' or 'V'.

2: **a(lda,*) – double array**

The first dimension of the array **a** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The n by n general matrix A.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the array **n**.

n, the order of the matrix A.

Constraint: $\mathbf{n} \geq 0$.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldvr, ldvi, work, lwork

5.4 Output Parameters

1: **a(lda,∗) – double array**

The first dimension of the array **a** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

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.

2: **wr(∗) – double array**

3: **wi(∗) – double array**

Note: the dimension of the arrays **wr** and **wi** must be at least $\max(1, \mathbf{n})$.

wr and **wi** hold the real and imaginary parts, respectively, of the computed eigenvalues. Complex conjugate pairs of eigenvalues are stored in consecutive elements of **wr** and **wi**, with the eigenvalue having positive imaginary part first.

4: **vr(ldvr,∗) – double array**

The first dimension, **ldvr**, of the array **vr** must satisfy

if **job** = 'N', $\mathbf{ldvr} \geq 1$;
if **job** = 'V', $\mathbf{ldvr} \geq \max(1, \mathbf{n})$.

The second dimension of the array must be at least $\max(1, \mathbf{n})$ if **job** = 'V', and at least 1 otherwise

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 $\mathbf{wr}(i)$ and $\mathbf{wi}(i)$).

If **job** = 'N', **vr** is not referenced.

5: **vi(ldvi,∗) – double array**

The first dimension, **ldvi**, of the array **vi** must satisfy

if **job** = 'N', $\mathbf{ldvi} \geq 1$;
if **job** = 'V', $\mathbf{ldvi} \geq \max(1, \mathbf{n})$.

The second dimension of the array must be at least $\max(1, \mathbf{n})$ if **job** = 'V', and at least 1 otherwise

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 $\mathbf{wr}(i)$ and $\mathbf{wi}(i)$).

If **job** = 'N', **vi** is not referenced.

6: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail = 1

On entry, **job** ≠ 'N' or 'V',
or $\mathbf{n} < 0$,
or $\mathbf{lda} < \max(1, \mathbf{n})$,
or $\mathbf{ldvr} < 1$, or $\mathbf{ldvr} < \mathbf{n}$ and **job** = 'V',

or **ldvi** < 1, or **ldvi** $<$ **n** and **job** $=$ 'V',
or **lwork** $< \max(1, 4 \times \mathbf{n})$.

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 f08ql, using the Schur form of the balanced matrix A' which is returned in the array **a** when **job** $=$ 'V'.

8 Further Comments

f02eb calls functions from LAPACK in Chapter F08. 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 $A' = QHQ^{\mathrm{T}}$. If only eigenvalues are required, the function uses the Hessenberg *QR* algorithm to compute the eigenvalues. If the eigenvectors are required, the function 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 $A' = ZTZ^{\mathrm{T}}$. 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 function is approximately proportional to n^3.

9 Example

```
job = 'Vectors';
a = [0.35, 0.45, -0.14, -0.17;
     0.09, 0.07000000000000001, -0.54, 0.35;
     -0.44, -0.33, -0.03, 0.17;
     0.25, -0.32, -0.13, 0.11];
[aOut, wr, wi, vr, vi, ifail] = f02eb(job, a)
```

```
aOut =
    0.7995    0.0060   -0.1144   -0.0336
         0   -0.0994   -0.6483   -0.2026
         0    0.2478   -0.0994   -0.3474
         0         0         0   -0.1007
wr =
    0.7995
   -0.0994
```

```
   -0.0994
   -0.1007
wi =
         0
    0.4008
   -0.4008
         0
vr =
    0.6551   -0.1933   -0.1933    0.1253
    0.5236    0.2519    0.2519    0.3320
   -0.5362    0.0972    0.0972    0.5938
    0.0956    0.6760    0.6760    0.7221
vi =
         0    0.2546   -0.2546         0
         0   -0.5224    0.5224         0
         0   -0.3084    0.3084         0
         0         0         0         0
ifail =
           0
```