NAG Toolbox for MATLAB

f08jv

1 Purpose

f08jv computes all the eigenvalues and, optionally, all the eigenvectors of a real n by n symmetric tridiagonal matrix, or of a complex full or banded Hermitian matrix which has been reduced to tridiagonal form.

2 Syntax

```
[d, e, z, info] = f08jv(compz, d, e, z, 'n', n)
```

3 Description

f08jv computes all the eigenvalues, and optionally the eigenvectors, of a real symmetric tridiagonal matrix T. That is, the function computes the spectral factorization of T given by

$$T = Z\Lambda Z^{\mathrm{T}},$$

where Λ is a diagonal matrix whose diagonal elements are the eigenvalues, λ_i, of T and Z is an orthogonal matrix whose columns are the eigenvectors, z_i, of T. Thus

$$Tz_i = \lambda_i z_i, \qquad i = 1, 2, \dots, n.$$

The function may also be used to compute all the eigenvalues and vectors of a real full, or banded, Hermitian matrix A which has been reduced to real tridiagonal form T as

$$A = QTQ^{\mathrm{H}},$$

where Q is unitary. The spectral factorization of A is then given by

$$A = (QZ)\Lambda(QZ)^{\mathrm{H}}.$$

In this case Q must be formed explicitly and passed to f08jv in the array **z**, and the function called with **compz** = 'V'. Functions which may be called to form T and Q are

full matrix	f08fs and f08ft
full matrix, packed storage	f08gs and f08gt
band matrix	f08hs, with **vect** = 'V'

When only eigenvalues are required then this function calls f08jf to compute the eigenvalues of the tridiagonal matrix T, but when eigenvectors of T are also required and the matrix is not too small, then a divide and conquer method is used, which can be much faster than f08js, although more storage is required.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D 1999 *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia URL: http://www.netlib.org/lapack/lug

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: **compz – string**

Indicates whether the eigenvectors are to be computed.

compz = 'N'

Only the eigenvalues are computed (and the array **z** is not referenced).

compz = 'I'

The eigenvalues and eigenvectors of T are computed (and the array **z** is initialized by the function).

compz = 'V'

The eigenvalues and eigenvectors of A are computed (and the array **z** must contain the matrix Q on entry).

Constraint: **compz** = 'N', 'V' or 'I'.

2: **d**$(*)$ **– double array**

Note: the dimension of the array **d** must be at least $\max(1,\mathbf{n})$.

The diagonal elements of the tridiagonal matrix.

3: **e**$(*)$ **– double array**

Note: the dimension of the array **e** must be at least $\max(1,\mathbf{n}-1)$.

The subdiagonal elements of the tridiagonal matrix.

4: **z**(**ldz**,$*$) **– complex array**

The first dimension, **ldz**, of the array **z** must satisfy

if **compz** = 'V' or 'I', **ldz** $\geq \max(1,\mathbf{n})$;
ldz ≥ 1 otherwise.

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

If **compz** = 'V', **z** must contain the unitary matrix used in the reduction to tridiagonal form.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the array **d** The second dimension of the array **z**.

n, the order of the symmetric tridiagonal matrix T.

Constraint: **n** ≥ 0.

5.3 Input Parameters Omitted from the MATLAB Interface

ldz, work, lwork, rwork, lrwork, iwork, liwork

5.4 Output Parameters

1: **d**$(*)$ **– double array**

Note: the dimension of the array **d** must be at least $\max(1,\mathbf{n})$.

If **info** $= 0$, the eigenvalues in ascending order.

2: **e**($*$) **– double array**

Note: the dimension of the array **e** must be at least $\max(1, \mathbf{n} - 1)$.

e is overwritten.

3: **z**(**ldz**,$*$) **– complex array**

The first dimension, **ldz**, of the array **z** must satisfy

if **compz** = 'V' or 'I', **ldz** $\geq \max(1, \mathbf{n})$;
ldz ≥ 1 otherwise.

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

If **info** $= 0$, then if **compz** = 'V', **z** contains the orthonormal eigenvectors of the original Hermitian matrix,and if **compz** = 'I', **z** contains the orthonormal eigenvectors of the symmetric tridiagonal matrix.

If **compz** = 'N', **z** is not referenced.

4: **info – int32 scalar**

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

6 Error Indicators and Warnings

Errors or warnings detected by the function:

info $= -i$

If **info** $= -i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **compz**, 2: **n**, 3: **d**, 4: **e**, 5: **z**, 6: **ldz**, 7: **work**, 8: **lwork**, 9: **rwork**, 10: **lrwork**, 11: **iwork**, 12: **liwork**, 13: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

info > 0

The algorithm failed to compute an eigenvalue while working on the submatrix lying in rows and columns $\mathbf{info}/(\mathbf{n} + 1)$ through $\mathrm{mod}(\mathbf{info}, \mathbf{n} + 1)$.

7 Accuracy

The computed eigenvalues and eigenvectors are exact for a nearby matrix $(T + E)$, where

$$\|E\|_2 = O(\epsilon)\|T\|_2,$$

and ϵ is the ***machine precision***.

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

$$\left|\tilde{\lambda}_i - \lambda_i\right| \leq c(n)\epsilon\|T\|_2,$$

where $c(n)$ is a modestly increasing function of n.

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

$$\theta(\tilde{z}_i, z_i) \leq \frac{c(n)\epsilon\|T\|_2}{\min\limits_{i \neq j}\left|\lambda_i - \lambda_j\right|}.$$

Thus the accuracy of a computed eigenvector depends on the gap between its eigenvalue and all the other eigenvalues.

See Section 4.7 of Anderson *et al.* 1999 for further details. See also f08fl.

8 Further Comments

If only eigenvalues are required, the total number of floating point operations is approximately proportional to n^2. When eigenvectors are required the number of operations is bounded above by approximately the same number of operations as f08js, but for large matrices f08jv is usually much faster.

The real analogue of this function is f08jh.

9 Example

```
compz = 'V';
d = [-3.13;
     -1.28529412359;
     -0.9961823111303629;
     -1.998523565279637];
e = [4.449280840765168;
     1.700240893426197;
     2.514369223733909];
z = [complex(1, +0), complex(0, +0), complex(0, +0), complex(0, +0);
       complex(0, +0), complex(0.4360255217484467, +0.4719863895215144),
complex(0.178928959228093, ...
                +0.2536888179436562),   complex(-0.6192449219137889,   -
0.3275251599103912);
       complex(0, +0), complex(-0.76416844017769, -0.05618885589541835),
complex(0.2548894129557567, ...
                   +0.05297877813842256),    complex(-0.5669454100367172,
+0.1539056392010643);
         complex(0,  +0),  complex(0,  +0),  complex(0.8716895470004201,  -
0.2756961753092227), ...
     complex(0.2890720654179352, -0.2838772508762855)];
[dOut, eOut, zOut, info] = f08jv(compz, d, e, z)
```

```
dOut =
   -7.0042
   -4.0038
    0.5968
    3.0012
eOut =
     0
     0
     0
zOut =
  -0.7293             0.2610            -0.3718            -0.5117
    0.1654 + 0.2046i  -0.5966 - 0.4235i  -0.4002 - 0.1869i  -0.2493 -
0.3718i
   -0.6081 - 0.0301i  -0.5413 + 0.0870i   0.0531 + 0.1444i   0.5521 -
0.0176i
   -0.1653 + 0.0303i  -0.3061 - 0.0482i   0.7234 - 0.3459i  -0.4461 +
0.1836i
info =
           0
```