NAG Toolbox for MATLAB

f08qx

1 Purpose

f08qx computes selected left and/or right eigenvectors of a complex upper triangular matrix.

2 Syntax

```
[t, vl, vr, m, info] = f08qx(job, howmny, select, t, vl, vr, mm, 'n', n)
```

3 Description

f08qx computes left and/or right eigenvectors of a complex upper triangular matrix T. Such a matrix arises from the Schur factorization of a complex general matrix, as computed by f08ps, for example.

The right eigenvector x, and the left eigenvector y, corresponding to an eigenvalue λ, are defined by:

$$Tx = \lambda x \text{ and } y^{\mathrm{H}}T = \lambda y^{\mathrm{H}} \left(\text{or } T^{\mathrm{H}}y = \bar{\lambda}y\right).$$

The function can compute the eigenvectors corresponding to selected eigenvalues, or it can compute all the eigenvectors. In the latter case the eigenvectors may optionally be pre-multiplied by an input matrix Q. Normally Q is a unitary matrix from the Schur factorization of a matrix A as $A = QTQ^{\mathrm{H}}$; if x is a (left or right) eigenvector of T, then Qx is an eigenvector of A.

The eigenvectors are computed by forward or backward substitution. They are scaled so that $\max|\mathrm{Re}(x_i)| + |\mathrm{Im}\, x_i| = 1$.

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 left and/or right eigenvectors are to be computed.

job = 'R'

Only right eigenvectors are computed.

job = 'L'

Only left eigenvectors are computed.

job = 'B'

Both left and right eigenvectors are computed.

Constraint: **job** = 'R', 'L' or 'B'.

2: **howmny – string**

Indicates how many eigenvectors are to be computed.

howmny = 'A'

All eigenvectors (as specified by **job**) are computed.

howmny = 'B' or 'O'

All eigenvectors (as specified by **job**) are computed and then pre-multiplied by the matrix Q (which is overwritten).

howmny = 'S'

Selected eigenvectors (as specified by **job** and **select**) are computed.

Constraint: **howmny** = 'A', 'B', 'O' or 'S'.

3: **select**($*$) – **logical array**

Note: the dimension of the array **select** must be at least $\max(1, \mathbf{n})$ if **howmny** = 'S', and at least 1 otherwise.

Specifies which eigenvectors are to be computed if **howmny** = 'S'. To obtain the eigenvector corresponding to the eigenvalue λ_j, **select**(j) must be set **true**.

If **howmny** = 'A', 'O' or 'B', **select** is not referenced.

4: **t**(**ldt**,$*$) – **complex array**

The first dimension of the array **t** 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 upper triangular matrix T, as returned by f08ps.

5: **vl**(**ldvl**,$*$) – **complex array**

The first dimension, **ldvl**, of the array **vl** must satisfy

if **job** = 'L' or 'B', **ldvl** $\geq \max(1, \mathbf{n})$;
if **job** = 'R', **ldvl** ≥ 1.

The second dimension of the array must be at least $\max(1, \mathbf{mm})$ if **job** = 'L' or 'B' and at least 1 if **job** = 'R'

If **howmny** = 'O' or 'B' and **job** = 'L' or 'B', **vl** must contain an n by n matrix Q (usually the matrix of Schur vectors returned by f08ps).

If **howmny** = 'A' or 'S', **vl** need not be set.

6: **vr**(**ldvr**,$*$) – **complex array**

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

if **job** = 'R' or 'B', **ldvr** $\geq \max(1, \mathbf{n})$;
if **job** = 'L', **ldvr** ≥ 1.

The second dimension of the array must be at least $\max(1, \mathbf{mm})$ if **job** = 'R' or 'B' and at least 1 if **job** = 'L'

If **howmny** = 'O' or 'B' and **job** = 'R' or 'B', **vr** must contain an n by n matrix Q (usually the matrix of Schur vectors returned by f08ps).

If **howmny** = 'A' or 'S', **vr** need not be set.

7: **mm** – **int32 scalar**

The number of columns in the arrays **vl** and/or **vr**. The precise number of columns required, m, is n if **howmny** = 'A', 'O' or 'B'; if **howmny** = 'S', m is the number of selected eigenvectors (see **select**), in which case $0 \leq m \leq n$.

Constraint: **mm** $\geq m$.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The second dimension of the array **t**.

n, the order of the matrix T.

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

5.3 Input Parameters Omitted from the MATLAB Interface

ldt, ldvl, ldvr, work, rwork

5.4 Output Parameters

1: **t(ldt,$*$) – complex array**

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

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

Is used as internal workspace prior to being restored and hence is unchanged.

2: **vl(ldvl,$*$) – complex array**

The first dimension, **ldvl**, of the array **vl** must satisfy

if **job** = 'L' or 'B', $\mathbf{ldvl} \geq \max(1, \mathbf{n})$;
if **job** = 'R', $\mathbf{ldvl} \geq 1$.

The second dimension of the array must be at least $\max(1, \mathbf{mm})$ if **job** = 'L' or 'B' and at least 1 if **job** = 'R'

If **job** = 'L' or 'B', **vl** contains the computed left eigenvectors (as specified by **howmny** and **select**). The eigenvectors are stored consecutively in the columns of the array, in the same order as their eigenvalues.

If **job** = 'R', **vl** is not referenced.

3: **vr(ldvr,$*$) – complex array**

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

if **job** = 'R' or 'B', $\mathbf{ldvr} \geq \max(1, \mathbf{n})$;
if **job** = 'L', $\mathbf{ldvr} \geq 1$.

The second dimension of the array must be at least $\max(1, \mathbf{mm})$ if **job** = 'R' or 'B' and at least 1 if **job** = 'L'

If **job** = 'R' or 'B', **vr** contains the computed right eigenvectors (as specified by **howmny** and **select**). The eigenvectors are stored consecutively in the columns of the array, in the same order as their eigenvalues.

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

4: **m – int32 scalar**

m, the number of selected eigenvectors. If **howmny** = 'A', 'O' or 'B', **m** is set to n.

5: **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: **job**, 2: **howmny**, 3: **select**, 4: **n**, 5: **t**, 6: **ldt**, 7: **vl**, 8: **ldvl**, 9: **vr**, 10: **ldvr**, 11: **mm**, 12: **m**, 13: **work**, 14: **rwork**, 15: **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.

7 Accuracy

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

$$\theta(\tilde{x}_i, x_i) \leq \frac{c(n)\epsilon\|T\|_2}{sep_i}$$

where sep_i is the reciprocal condition number of x_i.

The condition number sep_i may be computed by calling f08qy.

8 Further Comments

The real analogue of this function is f08qk.

9 Example

```
a = [complex(1.5, -2.75), complex(0, +0), complex(0, +0), complex(0, +0);
          complex(-8.06,  -1.24),  complex(-2.5,  -0.5),  complex(0,  +0),
complex(-0.75, +0.5);
      complex(-2.09, +7.56), complex(1.39, +3.97), complex(-1.25, +0.75),
complex(-4.82, -5.67);
          complex(6.18,  +9.79),  complex(-0.92,  -0.62),  complex(0,  +0),
complex(-2.5, -0.5)];
select = [false];
vl = [complex(0, 0)];
mm = int32(4);
% Balance a
[a, ilo, ihi, scale, info] = f08nv('Both', a);
% Reduce a to upper Hessenberg form
[a, tau, info] = f08ns(ilo, ihi, a);
% Form Q explicitly, storing result in vr
[vr, info] = f08nt(int32(1), int32(4), a, tau);
% Calculate the eigenvalues and Schur factorisation of a
[h, w, vr, info] = f08ps('Schur Form', 'Vectors', ilo, ihi, a, vr);
% Calculate the eigenvectors of a, storing the result in vrOut
[tOut, vlOut, vrOut, m, info] = f08qx('Right', 'Backtransform', select,
h, vl, vr, mm);
[v, info] = f08nw('Both', 'Right', ilo, ihi, scale, vrOut)
```

```
v =
        0                  0                  0              0.1452
          0             -0.0616 + 0.0413i    0.4613 - 0.0000i   -0.2072 -
0.2450i
    1.0000               0.6032 - 0.3968i    0.2983 + 0.7017i    0.7768 +
0.2232i
          0              0.0822               0.4251 + 0.2850i   -0.0119 +
0.4372i
```

```
info =
           0
```