NAG Toolbox for MATLAB

f08ps

1 Purpose

f08ps computes all the eigenvalues and, optionally, the Schur factorization of a complex Hessenberg matrix or a complex general matrix which has been reduced to Hessenberg form.

2 Syntax

```
[h, w, z, info] = f08ps(job, compz, ilo, ihi, h, z, 'n', n)
```

3 Description

f08ps computes all the eigenvalues and, optionally, the Schur factorization of a complex upper Hessenberg matrix H:

$$H = ZTZ^{\mathrm{H}},$$

where T is an upper triangular matrix (the Schur form of H), and Z is the unitary matrix whose columns are the Schur vectors z_i. The diagonal elements of T are the eigenvalues of H.

The function may also be used to compute the Schur factorization of a complex general matrix A which has been reduced to upper Hessenberg form H:

$$\begin{aligned} A &= QHQ^{\mathrm{H}}, \text{where } Q \text{ is unitary,} \\ &= (QZ)T(QZ)^{\mathrm{H}}. \end{aligned}$$

In this case, after f08ns has been called to reduce A to Hessenberg form, f08nt must be called to form Q explicitly; Q is then passed to f08ps, which must be called with **compz** = 'V'.

The function can also take advantage of a previous call to f08nv which may have balanced the original matrix before reducing it to Hessenberg form, so that the Hessenberg matrix H has the structure:

$$\begin{pmatrix} H_{11} & H_{12} & H_{13} \\ & H_{22} & H_{23} \\ & & H_{33} \end{pmatrix}$$

where H_{11} and H_{33} are upper triangular. If so, only the central diagonal block H_{22} (in rows and columns i_{lo} to i_{hi}) needs to be further reduced to Schur form (the blocks H_{12} and H_{23} are also affected). Therefore the values of i_{lo} and i_{hi} can be supplied to f08ps directly. Also, f08nw must be called after this function to permute the Schur vectors of the balanced matrix to those of the original matrix. If f08nv has not been called however, then i_{lo} must be set to 1 and i_{hi} to n. Note that if the Schur factorization of A is required, f08nv must **not** be called with **job** = 'S' or 'B', because the balancing transformation is not unitary.

f08ps uses a multishift form of the upper Hessenberg QR algorithm, due to Bai and Demmel 1989. The Schur vectors are normalized so that $\|z_i\|_2 = 1$, but are determined only to within a complex factor of absolute value 1.

4 References

Bai Z and Demmel J W 1989 On a block implementation of Hessenberg multishift *QR* iteration *Internat. J. High Speed Comput.* **1** 97–112

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 eigenvalues only or the Schur form T is required.

job = 'E'

Eigenvalues only are required.

job = 'S'

The Schur form T is required.

Constraint: **job** = 'E' or 'S'.

2: **compz – string**

Indicates whether the Schur vectors are to be computed.

compz = 'N'

No Schur vectors are computed (and the array **z** is not referenced).

compz = 'I'

The Schur vectors of H are computed (and the array **z** is initialized by the function).

compz = 'V'

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

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

3: **ilo – int32 scalar**
4: **ihi – int32 scalar**

If the matrix A has been balanced by f08nv, then **ilo** and **ihi** must contain the values returned by that function. Otherwise, **ilo** must be set to 1 and **ihi** to **n**.

Constraint: **ilo** ≥ 1 and $\min(\mathbf{ilo}, \mathbf{n}) \leq \mathbf{ihi} \leq \mathbf{n}$.

5: **h(ldh,$*$) – complex array**

The first dimension of the array **h** 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 Hessenberg matrix H, as returned by f08ns.

6: **z(ldz,$*$) – complex array**

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

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

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

If **compz** = 'V', **z** must contain the unitary matrix Q from the reduction to Hessenberg form.

If **compz** = 'I', **z** need not be set.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The first dimension of the array **h** and the second dimension of the array **h**. (An error is raised if these dimensions are not equal.)

n, the order of the matrix H.

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

5.3 Input Parameters Omitted from the MATLAB Interface

ldh, ldz, work, lwork

5.4 Output Parameters

1: **h(ldh,*) – complex array**

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

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

If **job** = 'E', the array contains no useful information.

If **job** = 'S', **h** contains the upper triangular matrix T from the Schur decomposition (the Schur form) unless **info** > 0.

2: **w(*) – complex array**

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

The computed eigenvalues, unless **info** > 0 (in which case see Section 6). The eigenvalues are stored in the same order as on the diagonal of the Schur form T (if computed).

3: **z(ldz,*) – complex array**

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

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

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

If **compz** = 'V' or 'I', **z** contains the unitary matrix of the required Schur vectors, unless **info** > 0.

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: **job**, 2: **compz**, 3: **n**, 4: **ilo**, 5: **ihi**, 6: **h**, 7: **ldh**, 8: **w**, 9: **z**, 10: **ldz**, 11: **work**, 12: **lwork**, 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 has failed to find all the eigenvalues after a total of $30 \times (\mathbf{ihi} - \mathbf{ilo} + 1)$ iterations. If $\mathbf{info} = i$, elements $1, 2, \ldots, \mathbf{ilo} - 1$ and $i + 1, i + 2, \ldots, n$ of **w** contain the eigenvalues which have been found.

If **job** = 'E', then on exit, the remaining unconverged eigenvalues are the eigenvalues of the upper Hessenberg matrix rows and columns of **ilo** through **info** of the final, output value of H.

If **job** = 'S', then on exit

$$(*) \qquad (\text{initial value of } H) * U = U * (\text{final value of } H)$$

where U is a unitary matrix. The final value of H is upper Hessenberg and triangular in rows and columns $\mathbf{info} + 1$ through **ihi**.

If **compz** = 'V', then on exit

$$(\text{final value of } Z) = (\text{initial value of } Z) * U$$

where U is the unitary matrix in $(*)$ (regardless of the value of **job**).

If **compz** = 'I', then on exit

$$(\text{final value of } Z) = U$$

where U is the unitary matrix in $(*)$ (regardless of the value of **job**).

If **info** > 0 and **compz** = 'N', then **z** is not accessed.

7 Accuracy

The computed Schur factorization is the exact factorization of a nearby matrix $(H + E)$, where

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

and ϵ is the ***machine precision***.

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

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

where $c(n)$ is a modestly increasing function of n, and s_i is the reciprocal condition number of λ_i. The condition numbers s_i may be computed by calling f08qy.

8 Further Comments

The total number of real floating-point operations depends on how rapidly the algorithm converges, but is typically about:

$25n^3$ if only eigenvalues are computed;

$35n^3$ if the Schur form is computed;

$70n^3$ if the full Schur factorization is computed.

The real analogue of this function is f08pe.

9 Example

```
job = 'Schur form';
compz = 'Initialize Z';
ilo = int32(1);
ihi = int32(4);
h = [complex(-3.97, -5.04), complex(-1.1318, -2.5693), complex(-4.6027, -
```

```
0.1426), complex(-1.4249, +1.733);
       complex(-5.4797, +0), complex(1.8585, -1.5502), complex(4.4145, -
0.7638), complex(-0.4805, -1.1976);
        complex(0, +0), complex(6.2673, +0), complex(-0.4504, -0.029),
complex(-1.3467, +1.6579);
     complex(0, +0), complex(0, +0), complex(-3.5, +0), complex(2.5619, -
3.3708)];
z = complex(zeros(4, 4));
[hOut, w, zOut, info] = f08ps(job, compz, ilo, ihi, h, z)
```

```
hOut =
   -6.0004 - 6.9998i  -0.2080 + 0.4719i  -0.4829 + 0.1768i   0.1301 +
0.9052i
         0            -5.0000 + 2.0060i  -0.6653 + 0.2814i   0.0038 +
0.2639i
         0                  0             7.9982 - 0.9964i   0.2004 +
1.0595i
         0                  0                  0             3.0023 -
3.9998i
w =
  -6.0004 - 6.9998i
  -5.0000 + 2.0060i
   7.9982 - 0.9964i
   3.0023 - 3.9998i
zOut =
    0.8457             0.1380 + 0.3602i  -0.2677 - 0.1091i  -0.2213 -
0.0582i
    0.2824 - 0.3304i  -0.4612 + 0.2075i   0.6846             0.2927 +
0.0320i
    0.0748 + 0.2800i   0.7239             0.5924 - 0.0189i  -0.0229 +
0.2005i
   0.0670 + 0.0860i   0.2169 + 0.1560i  -0.2745 + 0.1454i   0.9057
info =
           0
```