NAG Toolbox for MATLAB

f08ws

1 Purpose

f08ws reduces a pair of complex matrices (A, B), where B is upper triangular, to the generalized upper Hessenberg form using unitary transformations.

2 Syntax

```
[a, b, q, z, info] = f08ws(compq, compz, ilo, ihi, a, b, q, z, 'n', n)
```

3 Description

f08ws is usually the third step in the solution of the complex generalized eigenvalue problem

$$Ax = \lambda Bx.$$

The (optional) first step balances the two matrices using f08wv. In the second step, matrix B is reduced to upper triangular form using the QR factorization function f08as and this unitary transformation Q is applied to matrix A by calling f08au.

f08ws reduces a pair of complex matrices (A, B), where B is triangular, to the generalized upper Hessenberg form using unitary transformations. This two-sided transformation is of the form

$$Q^{\mathrm{H}}AZ = H$$
$$Q^{\mathrm{H}}BZ = T$$

where H is an upper Hessenberg matrix, T is an upper triangular matrix and Q and Z are unitary matrices determined as products of Givens rotations. They may either be formed explicitly, or they may be post multiplied into input matrices Q_1 and Z_1, so that

$$Q_1AZ_1^{\mathrm{H}} = (Q_1Q)H(Z_1Z)^{\mathrm{H}},$$
$$Q_1BZ_1^{\mathrm{H}} = (Q_1Q)T(Z_1Z)^{\mathrm{H}}.$$

4 References

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

Moler C B and Stewart G W 1973 An algorithm for generalized matrix eigenproblems *SIAM J. Numer. Anal.* **10** 241–256

5 Parameters

5.1 Compulsory Input Parameters

1: **compq – string**

Specifies the form of the computed unitary matrix Q.

compq = 'N'

Do not compute Q.

compq = 'I'

The unitary matrix Q is returned.

compq = 'V'

q must contain a unitary matrix Q_1, and the product Q_1Q is returned.

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

2: **compz – string**

Specifies the form of the computed unitary matrix Z.

compz = 'N'

Do not compute Z.

compz = 'I'

The unitary matrix Z is returned.

compz = 'V'

z must contain a unitary matrix Z_1, and the product Z_1Z is returned.

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

3: **ilo – int32 scalar**

4: **ihi – int32 scalar**

i_{lo} and i_{hi} as determined by a previous call to f08wv. Otherwise, they should be set to 1 and n, respectively.

Constraints:

if $\mathbf{n} > 0$, $1 \leq \mathbf{ilo} \leq \mathbf{ihi} \leq \mathbf{n}$;
if $\mathbf{n} = 0$, $\mathbf{ilo} = 1$ and $\mathbf{ihi} = 0$.

5: **a(lda,∗) – complex 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 matrix A of the matrix pair (A, B). Usually, this is the matrix A returned by f08au.

6: **b(ldb,∗) – complex array**

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

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

The upper triangular matrix B of the matrix pair (A, B). Usually, this is the matrix B returned by the QR factorization function f08as.

7: **q(ldq,∗) – complex array**

The first dimension, **ldq**, of the array **q** must satisfy

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

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

If **compq** = 'V', **q** must contain a unitary matrix Q_1.

If **compq** = 'N', **q** is not referenced.

8: **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 a unitary matrix Z_1.

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

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The second dimension of the array **a** The second dimension of the array **b**.

n, the order of the matrices A and B.

Constraint: **n** ≥ 0.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, ldq, ldz

5.4 Output Parameters

1: **a**(**lda**,∗) – **complex 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 $\max(1, \mathbf{n})$

a contains the upper Hessenberg matrix H.

2: **b**(**ldb**,∗) – **complex array**

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

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

b contains the upper triangular matrix T.

3: **q**(**ldq**,∗) – **complex array**

The first dimension, **ldq**, of the array **q** must satisfy

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

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

If **compq** = 'I', **q** contains the unitary matrix Q.

Iif **compq** = 'V', **q** contains $Q_1 Q$.

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

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

if **compz** = 'V' or 'I', **ldz** $\geq \max(1, \mathbf{n})$;
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** = 'I', **z** contains the unitary matrix Z.

If **compz** = 'V', **z** contains $Z_1 Z$.

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: **compq**, 2: **compz**, 3: **n**, 4: **ilo**, 5: **ihi**, 6: **a**, 7: **lda**, 8: **b**, 9: **ldb**, 10: **q**, 11: **ldq**, 12: **z**, 13: **ldz**, 14: **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

The reduction to the generalized Hessenberg form is implemented using unitary transformations which are backward stable.

8 Further Comments

This function is usually followed by f08xs which implements the QZ algorithm for computing generalized eigenvalues of a reduced pair of matrices.

The real analogue of this function is f08we.

9 Example

```
compq = 'V';
compz = 'V';
ilo = int32(1);
ihi = int32(4);
a    =    [complex(-2.867890104378767,    -1.594524360587801),    complex(-
4.687788932894173, ...
                      -1.335343722471056),     complex(-0.7905009462560725,
+0.07944449994448066), ...
     complex(-1.439015977717754, +0.801769669652453);
           complex(-2.515655332627996,  +0.5597332416374011),  complex(-
4.543587844404469, ...
               +0.03162421545477125),    complex(-0.9404703494183805,    -
0.1305718159625315), ...
     complex(-2.391088344496183, -0.5474958021881119);
                    complex(0.9137894874444608,    -0.09397719654058602),
complex(1.001627625946435, ...
                   +0.1053096496371175),     complex(-0.09620027967881145,
+0.03545150990580886), ...
     complex(-1.284793228244642, +0.04644135395523489);
         complex(-0.08371480021894673,  +0.004924400012879063),  complex(-
0.04267813344495323, ...
                -0.01149026669671837),    complex(0.02339090006117631,    -
0.002215980005795647), ...
```

```
     complex(0.09323530691051313, -0.000164146667095837)];
b   =   [complex(-1.774823934929885,   +0),   complex(-4.744132549876074,
+0.8845947866158476), ...
          complex(-0.9189643929748073,  +0.3876440848354797),  complex(-
1.548322594662643, +1.005733563126935);
           complex(0.3603832255487837,  +0.3603832255487837),  complex(-
2.271717062641915, ...
       +0),  complex(-0.8105776924797564,  -0.1026200900444425),  complex(-
2.074151659012492, -0.4057262470104196);
                    complex(0.03603832255487837,    +0.07207664510975674),
complex(0.149331810255888, ...
      -0.03084332363005019),  complex(-0.3174420096804637,  +0),  complex(-
1.43580010358989, -0.02051468561236543);
                     complex(0.03603832255487837,    +0.1081149676646351),
complex(0.3578350975272399, ...
                -0.04233510073426491),    complex(0.9002179361271462,    -
0.07846747522359962), complex(-0.04284228011204969, +0)];
q   =   [complex(-0.5634361698190111,   +0),   complex(0.296258668864855,   -
0.659594772189678), ...
                     complex(-0.2836756402009104,     +0.270405353127545),
complex(0.07304526685770857, -0.02790493340631572);
           complex(-0.5634361698190111,  -0.5634361698190111),  complex(-
0.3647335498760716, ...
                +0.07685956031871256),    complex(0.4401518581623766,    -
0.08035034801548335), ...
     complex(-0.160453367086315, +0.01436283337089795);
          complex(-0.05634361698190112,  -0.1126872339638022),  complex(-
0.1906117095904069, ...
                +0.0813313892827103),    complex(-0.03681884555986346,    -
0.01465002486487194), ...
     complex(0.9684653358662497, -0.04103666677399458);
          complex(-0.05634361698190112,  -0.1690308509457034),  complex(-
0.5208283296506205, ...
                    +0.1549768225335494),     complex(-0.8021082396915107,
+0.03565614517376269), ...
     complex(-0.1682503337733741, +0.004103666677399493)];
z = [complex(1, +0), complex(0, +0), complex(0, +0), complex(0, +0);
     complex(0, +0), complex(1, +0), complex(0, +0), complex(0, +0);
     complex(0, +0), complex(0, +0), complex(1, +0), complex(0, +0);
     complex(0, +0), complex(0, +0), complex(0, +0), complex(1, +0)];
[aOut, bOut, qOut, zOut, info] = f08ws(compq, compz, ilo, ihi, a, b, q,
z)
```

```
aOut =
   -2.8679 - 1.5945i   -0.8093 - 0.3277i   -4.9004 - 0.9865i   -0.0483 +
1.1626i
   -2.6719 + 0.5945i   -0.7895 + 0.0490i   -4.9549 - 0.1634i   -0.4386 -
0.5739i
         0              -0.0983 - 0.0115i   -1.1677 - 0.1366i   -1.7562 -
0.2054i
         0                    0              0.0873 + 0.0038i    0.0317 +
0.0014i
bOut =
   -1.7748              -0.7210 + 0.0429i   -5.0207 + 1.1898i   -0.1450 +
0.7257i
         0              -0.2176 + 0.0352i   -2.5411 - 0.1458i   -0.8229 -
0.4184i
         0                    0             -1.3958 - 0.1633i   -1.7475 -
0.2044i
         0                    0                   0             -0.0996 -
0.0044i
qOut =
   -0.5634               0.3864 - 0.7006i   -0.1970 + 0.0230i    0.0557 -
0.0024i
   -0.5634 - 0.5634i   -0.4982 + 0.0816i    0.3026 - 0.0354i   -0.1319 +
0.0058i
   -0.0563 - 0.1127i   -0.1383 + 0.0864i   -0.1414 + 0.0165i    0.9672 -
0.0423i
   -0.0563 - 0.1690i   -0.2267 + 0.1646i   -0.9144 + 0.1070i   -0.2052 +
0.0090i
```

```
zOut =
   1.0000                  0                  0                  0
         0               0.2838               0.9127 - 0.0000i  -0.2942 +
0.0000i
         0              -0.9458 - 0.0000i    0.3168 + 0.0000i    0.0706 +
0.0000i
         0               0.1576 + 0.0000i    0.2582 + 0.0000i    0.9531 -
0.0000i
info =
           0
```