NAG Toolbox for MATLAB

f08vs

1 Purpose

f08vs uses unitary transformations to simultaneously reduce the m by n matrix A and the p by n matrix B to upper triangular form. This factorization is usually used as a preprocessing step for computing the generalized singular value decomposition (GSVD).

2 Syntax

```
[a, b, k, l, u, v, q, info] = f08vs(jobu, jobv, jobq, a, b, tola, tolb,
'm', m, 'p', p, 'n', n)
```

3 Description

f08vs computes unitary matrices U, V and Q such that

$$U^{\mathrm{H}}AQ = \begin{cases} \begin{array}{c} \\ k \\ l \\ m-k-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & A_{12} & A_{13} \\ 0 & 0 & A_{23} \\ 0 & 0 & 0 \end{pmatrix} \end{array}, \text{ if } m-k-l \geq 0; \\ \begin{array}{c} \\ k \\ m-k \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & A_{12} & A_{13} \\ 0 & 0 & A_{23} \end{pmatrix} \end{array}, \text{ if } m-k-l < 0; \end{cases}$$

$$V^{\mathrm{H}}BQ = \begin{array}{c} \\ l \\ p-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & 0 & B_{13} \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

where the k by k matrix A_{12} and l by l matrix B_{13} are nonsingular upper triangular; A_{23} is l by l upper triangular if $m-k-l \geq 0$ and is $(m-k)$ by l upper trapezoidal otherwise. $(k+l)$ is the effective numerical rank of the $(m+p)$ by n matrix $\begin{pmatrix} A^{\mathrm{H}} & B^{\mathrm{H}} \end{pmatrix}^{\mathrm{H}}$.

This decomposition is usually used as the preprocessing step for computing the Generalized Singular Value Decomposition (GSVD), see function f08vn.

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: **jobu – string**

If **jobu** = 'U', the orthogonal matrix U is computed.

If **jobu** = 'N', **u** is not computed.

Constraint: **jobu** = 'U' or 'N'.

2: **jobv – string**

If **jobv** = 'V', the unitary matrix V is computed.

If **jobv** = 'N', **v** is not computed.

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

3: **jobq – string**

If **jobq** = 'Q', the unitary matrix Q is computed.

If **jobq** = 'N', **q** is not computed.

Constraint: **jobq** = 'Q' or 'N'.

4: **a**(**lda**,∗) **– complex array**

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

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

The m by n matrix A.

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

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

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

The p by n matrix B.

6: **tola – double scalar**

7: **tolb – double scalar**

tola and **tolb** are the thresholds to determine the effective numerical rank of matrix B and a subblock of A. Generally, they are set to

$$\begin{aligned} \mathbf{tola} &= \max(\mathbf{m}, \mathbf{n})\|A\|\epsilon, \\ \mathbf{tolb} &= \max(\mathbf{p}, \mathbf{n})\|B\|\epsilon, \end{aligned}$$

where ϵ is the ***machine precision***.

The size of **tola** and **tolb** may affect the size of backward errors of the decomposition.

5.2 Optional Input Parameters

1: **m – int32 scalar**

Default: The first dimension of the array **a**.

m, the number of rows of the matrix A.

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

2: **p – int32 scalar**

Default: The first dimension of the array **b**.

p, the number of rows of the matrix B.

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

3: **n – int32 scalar**

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

n, the number of columns of the matrices A and B.

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

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldb, ldu, ldv, ldq, iwork, rwork, tau, work

5.4 Output Parameters

1: **a**(**lda**,$*$) – **complex array**

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

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

Contains the triangular (or trapezoidal) matrix described in Section 3.

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

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

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

contains the triangular matrix described in Section 3.

3: **k – int32 scalar**

4: **l – int32 scalar**

k and **l** specify the dimension of the subblocks k and l as described in Section 3; $(k + l)$ is the effective numerical rank of $\begin{pmatrix} \mathbf{a}^{\mathrm{T}} & \mathbf{b}^{\mathrm{T}} \end{pmatrix}^{\mathrm{T}}$.

5: **u**(**ldu**,$*$) – **complex array**

The first dimension, **ldu**, of the array **u** must satisfy

if **jobu** = 'U', **ldu** $\geq \max(1, \mathbf{m})$;
ldu ≥ 1 otherwise.

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

If **jobu** = 'U', **u** contains the unitary matrix U.

If **jobu** = 'N', **u** is not referenced.

6: **v**(**ldv**,$*$) – **complex array**

The first dimension, **ldv**, of the array **v** must satisfy

if **jobv** = 'V', **ldv** $\geq \max(1, \mathbf{p})$;
ldv ≥ 1 otherwise.

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

If **jobv** = 'V', **v** contains the unitary matrix V.

If **jobv** = 'N', **v** is not referenced.

7: **q**(**ldq**,$*$) – **complex array**

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

if **jobq** = 'Q', **ldq** $\geq \max(1, \mathbf{n})$;
ldq ≥ 1 otherwise.

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

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

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

8: **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: **jobu**, 2: **jobv**, 3: **jobq**, 4: **m**, 5: **p**, 6: **n**, 7: **a**, 8: **lda**, 9: **b**, 10: **ldb**, 11: **tola**, 12: **tolb**, 13: **k**, 14: **l**, 15: **u**, 16: **ldu**, 17: **v**, 18: **ldv**, 19: **q**, 20: **ldq**, 21: **iwork**, 22: **rwork**, 23: **tau**, 24: **work**, 25: **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 computed factorization is nearly the exact factorization for nearby matrices $(A + E)$ and $(B + F)$, where

$$\|E\|_2 = O(\epsilon)\|A\|_2 \qquad \text{and} \qquad \|F\|_2 = O(\epsilon)\|B\|_2,$$

and ϵ is the ***machine precision***.

8 Further Comments

The real analogue of this function is f08ve.

9 Example

```
jobu = 'U';
jobv = 'V';
jobq = 'Q';
a = [complex(0.96, -0.8100000000000001), complex(-0.03, +0.96), complex(-
0.91, +2.06), complex(-0.05, +0.41);
      complex(-0.98, +1.98), complex(-1.2, +0.19), complex(-0.66, +0.42),
...
     complex(-0.8100000000000001, +0.5600000000000001);
       complex(0.62, -0.46), complex(1.01, +0.02), complex(0.63, -0.17),
complex(-1.11, +0.6);
      complex(0.37, +0.38), complex(0.19, -0.54), complex(-0.98, -0.36),
complex(0.22, -0.2);
       complex(0.83, +0.51), complex(0.2, +0.01), complex(-0.17, -0.46),
complex(1.47, +1.59);
                complex(1.08,   -0.28),   complex(0.2,   -0.12),   complex(-
0.07000000000000001, +1.23), complex(0.26, +0.26)];
b = [complex(1, +0), complex(0, +0), complex(-1, +0), complex(0, +0);
     complex(0, +0), complex(1, +0), complex(0, +0), complex(-1, +0)];
tola = 4.572525929912148e-15;
tolb = 4.445228907190568e-16;
[aOut, bOut, k, l, u, v, q, info] = f08vs(jobu, jobv, jobq, a, b, tola,
tolb)
```

```
aOut =
   -2.7118              -1.4390 - 1.0315i   -0.1054 + 1.3176i   -0.3924 -
0.1950i
         0              -1.8583             -0.9453 + 0.1928i    1.4355 +
0.2631i
         0                    0              2.9079             -0.2395 +
```

```
0.1886i
        0                   0                   0            -1.5759
        0                   0                   0                 0
        0                   0                   0                 0
bOut =
         0         0    1.4142         0
         0         0         0    1.4142
k =
           2
l =
           2
u =
  Columns 1 through 4
   -0.0130 - 0.3259i   -0.1404 - 0.2617i    0.2436 - 0.7796i   -0.0740 -
0.2782i
    0.4276 - 0.6258i    0.0863 - 0.0382i   -0.3203 + 0.1448i    0.1074 +
0.1882i
   -0.3259 + 0.1643i    0.3816 - 0.1822i    0.1722 - 0.0014i   -0.4977 +
0.1783i
    0.1591 - 0.0052i   -0.2821 + 0.1973i    0.2531 + 0.1905i   -0.3779 +
0.2682i
   -0.1721 - 0.0130i   -0.5094 - 0.5032i    0.0321 + 0.1836i    0.2042 +
0.1660i
   -0.2634 - 0.2477i   -0.1086 + 0.2847i    0.1414 - 0.1571i   -0.0873 +
0.5468i
  Columns 5 through 6
  -0.0459 + 0.0001i  -0.0528 - 0.2249i
  -0.0803 - 0.4360i  -0.0381 - 0.2191i
   0.0597 - 0.5897i  -0.1385 - 0.0909i
  -0.0464 + 0.3086i  -0.3735 - 0.5515i
   0.5784 - 0.1244i  -0.0188 - 0.0557i
   0.0158 + 0.0471i   0.6501 + 0.0049i
v =
     1     0     0     0     0     0
     0     1     0     0     0     0
q =
    0.7071         0    0.7071         0
         0    0.7071         0    0.7071
    0.7071         0   -0.7071         0
         0    0.7071         0   -0.7071
info =
           0
```