NAG Toolbox for MATLAB

f08vn

1 Purpose

f08vn computes the generalized singular value decomposition (GSVD) of an m by n complex matrix A and a p by n complex matrix B.

2 Syntax

```
[k, l, a, b, alpha, beta, u, v, q, info] = f08vn(jobu, jobv, jobq, a, b,
'm', m, 'n', n, 'p', p)
```

3 Description

The generalized singular value decomposition is given by

$$U^{\mathrm{H}}AQ = D_1(\,0 \quad R\,), \qquad V^{\mathrm{H}}BQ = D_2(\,0 \quad R\,)$$

where U, V and Q are unitary matrices. Let $(k+l)$ be the effective numerical rank of the matrix $\begin{pmatrix} A^{\mathrm{H}} & B^{\mathrm{H}} \end{pmatrix}^{\mathrm{H}}$, then R is a $(k+l)$ by $(k+l)$ nonsingular upper triangular matrix, D_1 and D_2 are m by $(k+l)$ and P by $(k+l)$ 'diagonal' matrices structured as follows:

if $m-k-l \geq 0$,

$$D_1 = \begin{array}{c c} & \begin{array}{cc} k & l \end{array} \\ \begin{array}{r} k \\ l \\ m-k-l \end{array} & \begin{pmatrix} I & 0 \\ 0 & C \\ 0 & 0 \end{pmatrix} \end{array}$$

$$D_2 = \begin{array}{c c} & \begin{array}{cc} k & l \end{array} \\ \begin{array}{r} l \\ p-l \end{array} & \begin{pmatrix} 0 & S \\ 0 & 0 \end{pmatrix} \end{array}$$

$$(\,0 \quad R\,) = \begin{array}{c c} & \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{array}{r} k \\ l \end{array} & \begin{pmatrix} 0 & R_{11} & R_{12} \\ 0 & 0 & R_{22} \end{pmatrix} \end{array}$$

where

$$C = \mathrm{diag}(\alpha_{k+1}, \ldots, \alpha_{k+l}),$$

$$S = \mathrm{diag}(\beta_{k+1}, \ldots, \beta_{k+l}),$$

and

$$C^2 + S^2 = I.$$

R is stored in $\mathbf{a}(1:k+l, n-k-l+1:n)$ on exit.

If $m-k-l < 0$,

$$D_1 = \begin{array}{c c} & \begin{array}{ccc} k & m-k & k+l-m \end{array} \\ \begin{array}{r} k \\ m-k \end{array} & \begin{pmatrix} I & 0 & 0 \\ 0 & C & 0 \end{pmatrix} \end{array}$$

$$D_2 = \begin{array}{c} \\ m-k \\ k+l-m \\ p-l \end{array}\begin{array}{c} \begin{array}{ccc} k & m-k & k+l-m \end{array} \\ \begin{pmatrix} 0 & S & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$\begin{pmatrix} 0 & R \end{pmatrix} = \begin{array}{c} \\ k \\ m-k \\ k+l-m \end{array}\begin{array}{c} \begin{array}{cccc} n-k-l & k & m-k & k+l-m \end{array} \\ \begin{pmatrix} 0 & R_{11} & R_{12} & R_{13} \\ 0 & 0 & R_{22} & R_{23} \\ 0 & 0 & 0 & R_{33} \end{pmatrix} \end{array}$$

where

$$C = \mathrm{diag}(\alpha_{k+1}, \ldots, \alpha_m),$$

$$S = \mathrm{diag}(\beta_{k+1}, \ldots, \beta_m),$$

and

$$C^2 + S^2 = I.$$

$\begin{pmatrix} R_{11} & R_{12} & R_{13} \end{pmatrix}$ is stored in $\mathbf{a}(1:m, n-k-l+1:n)$ and R_{33} is stored $\begin{pmatrix} 0 & R_{22} & R_{23} \end{pmatrix}$ in $\mathbf{b}(m-k+1:l, n+m-k-l+1:n)$ on exit.

The function computes C, S, R and, optionally, the unitary transformation matrices U, V and Q.

In particular, if B is an n by n nonsingular matrix, then the GSVD of A and B implicitly gives the SVD of $A \times B^{-1}$:

$$AB^{-1} = U\left(D_1 D_2^{-1}\right)V^{\mathrm{H}}.$$

If $\begin{pmatrix} A^{\mathrm{H}} & B^{\mathrm{H}} \end{pmatrix}^{\mathrm{H}}$ has orthonormal columns, then the GSVD of A and B is also equal to the CS decomposition of A and B. Furthermore, the GSVD can be used to derive the solution of the eigenvalue problem:

$$A^{\mathrm{H}}Ax = \lambda B^{\mathrm{H}}Bx.$$

In some literature, the GSVD of A and B is presented in the form

$$U^{\mathrm{H}}AX = \begin{pmatrix} 0 & D_1 \end{pmatrix}, \quad V^{\mathrm{H}}BX = \begin{pmatrix} 0 & D_2 \end{pmatrix},$$

where U and V are orthogonal and X is nonsingular, and D_1 and D_2 are 'diagonal'. The former GSVD form can be converted to the latter form by taking the nonsingular matrix X as

$$X = Q\begin{pmatrix} I & 0 \\ 0 & R^{-1} \end{pmatrix}$$

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 unitary 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.

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: **n – int32 scalar**

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

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

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

3: **p – int32 scalar**

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

p, the number of rows of the matrix B.

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

5.3 Input Parameters Omitted from the MATLAB Interface

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

5.4 Output Parameters

1: **k – int32 scalar**

2: **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 $\left(\mathbf{a}^{\mathrm{T}} \quad \mathbf{b}^{\mathrm{T}} \right)^{\mathrm{T}}$.

3: **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 matrix R, or part of R. See Section 3 for details.

4: **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 R if $m-k-l<0$. See Section 3 for details.

5: **alpha($*$) – double array**

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

See the description of **beta**.

6: **beta($*$) – double array**

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

alpha and **beta** contain the generalized singular value pairs of A and B, α_i and β_i;

$\mathbf{alpha}(1:\mathbf{k})=1$,

$\mathbf{beta}(1:\mathbf{k})=0$,

and if $m-k-l \geq 0$,

$\mathbf{alpha}(\mathbf{k}+1:\mathbf{k}+\mathbf{l})=C$,

$\mathbf{beta}(\mathbf{k}+1:\mathbf{k}+\mathbf{l})=S$,

or if $m-k-l<0$,

$\mathbf{alpha}(\mathbf{k}+1:\mathbf{m})=C$,

$\mathbf{alpha}(\mathbf{m}+1:\mathbf{k}+\mathbf{l})=0$,

$\mathbf{beta}(\mathbf{k}+1:\mathbf{m})=S$,

$\mathbf{beta}(\mathbf{m}+1:\mathbf{k}+\mathbf{l})=1$, and

$\mathbf{alpha}(\mathbf{k}+\mathbf{l}+1:\mathbf{n})=0$,

$\mathbf{beta}(\mathbf{k}+\mathbf{l}+1:\mathbf{n})=0$.

7: **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 m by m unitary matrix U.

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

8: **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 $\max(1, \mathbf{p})$

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

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

9: **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 n by n unitary matrix Q.

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

10: **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: **n**, 6: **p**, 7: **k**, 8: **l**, 9: **a**, 10: **lda**, 11: **b**, 12: **ldb**, 13: **alpha**, 14: **beta**, 15: **u**, 16: **ldu**, 17: **v**, 18: **ldv**, 19: **q**, 20: **ldq**, 21: **work**, 22: **rwork**, 23: **iwork**, 24: **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 = 1

If **info** = 1, the Jacobi-type procedure failed to converge.

7 Accuracy

The computed generalized singular value decomposition is nearly the exact generalized singular value decomposition for nearby matrices $(A + E)$ and $(B + F)$, where

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

and ϵ is the ***machine precision***. See Section 4.12 of Anderson *et al.* 1999 for further details.

8 Further Comments

The diagonal elements of the matrix R are real.

The real analogue of this function is f08va.

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)];
[k, l, aOut, bOut, alpha, beta, u, v, q, info] = f08vn(jobu, jobv, jobq,
a, b)
```

```
k =
           2
l =
           2
aOut =
  -2.7118              -1.4390 - 1.0315i  -0.0769 + 1.3613i  -0.2814 -
0.0324i
         0            -1.8583             -1.0760 + 0.0310i   1.3292 +
0.3677i
         0                  0              3.2537             -0.0000 +
0.0000i
        0                  0                  0            -2.1084
        0                  0                  0                  0
        0                  0                  0                  0
bOut =
         0                  0              1.4142             -0.0000 +
0.0000i
        0                  0                  0            -1.4142
alpha =
    1.0000
    1.0000
    0.9006
    0.7417
beta =
         0
         0
    0.4346
    0.6707
u =
  Columns 1 through 4
   -0.0130 - 0.3259i  -0.1404 - 0.2617i   0.2518 - 0.7979i  -0.0510 -
0.2175i
    0.4276 - 0.6258i   0.0863 - 0.0382i  -0.3219 + 0.1611i   0.1198 +
0.1632i
   -0.3259 + 0.1643i   0.3816 - 0.1822i   0.1323 - 0.0146i  -0.5067 +
0.1862i
    0.1591 - 0.0052i  -0.2821 + 0.1973i   0.2160 + 0.1881i  -0.4016 +
0.2679i
   -0.1721 - 0.0130i  -0.5094 - 0.5032i   0.0365 + 0.2032i   0.1927 +
0.1557i
   -0.2634 - 0.2477i  -0.1086 + 0.2847i   0.1091 - 0.1271i  -0.0882 +
0.5617i
  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 =
   0.9893 + 0.0000i  -0.1146 + 0.0903i
  -0.1146 - 0.0903i  -0.9893 + 0.0000i
q =
    0.7071                   0                0.6995 + 0.0000i   0.0810 -
0.0638i
         0              0.7071               -0.0810 - 0.0638i   0.6995 -
0.0000i
    0.7071                   0               -0.6995 - 0.0000i  -0.0810 +
0.0638i
         0              0.7071                0.0810 + 0.0638i  -0.6995 +
0.0000i
info =
           0
```