NAG Toolbox

nag_lapack_ztgsja (f08ys)

1 Purpose

nag_lapack_ztgsja (f08ys) computes the generalized singular value decomposition (GSVD) of two complex upper trapezoidal matrices A and B, where A is an m by n matrix and B is a p by n matrix. A and B are assumed to be in the form returned by nag_lapack_zggsvp (f08vs).

2 Syntax

```
[a, b, alpha, beta, u, v, q, ncycle, info] = nag_lapack_ztgsja(jobu, jobv, jobq, k, l, a, b, tola, tolb, u, v, q, 'm', m, 'p', p, 'n', n)
[a, b, alpha, beta, u, v, q, ncycle, info] = f08ys(jobu, jobv, jobq, k, l, a, b, tola, tolb, u, v, q, 'm', m, 'p', p, 'n', n)
```

3 Description

nag_lapack_ztgsja (f08ys) computes the GSVD of the matrices A and B which are assumed to have the form as returned by nag_lapack_zggsvp (f08vs)

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

$$B = \begin{array}{r} \\ l \\ p-l \end{array} \begin{array}{c} \begin{array}{ccc} n-k-l & k & l \end{array} \\ \begin{pmatrix} 0 & 0 & B_{13} \\ 0 & 0 & 0 \end{pmatrix} \end{array},$$

where the k by k matrix A_{12} and the 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.

nag_lapack_ztgsja (f08ys) computes unitary matrices Q, U and V, diagonal matrices D_1 and D_2, and an upper triangular matrix R such that

$$U^{\mathrm{H}}AQ = D_1\begin{pmatrix} 0 & R \end{pmatrix}, \quad V^{\mathrm{H}}BQ = D_2\begin{pmatrix} 0 & R \end{pmatrix}.$$

Optionally Q, U and V may or may not be computed, or they may be premultiplied by matrices Q_1, U_1 and V_1 respectively.

If $(m-k-l) \geq 0$ then D_1, D_2 and R have the form

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

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

$$R = \begin{array}{c} \\ k \\ l \end{array} \begin{array}{c} \begin{array}{cc} k & l \end{array} \\ \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \end{array},$$

where $C = \text{diag}(\alpha_{k+1}, , , \ldots, , , \alpha_{k+l})$, $\quad S = \text{diag}(\beta_{k+1}, , , \ldots, , , \beta_{k+l})$.

If $(m-k-l) < 0$ then D_1, D_2 and R have the form

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

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

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

where $C = \text{diag}(\alpha_{k+1}, , , \ldots, , , \alpha_m)$, $\quad S = \text{diag}(\beta_{k+1}, , , \ldots, , , \beta_m)$.

In both cases the diagonal matrix C has real non-negative diagonal elements, the diagonal matrix S has real positive diagonal elements, so that S is nonsingular, and $C^2 + S^2 = 1$. See Section 2.3.5.3 of Anderson *et al.* (1999) for further information.

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 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** – CHARACTER(1)

If **jobu** = 'U', **u** must contain a unitary matrix U_1 on entry, and the product U_1U is returned.

If **jobu** = 'I', **u** is initialized to the unit matrix, and the unitary matrix U is returned.

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

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

2: **jobv** – CHARACTER(1)

If **jobv** = 'V', **v** must contain a unitary matrix V_1 on entry, and the product V_1V is returned.

If **jobv** = 'I', **v** is initialized to the unit matrix, and the unitary matrix V is returned.

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

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

3: **jobq** – CHARACTER(1)

If **jobq** = 'Q', **q** must contain a unitary matrix Q_1 on entry, and the product $Q_1 Q$ is returned.

If **jobq** = 'I', **q** is initialized to the unit matrix, and the unitary matrix Q is returned.

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

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

4: **k** – INTEGER

5: **l** – INTEGER

k and **l** specify the sizes, k and l, of the subblocks of A and B, whose GSVD is to be computed by nag_lapack_ztgsja (f08ys).

6: **a**$(lda, :)$ – COMPLEX (KIND=nag_wp) array

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

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

The m by n matrix A.

7: **b**$(ldb, :)$ – COMPLEX (KIND=nag_wp) array

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

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

The p by n matrix B.

8: **tola** – REAL (KIND=nag_wp)

9: **tolb** – REAL (KIND=nag_wp)

tola and **tolb** are the convergence criteria for the Jacobi–Kogbetliantz iteration procedure. Generally, they should be the same as used in the preprocessing step performed by nag_lapack_zggsvp (f08vs), say

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

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

10: **u**$(ldu, :)$ – COMPLEX (KIND=nag_wp) array

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

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

The second dimension of the array **u** must be at least $\max(1, \mathbf{m})$ if **jobu** = 'U' or 'I', and at least 1 otherwise.

If **jobu** = 'U', **u** must contain an m by m matrix U_1 (usually the unitary matrix returned by nag_lapack_zggsvp (f08vs)).

11: **v**$(ldv,:)$ – COMPLEX (KIND=nag_wp) array

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

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

The second dimension of the array **v** must be at least $\max(1,\mathbf{p})$ if **jobv** = 'V' or 'I', and at least 1 otherwise.

If **jobv** = 'V', **v** must contain an p by p matrix V_1 (usually the unitary matrix returned by nag_lapack_zggsvp (f08vs)).

12: **q**$(ldq,:)$ – COMPLEX (KIND=nag_wp) array

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

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

The second dimension of the array **q** must be at least $\max(1,\mathbf{n})$ if **jobq** = 'Q' or 'I', and at least 1 otherwise.

If **jobq** = 'Q', **q** must contain an n by n matrix Q_1 (usually the unitary matrix returned by nag_lapack_zggsvp (f08vs)).

5.2 Optional Input Parameters

1: **m** – INTEGER

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

m, the number of rows of the matrix A.

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

2: **p** – INTEGER

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

p, the number of rows of the matrix B.

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

3: **n** – INTEGER

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 Output Parameters

1: **a**$(lda,:)$ – COMPLEX (KIND=nag_wp) array

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

The second dimension of the array **a** will be $\max(1,\mathbf{n})$.

If $m-k-l \geq 0$, $\mathbf{a}(1:k+l, n-k-l+1:n)$ contains the $(k+l)$ by $(k+l)$ upper triangular matrix R.

If $m-k-l < 0$, $\mathbf{a}(1:m, n-k-l+1:n)$ contains the first m rows of the $(k+l)$ by $(k+l)$ upper triangular matrix R, and the submatrix R_{33} is returned in $\mathbf{b}(m-k+1:l, n+m-k-l+1:n)$.

2: **b**$(ldb,:)$ – COMPLEX (KIND=nag_wp) array

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

The second dimension of the array **b** will be $\max(1,\mathbf{n})$.

If $m-k-l<0$, **b**$(m-k+1:l, n+m-k-l+1:n)$ contains the submatrix R_{33} of R.

3: **alpha**(**n**) – REAL (KIND=nag_wp) array

See the description of **beta**.

4: **beta**(**n**) – REAL (KIND=nag_wp) array

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

alpha$(i)=1$, **beta**$(i)=0$, for $i=1,2,\ldots,k$, and

if $m-k-l\geq 0$, **alpha**$(i)=\alpha_i$, **beta**$(i)=\beta_i$, for $i=k+1,\ldots,k+l$, or

if $m-k-l<0$, **alpha**$(i)=\alpha_i$, **beta**$(i)=\beta_i$, for $i=k+1,\ldots,m$ and **alpha**$(i)=0$, **beta**$(i)=1$, for $i=m+1,\ldots,k+l$.

Furthermore, if $k+l<n$, **alpha**$(i)=$ **beta**$(i)=0$, for $i=k+l+1,\ldots,n$.

5: **u**$(ldu,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension, ldu, of the array **u** will be

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

The second dimension of the array **u** will be $\max(1,\mathbf{m})$ if **jobu** = 'U' or 'I' and 1 otherwise.

If **jobu** = 'U', **u** contains the product U_1U.

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

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

6: **v**$(ldv,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension, ldv, of the array **v** will be

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

The second dimension of the array **v** will be $\max(1,\mathbf{p})$ if **jobv** = 'V' or 'I' and 1 otherwise.

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

If **jobv** = 'V', **v** contains the product V_1V.

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

7: **q**$(ldq,:)$ – COMPLEX (KIND=nag_wp) array

The first dimension, ldq, of the array **q** will be

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

The second dimension of the array **q** will be $\max(1,\mathbf{n})$ if **jobq** = 'Q' or 'I' and 1 otherwise.

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

If **jobq** = 'Q', **q** contains the product Q_1Q.

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

8: **ncycle** – INTEGER

The number of cycles required for convergence.

9: **info** – INTEGER

info $= 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

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: **k**, 8: **l**, 9: **a**, 10: **lda**, 11: **b**, 12: **ldb**, 13: **tola**, 14: **tolb**, 15: **alpha**, 16: **beta**, 17: **u**, 18: **ldu**, 19: **v**, 20: **ldv**, 21: **q**, 22: **ldq**, 23: **work**, 24: **ncycle**, 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.

info $= 1$

The procedure does not converge after 40 cycles.

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

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

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

8 Further Comments

The real analogue of this function is nag_lapack_dtgsja (f08ye).

9 Example

This example finds the generalized singular value decomposition

$$A = U\Sigma_1 \begin{pmatrix} 0 & R \end{pmatrix} Q^{\mathrm{H}}, \quad B = V\Sigma_2 \begin{pmatrix} 0 & R \end{pmatrix} Q^{\mathrm{H}},$$

of the matrix pair (A, B), where

$$A = \begin{pmatrix} 0.96 - 0.81i & -0.03 + 0.96i & -0.91 + 2.06i & -0.05 + 0.41i \\ -0.98 + 1.98i & -1.20 + 0.19i & -0.66 + 0.42i & -0.81 + 0.56i \\ 0.62 - 0.46i & 1.01 + 0.02i & 0.63 - 0.17i & -1.11 + 0.60i \\ 0.37 + 0.38i & 0.19 - 0.54i & -0.98 - 0.36i & 0.22 - 0.20i \\ 0.83 + 0.51i & 0.20 + 0.01i & -0.17 - 0.46i & 1.47 + 1.59i \\ 1.08 - 0.28i & 0.20 - 0.12i & -0.07 + 1.23i & 0.26 + 0.26i \end{pmatrix}$$

and

$$B = \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \end{pmatrix}.$$

9.1 Program Text

```
     function f08ys_example

fprintf('f08ys example results\n\n');

% Compute generalized SVD of (A,B)
A = [ 0.96-0.81i -0.03+0.96i -0.91+2.06i -0.05+0.41i;
     -0.98+1.98i -1.20+0.19i -0.66+0.42i -0.81+0.56i;
      0.62-0.46i  1.01+0.02i  0.63-0.17i -1.11+0.60i;
      0.37+0.38i  0.19-0.54i -0.98-0.36i  0.22-0.20i;
      0.83+0.51i  0.20+0.01i -0.17-0.46i  1.47+1.59i;
      1.08-0.28i  0.20-0.12i -0.07+1.23i  0.26+0.26i];
B = complex([ 1 0 -1  0;
              0 1  0 -1]);
[m, n] = size(A);
p      = size(B,1);

% Reduce A and B to upper triangular form S = U'H A Q, T = V'H B Q
tola = max(m,n)*norm(A,1)*x02aj;
tolb = max(p,n)*norm(B,1)*x02aj;
[S, T, k, l, U, V, Q, info] = ...
  f08vs(...
        'U', 'V', 'Q', A, B, tola, tolb);

% Compute singular values
[S, T, alpha, beta, U, V, Q, ncycle, info] = ...
  f08ys('U', 'V', 'Q', k, l, S, T, tola, tolb, U, V, Q);

fprintf('Number of infinite generalized singular values = %3d\n',k);
fprintf('Number of   finite generalized singular values = %3d\n',l);
fprintf('Effective rank of the matrix pair  (A'T B'T)'T = %3d\n',k+l);
fprintf('Number of cycles of the Kogbetliantz method    = %3d\n\n',ncycle);
disp('Finite generalized singular values');
disp(alpha(k+1:k+l)./beta(k+1:k+l));
```

9.2 Program Results

```
     f08ys example results

Number of infinite generalized singular values =   2
Number of   finite generalized singular values =   2
Effective rank of the matrix pair  (A'T B'T)'T =   4
Number of cycles of the Kogbetliantz method    =   2

Finite generalized singular values
    2.0720
    1.1058
```