NAG Library Routine Document

F08VSF (ZGGSVP)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F08VSF (ZGGSVP) 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 Specification

```
SUBROUTINE F08VSF (JOBU, JOBV, JOBQ, M, P, N, A, LDA, B, LDB, TOLA, TOLB,       &
                   K, L, U, LDU, V, LDV, Q, LDQ, IWORK, RWORK, TAU, WORK,       &
                   INFO)

INTEGER                 M, P, N, LDA, LDB, K, L, LDU, LDV, LDQ, IWORK(N),       &
                        INFO
REAL (KIND=nag_wp)      TOLA, TOLB, RWORK(2*N)
COMPLEX (KIND=nag_wp)   A(LDA,*), B(LDB,*), U(LDU,*), V(LDV,*), Q(LDQ,*),       &
                        TAU(N), WORK(max(3*N,M,P))
CHARACTER(1)            JOBU, JOBV, JOBQ
```

The routine may be called by its LAPACK name ***zggsvp***.

3 Description

F08VSF (ZGGSVP) 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 routine F08VNF (ZGGSVD).

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

1: JOBU – CHARACTER(1) *Input*

On entry: if JOBU = 'U', the unitary matrix U is computed.

If JOBU = 'N', U is not computed.

Constraint: JOBU = 'U' or 'N'.

2: JOBV – CHARACTER(1) *Input*

On entry: if JOBV = 'V', the unitary matrix V is computed.

If JOBV = 'N', V is not computed.

Constraint: JOBV = 'V' or 'N'.

3: JOBQ – CHARACTER(1) *Input*

On entry: if JOBQ = 'Q', the unitary matrix Q is computed.

If JOBQ = 'N', Q is not computed.

Constraint: JOBQ = 'Q' or 'N'.

4: M – INTEGER *Input*

On entry: m, the number of rows of the matrix A.

Constraint: M $\geq$ 0.

5: P – INTEGER *Input*

On entry: p, the number of rows of the matrix B.

Constraint: P $\geq$ 0.

6: N – INTEGER *Input*

On entry: n, the number of columns of the matrices A and B.

Constraint: N $\geq$ 0.

7: A(LDA,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array A must be at least $\max(1,\mathrm{N})$.

On entry: the m by n matrix A.

On exit: contains the triangular (or trapezoidal) matrix described in Section 3.

8: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08VSF (ZGGSVP) is called.

Constraint: LDA $\geq \max(1,\mathrm{M})$.

9: B(LDB,$*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array B must be at least $\max(1, \mathrm{N})$.

On entry: the p by n matrix B.

On exit: contains the triangular matrix described in Section 3.

10: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08VSF (ZGGSVP) is called.

Constraint: $\mathrm{LDB} \geq \max(1, \mathrm{P})$.

11: TOLA – REAL (KIND=nag_wp) *Input*
12: TOLB – REAL (KIND=nag_wp) *Input*

On entry: 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

$$\mathrm{TOLA} = \max(\mathrm{M}, \mathrm{N})\|A\|\epsilon,$$
$$\mathrm{TOLB} = \max(\mathrm{P}, \mathrm{N})\|B\|\epsilon,$$

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

The size of TOLA and TOLB may affect the size of backward errors of the decomposition.

13: K – INTEGER *Output*
14: L – INTEGER *Output*

On exit: 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 $\begin{pmatrix} \mathrm{A}^\mathrm{T} & \mathrm{B}^\mathrm{T} \end{pmatrix}^\mathrm{T}$.

15: U(LDU,$*$) – COMPLEX (KIND=nag_wp) array *Output*

Note: the second dimension of the array U must be at least $\max(1, \mathrm{M})$ if JOBU = 'U', and at least 1 otherwise.

On exit: if JOBU = 'U', U contains the unitary matrix U.

If JOBU = 'N', U is not referenced.

16: LDU – INTEGER *Input*

On entry: the first dimension of the array U as declared in the (sub)program from which F08VSF (ZGGSVP) is called.

Constraints:

if JOBU = 'U', $\mathrm{LDU} \geq \max(1, \mathrm{M})$;
otherwise $\mathrm{LDU} \geq 1$.

17: V(LDV,$*$) – COMPLEX (KIND=nag_wp) array *Output*

Note: the second dimension of the array V must be at least $\max(1, \mathrm{P})$ if JOBV = 'V', and at least 1 otherwise.

On exit: if JOBV = 'V', V contains the unitary matrix V.

If JOBV = 'N', V is not referenced.

18: LDV – INTEGER *Input*

On entry: the first dimension of the array V as declared in the (sub)program from which F08VSF (ZGGSVP) is called.

Constraints:

if JOBV = 'V', $\text{LDV} \geq \max(1, \text{P})$;
otherwise $\text{LDV} \geq 1$.

19: Q(LDQ,$*$) – COMPLEX (KIND=nag_wp) array *Output*

Note: the second dimension of the array Q must be at least $\max(1, \text{N})$ if JOBQ = 'Q', and at least 1 otherwise.

On exit: if JOBQ = 'Q', Q contains the unitary matrix Q.

If JOBQ = 'N', Q is not referenced.

20: LDQ – INTEGER *Input*

On entry: the first dimension of the array Q as declared in the (sub)program from which F08VSF (ZGGSVP) is called.

Constraints:

if JOBQ = 'Q', $\text{LDQ} \geq \max(1, \text{N})$;
otherwise $\text{LDQ} \geq 1$.

21: IWORK(N) – INTEGER array *Workspace*

22: RWORK($2 \times \text{N}$) – REAL (KIND=nag_wp) array *Workspace*

23: TAU(N) – COMPLEX (KIND=nag_wp) array *Workspace*

24: WORK($\max(3 \times \text{N}, \text{M}, \text{P})$) – COMPLEX (KIND=nag_wp) array *Workspace*

25: INFO – INTEGER *Output*

On exit: INFO = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO $= -i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

7 Accuracy

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

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

and ϵ is the ***machine precision***.

8 Further Comments

The real analogue of this routine is F08VEF (DGGSVP).

9 Example

This example finds the generalized factorization

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

of the matrix pair $\begin{pmatrix} A & B \end{pmatrix}$, 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

```
    Program f08vsfe

!     F08VSF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: f06uaf, nag_wp, x02ajf, x04dbf, zggsvp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: eps, tola, tolb
      Integer                          :: i, ifail, info, irank, k, l, lda,    &
                                          ldb, ldq, ldu, ldv, m, n, p
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), q(:,:), tau(:),    &
                                            u(:,:), v(:,:), work(:)
      Real (Kind=nag_wp), Allocatable  :: rwork(:)
      Integer, Allocatable             :: iwork(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, real
!     .. Executable Statements ..
      Write (nout,*) 'F08VSF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) m, n, p
      lda = m
      ldb = p
      ldq = n
      ldu = m
      ldv = p
      Allocate (a(lda,n),b(ldb,n),q(ldq,n),tau(n),u(ldu,m),v(ldv,p), &
        work(m+3*n+p),rwork(2*n),iwork(n))

!     Read the m by n matrix A and p by n matrix B from data file

      Read (nin,*)(a(i,1:n),i=1,m)
      Read (nin,*)(b(i,1:n),i=1,p)

!     Compute tola and tolb as
!         tola = max(m,n)*norm(A)*macheps
!         tolb = max(p,n)*norm(B)*macheps

      eps = x02ajf()
      tola = real(max(m,n),kind=nag_wp)*f06uaf('One-norm',m,n,a,lda,rwork)*eps
      tolb = real(max(p,n),kind=nag_wp)*f06uaf('One-norm',p,n,b,ldb,rwork)*eps
```

```
!     Compute the factorization of (A, B)
!         (A = U*S*(Q**H), B = V*T*(Q**H))

!     The NAG name equivalent of zggsvp is f08vsf
      Call zggsvp('U','V','Q',m,p,n,a,lda,b,ldb,tola,tolb,k,l,u,ldu,v,ldv,q, &
        ldq,iwork,rwork,tau,work,info)

!     Print solution

      irank = k + l
      Write (nout,*) 'Numerical rank of (A**T B**T)**T (K+L)'
      Write (nout,99999) irank

      Write (nout,*)
      Flush (nout)
      If (m>=irank) Then

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04dbf('Upper','Non-unit',irank,irank,a(1,n-irank+1),lda, &
          'Bracketed','1P,E12.4','Upper triangular matrix S','Integer',rlabs, &
          'Integer',clabs,80,0,ifail)

      Else

        ifail = 0
        Call x04dbf('Upper','Non-unit',m,irank,a(1,n-irank+1),lda,'Bracketed', &
          '1P,E12.4','Upper trapezoidal matrix S','Integer',rlabs,'Integer', &
          clabs,80,0,ifail)

      End If
      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04dbf('Upper','Non-unit',l,l,b(1,n-l+1),ldb,'Bracketed', &
        '1P,E12.4','Upper triangular matrix T','Integer',rlabs,'Integer', &
        clabs,80,0,ifail)

      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04dbf('General',' ',m,m,u,ldu,'Bracketed','1P,E12.4', &
        'Orthogonal matrix U','Integer',rlabs,'Integer',clabs,80,0,ifail)

      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04dbf('General',' ',p,p,v,ldv,'Bracketed','1P,E12.4', &
        'Orthogonal matrix V','Integer',rlabs,'Integer',clabs,80,0,ifail)

      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04dbf('General',' ',n,n,q,ldq,'Bracketed','1P,E12.4', &
        'Orthogonal matrix Q','Integer',rlabs,'Integer',clabs,80,0,ifail)

99999 Format (1X,I5)
    End Program f08vsfe
```

9.2 Program Data

```
F08VSF Example Program Data

  6           4           2                           :Values of M, N and P

( 0.96,-0.81) (-0.03, 0.96) (-0.91, 2.06) (-0.05, 0.41)
(-0.98, 1.98) (-1.20, 0.19) (-0.66, 0.42) (-0.81, 0.56)
( 0.62,-0.46) ( 1.01, 0.02) ( 0.63,-0.17) (-1.11, 0.60)
( 0.37, 0.38) ( 0.19,-0.54) (-0.98,-0.36) ( 0.22,-0.20)
( 0.83, 0.51) ( 0.20, 0.01) (-0.17,-0.46) ( 1.47, 1.59)
( 1.08,-0.28) ( 0.20,-0.12) (-0.07, 1.23) ( 0.26, 0.26) :End of matrix A

( 1.00, 0.00) ( 0.00, 0.00) (-1.00, 0.00) ( 0.00, 0.00)
( 0.00, 0.00) ( 1.00, 0.00) ( 0.00, 0.00) (-1.00, 0.00) :End of matrix B
```

9.3 Program Results

```
 F08VSF Example Program Results

 Numerical rank of (A**T B**T)**T (K+L)
     4

 Upper triangular matrix S
                                1                              2
 1  ( -2.7118E+00,  0.0000E+00) ( -1.4390E+00, -1.0315E+00)
 2                              ( -1.8583E+00,  0.0000E+00)
 3
 4

                                3                              4
 1  ( -1.0543E-01,  1.3176E+00) ( -3.9240E-01, -1.9504E-01)
 2  ( -9.4529E-01,  1.9279E-01) (  1.4355E+00,  2.6313E-01)
 3  (  2.9079E+00,  0.0000E+00) ( -2.3946E-01,  1.8856E-01)
 4                              ( -1.5759E+00,  0.0000E+00)

 Upper triangular matrix T
                                1                              2
 1  (  1.4142E+00,  0.0000E+00) (  0.0000E+00,  0.0000E+00)
 2                              (  1.4142E+00,  0.0000E+00)

 Orthogonal matrix U
                                1                              2
 1  ( -1.3038E-02, -3.2595E-01) ( -1.4039E-01, -2.6167E-01)
 2  (  4.2764E-01, -6.2582E-01) (  8.6298E-02, -3.8174E-02)
 3  ( -3.2595E-01,  1.6428E-01) (  3.8163E-01, -1.8219E-01)
 4  (  1.5906E-01, -5.2151E-03) ( -2.8207E-01,  1.9732E-01)
 5  ( -1.7210E-01, -1.3038E-02) ( -5.0942E-01, -5.0319E-01)
 6  ( -2.6336E-01, -2.4772E-01) ( -1.0861E-01,  2.8474E-01)

                                3                              4
 1  (  2.4357E-01, -7.7956E-01) ( -7.4007E-02, -2.7823E-01)
 2  ( -3.2035E-01,  1.4475E-01) (  1.0740E-01,  1.8824E-01)
 3  (  1.7217E-01, -1.4009E-03) ( -4.9770E-01,  1.7826E-01)
 4  (  2.5307E-01,  1.9053E-01) ( -3.7794E-01,  2.6816E-01)
 5  (  3.2057E-02,  1.8358E-01) (  2.0422E-01,  1.6601E-01)
 6  (  1.4142E-01, -1.5707E-01) ( -8.7335E-02,  5.4683E-01)

                                5                              6
 1  ( -4.5947E-02,  1.4052E-04) ( -5.2773E-02, -2.2492E-01)
 2  ( -8.0311E-02, -4.3605E-01) ( -3.8117E-02, -2.1907E-01)
 3  (  5.9714E-02, -5.8974E-01) ( -1.3850E-01, -9.0941E-02)
 4  ( -4.6443E-02,  3.0864E-01) ( -3.7354E-01, -5.5148E-01)
 5  (  5.7843E-01, -1.2439E-01) ( -1.8815E-02, -5.5686E-02)
 6  (  1.5763E-02,  4.7130E-02) (  6.5007E-01,  4.9173E-03)

 Orthogonal matrix V
                                1                              2
 1  (  1.0000E+00,  0.0000E+00) (  0.0000E+00,  0.0000E+00)
 2  (  0.0000E+00,  0.0000E+00) (  1.0000E+00,  0.0000E+00)
```

```
 Orthogonal matrix Q
                             1                           2
 1  (   7.0711E-01,  0.0000E+00) (   0.0000E+00,  0.0000E+00)
 2  (   0.0000E+00,  0.0000E+00) (   7.0711E-01,  0.0000E+00)
 3  (   7.0711E-01,  0.0000E+00) (   0.0000E+00,  0.0000E+00)
 4  (   0.0000E+00,  0.0000E+00) (   7.0711E-01,  0.0000E+00)

                             3                           4
 1  (   7.0711E-01,  0.0000E+00) (   0.0000E+00,  0.0000E+00)
 2  (   0.0000E+00,  0.0000E+00) (   7.0711E-01,  0.0000E+00)
 3  (  -7.0711E-01,  0.0000E+00) (   0.0000E+00,  0.0000E+00)
 4  (   0.0000E+00,  0.0000E+00) (  -7.0711E-01,  0.0000E+00)
```