NAG Library Routine Document

F08YSF (ZTGSJA)

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

F08YSF (ZTGSJA) 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 F08VSF (ZGGSVP) or F08VUF (ZGGSVP3).

2 Specification

```
SUBROUTINE F08YSF (JOBU, JOBV, JOBQ, M, P, N, K, L, A, LDA, B, LDB,          &
                   TOLA, TOLB, ALPHA, BETA, U, LDU, V, LDV, Q, LDQ,          &
                   WORK, NCYCLE, INFO)
INTEGER               M, P, N, K, L, LDA, LDB, LDU, LDV, LDQ, NCYCLE,       &
                      INFO
REAL (KIND=nag_wp)    TOLA, TOLB, ALPHA(N), BETA(N)
COMPLEX (KIND=nag_wp) A(LDA,*), B(LDB,*), U(LDU,*), V(LDV,*),               &
                      Q(LDQ,*), WORK(2*N)
CHARACTER(1)          JOBU, JOBV, JOBQ
```

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

3 Description

F08YSF (ZTGSJA) computes the GSVD of the matrices A and B which are assumed to have the form as returned by F08VSF (ZGGSVP) or F08VUF (ZGGSVP3)

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

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

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

F08YSF (ZTGSJA) 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

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

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

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

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

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

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

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

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

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

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 Arguments

1: JOBU – CHARACTER(1) *Input*

On entry: 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) *Input*

On entry: if JOBV = 'V', V must contain a unitary matrix V_1 on entry, and the product $V_1 V$ 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) *Input*

On entry: 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: M – INTEGER *Input*

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

Constraint: $\mathrm{M} \geq 0$.

5: P – INTEGER *Input*

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

Constraint: $\mathrm{P} \geq 0$.

6: N – INTEGER *Input*

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

Constraint: $\mathrm{N} \geq 0$.

7: K – INTEGER *Input*
8: L – INTEGER *Input*

On entry: K and L specify the sizes, k and l, of the subblocks of A and B, whose GSVD is to be computed by F08YSF (ZTGSJA).

9: 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: if $m - k - l \geq 0$, $\mathrm{A}(1 : k + l, n - k - l + 1 : n)$ contains the $(k + l)$ by $(k + l)$ upper triangular matrix R.

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

10: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08YSF (ZTGSJA) is called.

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

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

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

On entry: the p by n matrix B.

On exit: if $m - k - l < 0$, $B(m - k + 1 : l, n + m - k - l + 1 : n)$ contains the submatrix R_{33} of R.

12: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08YSF (ZTGSJA) is called.

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

13: TOLA – REAL (KIND=nag_wp) *Input*
14: TOLB – REAL (KIND=nag_wp) *Input*

On entry: 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 F08VSF (ZGGSVP) or F08VUF (ZGGSVP3), say

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

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

15: ALPHA(N) – REAL (KIND=nag_wp) array *Output*

On exit: see the description of BETA.

16: BETA(N) – REAL (KIND=nag_wp) array *Output*

On exit: 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 $i = k + l + 1, \ldots, n$.

17: U(LDU, $*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: if JOBU = 'U', U must contain an m by m matrix U_1 (usually the unitary matrix returned by F08VSF (ZGGSVP) or F08VUF (ZGGSVP3)).

On exit: if JOBU = 'U', U contains the product $U_1 U$.

If JOBU = 'I', U contains the unitary matrix U.

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

18: LDU – INTEGER *Input*

On entry: the first dimension of the array U as declared in the (sub)program from which F08YSF (ZTGSJA) is called.

Constraints:

if JOBU = 'U' or 'I', $LDU \geq \max(1, M)$;
otherwise $LDU \geq 1$.

19: V(LDV, $*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: if JOBV = 'V', V must contain an p by p matrix V_1 (usually the unitary matrix returned by F08VSF (ZGGSVP) or F08VUF (ZGGSVP3)).

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

If JOBV = 'V', V contains the product $V_1 V$.

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

20: LDV – INTEGER *Input*

On entry: the first dimension of the array V as declared in the (sub)program from which F08YSF (ZTGSJA) is called.

Constraints:

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

21: Q(LDQ, $*$) – COMPLEX (KIND=nag_wp) array *Input/Output*

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

On entry: if JOBQ = 'Q', Q must contain an n by n matrix Q_1 (usually the unitary matrix returned by F08VSF (ZGGSVP) or F08VUF (ZGGSVP3)).

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

If JOBQ = 'Q', Q contains the product $Q_1 Q$.

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

22: LDQ – INTEGER *Input*

On entry: the first dimension of the array Q as declared in the (sub)program from which F08YSF (ZTGSJA) is called.

Constraints:

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

23: WORK($2 \times \mathrm{N}$) – COMPLEX (KIND=nag_wp) array *Workspace*

24: NCYCLE – INTEGER *Output*

On exit: the number of cycles required for convergence.

25: INFO – INTEGER *Output*

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

6 Error Indicators and Warnings

INFO < 0

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

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 Parallelism and Performance

F08YSF (ZTGSJA) makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The real analogue of this routine is F08YEF (DTGSJA).

10 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

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

10.1 Program Text

```
    Program f08ysfe

!     F08YSF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f06uaf, nag_wp, x02ajf, x04dbf, zggsvp, ztgsja
!     .. 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, j, k, l, lda, &
```

```
                                    ldb, ldq, ldu, ldv, m, n, ncycle, p
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), q(:,:), tau(:),    &
                                    u(:,:), v(:,:), work(:)
      Real (Kind=nag_wp), Allocatable  :: alpha(:), beta(:), rwork(:)
      Integer, Allocatable             :: iwork(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, real
!     .. Executable Statements ..
      Write (nout,*) 'F08YSF 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),alpha(n),beta(n),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 = U1*S*(Q1**H), B = V1*T*(Q1**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)

!     Compute the generalized singular value decomposition of (A, B)
!         (A = U*D1*(0 R)*(Q**H), B = V*D2*(0 R)*(Q**H))

!     The NAG name equivalent of ztgsja is f08ysf
      Call ztgsja('U','V','Q',m,p,n,k,l,a,lda,b,ldb,tola,tolb,alpha,beta,u,    &
        ldu,v,ldv,q,ldq,work,ncycle,info)

      If (info==0) Then

!       Print solution

        irank = k + l
        Write (nout,*) 'Number of infinite generalized singular values (K)'
        Write (nout,99999) k
        Write (nout,*) 'Number of finite generalized singular values (L)'
        Write (nout,99999) l
        Write (nout,*) ' Effective Numerical rank of (A**T B**T)**T (K+L)'
        Write (nout,99999) irank
        Write (nout,*)
        Write (nout,*) 'Finite generalized singular values'
        Write (nout,99998)(alpha(j)/beta(j),j=k+1,irank)
        Write (nout,*)
        Flush (nout)

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
```

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

        Write (nout,*)
        Flush (nout)

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

        Write (nout,*)
        Flush (nout)

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

        Write (nout,*)
        Flush (nout)

        Call x04dbf('Upper triangular','Non-unit',irank,irank,a(1,n-irank+1),  &
          lda,'Bracketed','1P,E12.4','Nonsingular upper triangular matrix R',  &
          'Integer',rlabs,'Integer',clabs,80,0,ifail)

        Write (nout,*)
        Write (nout,*) 'Number of cycles of the Kogbetliantz method'
        Write (nout,99999) ncycle
      Else
        Write (nout,99997) 'Failure in ZTGSJA. INFO =', info
      End If

99999 Format (1X,I5)
99998 Format (3X,8(1P,E12.4))
99997 Format (1X,A,I4)
    End Program f08ysfe
```

10.2 Program Data

```
F08YSF 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
```

10.3 Program Results

```
 F08YSF Example Program Results

 Number of infinite generalized singular values (K)
     2
 Number of finite generalized singular values (L)
     2
  Effective Numerical rank of (A**T B**T)**T (K+L)
     4

 Finite generalized singular values
     2.0720E+00  1.1058E+00

 Unitary 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  (  2.5177E-01, -7.9789E-01) ( -5.0956E-02, -2.1750E-01)
 2  ( -3.2188E-01,  1.6112E-01) (  1.1979E-01,  1.6319E-01)
 3  (  1.3231E-01, -1.4565E-02) ( -5.0671E-01,  1.8615E-01)
 4  (  2.1598E-01,  1.8813E-01) ( -4.0163E-01,  2.6787E-01)
 5  (  3.6488E-02,  2.0316E-01) (  1.9271E-01,  1.5574E-01)
 6  (  1.0906E-01, -1.2712E-01) ( -8.8159E-02,  5.6169E-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)

 Unitary matrix V
                             1                           2
 1  (  9.8930E-01,  1.0471E-19) ( -1.1461E-01,  9.0250E-02)
 2  ( -1.1461E-01, -9.0250E-02) ( -9.8930E-01,  1.0471E-19)

 Unitary 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  (  6.9954E-01, -1.1784E-18) (  8.1044E-02, -6.3817E-02)
 2  ( -8.1044E-02, -6.3817E-02) (  6.9954E-01,  1.1784E-18)
 3  ( -6.9954E-01,  1.1784E-18) ( -8.1044E-02,  6.3817E-02)
 4  (  8.1044E-02,  6.3817E-02) ( -6.9954E-01, -1.1784E-18)

 Nonsingular upper triangular matrix R
                             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  ( -7.6930E-02,  1.3613E+00) ( -2.8137E-01, -3.2425E-02)
 2  ( -1.0760E+00,  3.1016E-02) (  1.3292E+00,  3.6772E-01)
 3  (  3.2537E+00,  0.0000E+00) ( -6.3858E-17,  3.4216E-33)
 4                              ( -2.1084E+00,  0.0000E+00)

 Number of cycles of the Kogbetliantz method
     2
```