NAG Library Routine Document

F08YTF (ZTGEXC)

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

F08YTF (ZTGEXC) reorders the generalized Schur factorization of a complex matrix pair in generalized Schur form.

2 Specification

```
SUBROUTINE F08YTF (WANTQ, WANTZ, N, A, LDA, B, LDB, Q, LDQ, Z, LDZ,        &
                   IFST, ILST, INFO)
INTEGER               N, LDA, LDB, LDQ, LDZ, IFST, ILST, INFO
COMPLEX (KIND=nag_wp) A(LDA,*), B(LDB,*), Q(LDQ,*), Z(LDZ,*)
LOGICAL               WANTQ, WANTZ
```

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

3 Description

F08YTF (ZTGEXC) reorders the generalized complex n by n matrix pair (S,T) in generalized Schur form, so that the diagonal element of (S,T) with row index i_1 is moved to row i_2, using a unitary equivalence transformation. That is, S and T are factorized as

$$S = \hat{Q}\hat{S}\hat{Z}^{\mathrm{H}}, \quad T = \hat{Q}\hat{T}\hat{Z}^{\mathrm{H}},$$

where $\left(\hat{S},\hat{T}\right)$ are also in generalized Schur form.

The pair (S,T) are in generalized Schur form if S and T are upper triangular as returned, for example, by F08XNF (ZGGES), or F08XSF (ZHGEQZ) with JOB = 'S'.

If S and T are the result of a generalized Schur factorization of a matrix pair (A,B)

$$A = QSZ^{\mathrm{H}}, \quad B = QTZ^{\mathrm{H}}$$

then, optionally, the matrices Q and Z can be updated as $Q\hat{Q}$ and $Z\hat{Z}$.

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

5 Arguments

1: WANTQ – LOGICAL *Input*

On entry: if WANTQ = .TRUE., update the left transformation matrix Q.

If WANTQ = .FALSE., do not update Q.

2: WANTZ – LOGICAL *Input*

On entry: if WANTZ = .TRUE., update the right transformation matrix Z.

If WANTZ = .FALSE., do not update Z.

3: N – INTEGER *Input*

On entry: n, the order of the matrices S and T.

Constraint: $N \geq 0$.

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

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

On entry: the matrix S in the pair (S, T).

On exit: the updated matrix $\hat{S}$.

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08YTF (ZTGEXC) is called.

Constraint: $LDA \geq \max(1, N)$.

6: $B(LDB,*)$ – 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 matrix T, in the pair (S, T).

On exit: the updated matrix $\hat{T}$

7: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08YTF (ZTGEXC) is called.

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

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

Note: the second dimension of the array Q must be at least $\max(1, N)$ if WANTQ = .TRUE., and at least 1 otherwise.

On entry: if WANTQ = .TRUE., the unitary matrix Q.

On exit: if WANTQ = .TRUE., the updated matrix $Q\hat{Q}$.

If WANTQ = .FALSE., Q is not referenced.

9: LDQ – INTEGER *Input*

On entry: the first dimension of the array Q as declared in the (sub)program from which F08YTF (ZTGEXC) is called.

Constraints:

if WANTQ = .TRUE., $LDQ \geq \max(1, N)$;
otherwise $LDQ \geq 1$.

10: $Z(LDZ,*)$ – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array Z must be at least $\max(1, N)$ if WANTZ = .TRUE., and at least 1 otherwise.

On entry: if WANTZ = .TRUE., the unitary matrix Z.

On exit: if WANTZ = .TRUE., the updated matrix $Z\hat{Z}$.

If WANTZ = .FALSE., Z is not referenced.

11: LDZ – INTEGER *Input*

On entry: the first dimension of the array Z as declared in the (sub)program from which F08YTF (ZTGEXC) is called.

Constraints:

if WANTZ = .TRUE., LDZ $\geq \max(1, N)$;
otherwise LDZ ≥ 1.

12: IFST – INTEGER *Input*
13: ILST – INTEGER *Input/Output*

On entry: the indices i_1 and i_2 that specify the reordering of the diagonal elements of (S, T). The element with row index IFST is moved to row ILST, by a sequence of swapping between adjacent diagonal elements.

On exit: ILST points to the row in its final position.

Constraint: $1 \leq$ IFST $\leq$ N and $1 \leq$ ILST $\leq$ N.

14: 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 transformed matrix pair $\left(\hat{S}, \hat{T}\right)$ would be too far from generalized Schur form; the problem is ill-conditioned. (S, T) may have been partially reordered, and ILST points to the first row of the current position of the block being moved.

7 Accuracy

The computed generalized Schur form is nearly the exact generalized Schur form for nearby matrices $(S + E)$ and $(T + F)$, where

$$\|E\|_2 = O\,\epsilon\|S\|_2 \quad \text{and} \quad \|F\|_2 = O\,\epsilon\|T\|_2,$$

and ϵ is the ***machine precision***. See Section 4.11 of Anderson *et al.* (1999) for further details of error bounds for the generalized nonsymmetric eigenproblem.

8 Parallelism and Performance

F08YTF (ZTGEXC) is not threaded in any implementation.

9 Further Comments

The real analogue of this routine is F08YFF (DTGEXC).

10 Example

This example exchanges rows 4 and 1 of the matrix pair (S, T), where

$$S = \begin{pmatrix} 4.0+4.0i & 1.0+1.0i & 1.0+1.0i & 2.0-1.0i \\ 0 & 2.0+1.0i & 1.0+1.0i & 1.0+1.0i \\ 0 & 0 & 2.0-1.0i & 1.0+1.0i \\ 0 & 0 & 0 & 6.0-2.0i \end{pmatrix}$$

and

$$T = \begin{pmatrix} 2.0 & 1.0+1.0i & 1.0+1.0i & 3.0-1.0i \\ 0 & 1.0 & 2.0+1.0i & 1.0+1.0i \\ 0 & 0 & 1.0 & 1.0+1.0i \\ 0 & 0 & 0 & 2.0 \end{pmatrix}.$$

10.1 Program Text

```
    Program f08ytfe

!     F08YTF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dbf, ztgexc
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
      Logical, Parameter              :: wantq = .False., wantz = .False.
!     .. Local Scalars ..
      Integer                         :: i, ifail, ifst, ilst, info, lda,      &
                                         ldb, ldq, ldz, n
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: a(:,:), b(:,:), q(:,:), z(:,:)
      Character (1)                   :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F08YTF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      ldq = 1
      ldz = 1
      lda = n
      ldb = n
      Allocate (a(lda,n),b(ldb,n),q(ldq,1),z(ldz,1))

!     Read A and B from data file

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

!     Read the row indices

      Read (nin,*) ifst, ilst

!     Reorder the A and B

!     The NAG name equivalent of ztgexc is f08ytf
      Call ztgexc(wantq,wantz,n,a,lda,b,ldb,q,ldq,z,ldz,ifst,ilst,info)

      If (info/=0) Then
        Write (nout,99999) info, ilst
        Write (nout,*)
        Flush (nout)
      End If
```

```
!     Print reordered generalized Schur form

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call x04dbf('General',' ',n,n,a,lda,'Bracketed','F7.4',               &
        'Reordered Schur matrix A','Integer',rlabs,'Integer',clabs,80,0,ifail)

      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04dbf('General',' ',n,n,b,ldb,'Bracketed','F7.4',               &
        'Reordered Schur matrix B','Integer',rlabs,'Integer',clabs,80,0,ifail)

99999 Format (' Reordering could not be completed. INFO = ',I3,' ILST = ',I5)
    End Program f08ytfe
```

10.2 Program Data

```
F08YTF Example Program Data
  4                                                 :Value of N
 ( 4.0, 4.0) ( 1.0, 1.0) ( 1.0, 1.0) ( 2.0,-1.0)
 ( 0.0, 0.0) ( 2.0, 1.0) ( 1.0, 1.0) ( 1.0, 1.0)
 ( 0.0, 0.0) ( 0.0, 0.0) ( 2.0,-1.0) ( 1.0, 1.0)
 ( 0.0, 0.0) ( 0.0, 0.0) ( 0.0, 0.0) ( 6.0,-2.0)   :End of matrix A
 ( 2.0, 0.0) ( 1.0, 1.0) ( 1.0, 1.0) ( 3.0,-1.0)
 ( 0.0, 0.0) ( 1.0, 0.0) ( 2.0, 1.0) ( 1.0, 1.0)
 ( 0.0, 0.0) ( 0.0, 0.0) ( 1.0, 0.0) ( 1.0, 1.0)
 ( 0.0, 0.0) ( 0.0, 0.0) ( 0.0, 0.0) ( 2.0, 0.0)   :End of matrix B
  1  4                                              :Values of IFST and ILST
```

10.3 Program Results

```
 F08YTF Example Program Results

 Reordered Schur matrix A
                   1                  2                  3                  4
 1  ( 3.7081, 3.7081) (-2.0834,-0.5688) ( 2.6374, 1.0772) ( 0.2845, 0.7991)
 2  ( 0.0000, 0.0000) ( 1.6097, 1.5656) (-0.0634, 1.9234) (-0.0301, 0.9720)
 3  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 4.7029,-2.1187) ( 1.1379,-3.1199)
 4  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 2.3085,-1.8289)

 Reordered Schur matrix B
                   1                  2                  3                  4
 1  ( 2.2249, 0.7416) (-1.1631, 1.5347) ( 2.2608, 2.0851) ( 1.1094,-0.3205)
 2  ( 0.0000, 0.0000) ( 0.3308, 0.9482) ( 0.3919, 1.8172) (-0.6305, 1.6053)
 3  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 1.6227,-0.1653) ( 0.9966,-0.9074)
 4  ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.0000, 0.0000) ( 0.1199,-1.0343)
```