NAG Fortran Library Routine Document

F08YVF (ZTGSYL)

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

F08YVF (ZTGSYL) solves the generalized complex triangular Sylvester equations.

2 Specification

```
 SUBROUTINE F08YVF (TRANS, IJOB, M, N, A, LDA, B, LDB, C, LDC, D, LDD, E,
1                   LDE, F, LDF, DIF, SCALE, WORK, LWORK, IWORK, INFO)

 INTEGER            IJOB, M, N, LDA, LDB, LDC, LDD, LDE, LDF, LWORK,
1                   IWORK(*), INFO
 double precision   DIF, SCALE
 complex*16         A(LDA,*), B(LDB,*), C(LDC,*), D(LDD,*), E(LDE,*),
1                   F(LDF,*), WORK(*)
 CHARACTER*1        TRANS
```

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

3 Description

F08YVF (ZTGSYL) solves either the generalized complex Sylvester equations

$$\begin{aligned} AR - LB &= \alpha C \\ DR - LE &= \alpha F\,', \end{aligned} \tag{1}$$

or the equations

$$\begin{aligned} A^{\mathrm{H}}R + D^{\mathrm{H}}L &= \alpha C \\ RB^{\mathrm{H}} + LE^{\mathrm{H}} &= -\alpha F\,', \end{aligned} \tag{2}$$

where the pair (A, D) are given m by m matrices in generalized Schur form, (B, E) are given n by n matrices in generalized Schur form and (C, F) are given m by n matrices. The pair (R, L) are the m by n solution matrices, and α is an output scaling factor determined by the routine to avoid overflow in computing (R, L).

Equations (1) are equivalent to equations of the form

$$Zx = \alpha b,$$

where

$$Z = \begin{pmatrix} I \otimes A - B^{\mathrm{H}} \otimes I \\ I \otimes D - E^{\mathrm{H}} \otimes I \end{pmatrix}$$

and $\otimes$ is the Kronecker product. Equations (2) are then equivalent to

$$Z^{\mathrm{H}}y = \alpha b.$$

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'.

Optionally, the routine estimates $\mathrm{Dif}[(A, D), (B, E)]$, the separation between the matrix pairs (A, D) and (B, E), which is the smallest singular value of Z. The estimate can be based on either the Frobenius norm, or the one norm. The one norm estimate can be three to ten times more expensive than the Frobenius norm estimate, but makes the condition estimation uniform with the nonsymmetric eigenproblem. The Frobenius norm estimate provides a low cost, but equally reliable estimate. For more information see Anderson *et al.* (1999) (Sections 2.4.8.3 and 4.11.1.3) and Kågström and Poromaa (1996).

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

Kågström B (1994) A perturbation analysis of the generalized Sylvester equation $(AR - LB, DR - LE) = (c, F)$ *SIAM J. Matrix Anal. Appl.* **15** 1045–1060

Kågström B and Poromaa P (1996) LAPACK-style algorithms and software for solving the generalized Sylvester equation and estimating the separation between regular matrix pairs *ACM Trans. Math. Software* **22** 78–103

5 Parameters

1: TRANS – CHARACTER*1 *Input*

On entry: if TRANS = 'N', solve the generalized Sylvester equation (1).

If TRANS = 'C', solve the 'conjugate transposed' system (2).

Constraint: TRANS = 'N' or 'C'.

2: IJOB – INTEGER *Input*

On entry: specifies what kind of functionality is to be performed when TRANS = 'N'.

IJOB = 0

Solve (1) only.

IJOB = 1

The functionality of IJOB = 0 and 3.

IJOB = 2

The functionality of IJOB = 0 and 4.

IJOB = 3

Only an estimate of $\mathrm{Dif}[(A, D), (B, E)]$ is computed based on the Frobenius norm.

IJOB = 4

Only an estimate of $\mathrm{Dif}[(A, D), (B, E)]$ is computed based on the one norm.

If TRANS = 'C', IJOB is not referenced.

3: M – INTEGER *Input*

On entry: m, the order of the matrices A and D, and the row dimension of the matrices C, F, R and L.

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

4: N – INTEGER *Input*

On entry: n, the order of the matrices B and E, and the column dimension of the matrices C, F, R and L.

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

5: A(LDA,$*$) – ***complex*16*** array *Input*

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

On entry: the upper triangular matrix A.

6: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

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

7: B(LDB,$*$) – ***complex*16*** array *Input*

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

On entry: the upper triangular matrix B.

8: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

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

9: C(LDC,$*$) – ***complex*16*** array *Input/Output*

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

On entry: contains the right-hand-side matrix C.

On exit: if TRANS = 'C' or 'N' and IJOB = 0, 1 or 2, C is overwritten by the solution matrix R.

If TRANS = 'N' and IJOB = 3 or 4, C holds R, the solution achieved during the computation of the Dif estimate.

10: LDC – INTEGER *Input*

On entry: the first dimension of the array C as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

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

11: D(LDD,$*$) – ***complex*16*** array *Input*

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

On entry: the upper triangular matrix D.

12: LDD – INTEGER *Input*

On entry: the first dimension of the array D as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

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

13: E(LDE,$*$) – ***complex*16*** array *Input*

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

On entry: the upper triangular matrix E.

14: LDE – INTEGER *Input*

On entry: the first dimension of the array E as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

Constraint: LDE $\geq \max(1, \mathrm{N})$.

15: F(LDF,$*$) – ***complex*16*** array *Input/Output*

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

On entry: contains the right-hand side matrix F.

On exit: if TRANS = 'C' or 'N' and IJOB = 0, 1 or 2, F is overwritten by the solution matrix L.

If TRANS = 'N' and IJOB = 3 or 4, F holds L, the solution achieved during the computation of the Dif estimate.

16: LDF – INTEGER *Input*

On entry: the first dimension of the array F as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

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

17: DIF – ***double precision*** *Output*

On exit: the estimate of Dif. If TRANS = 'C' or 'N' and IJOB = 0, DIF is not referenced.

18: SCALE – ***double precision*** *Output*

On exit: α, the scaling factor in (1) or (2).

If $0 <$ SCALE < 1, C and F hold the solutions R and L, respectively, to a slightly perturbed system but the input arrays A, B, D and E have not been changed.

If SCALE = 0, C and F hold the solutions R and L, respectively, to the homogeneous system with $C = F = 0$. In this case DIF is not referenced.

Normally, SCALE = 1.

19: WORK($*$) – ***complex*16*** array *Workspace*

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

On exit: if INFO = 0, the real part of WORK(1) contains the minimum value of LWORK required for optimal performance.

20: LWORK – INTEGER *Input*

On entry: the dimension of the array WORK as declared in the (sub)program from which F08YVF (ZTGSYL) is called.

If LWORK $= -1$, a workspace query is assumed; the routine only calculates the minimum size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued.

Constraints:

if LWORK $\neq -1$,
    if TRANS = 'N' and IJOB = 1 or 2, LWORK $\geq 2 \times \mathrm{M} \times \mathrm{N}$;
    LWORK ≥ 1 otherwise.

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

Note: the dimension of the array IWORK must be at least $\mathrm{M} + \mathrm{N} + 2$.

22: 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 $= -i$, the ith parameter had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO > 0

(A, D) and (B, E) have common or very close eigenvalues and so no solution could be computed.

7 Accuracy

See Kågström (1994) for a perturbation analysis of the generalized Sylvester equation.

8 Further Comments

The total number of floating point operations needed to solve the generalized Sylvester equations is approximately $8mn(n+m)$. The Frobenius norm estimate of dif does not require additional significant computation, but the one norm estimate is typically five times more expensive.

The real analogue of this routine is F08YHF (DTGSYL).

9 Example

This example solves the generalized Sylvester equations

$$\begin{aligned} AR - LB &= \alpha C \\ DR - LE &= \alpha F \text{'} \end{aligned}$$

where

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

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

$$D = \begin{pmatrix} 1.0+1.0i & 1.0-1.0i & 1.0+1.0i & 1.0-1.0i \\ 0 & 6.0-4.0i & 1.0-1.0i & 1.0+1.0i \\ 0 & 0 & 2.0+4.0i & 1.0-1.0i \\ 0 & 0 & 0 & 2.0+3.0i \end{pmatrix},$$

$$E = \begin{pmatrix} 1.0 & 1.0+1.0i & 1.0-1.0i & 1.0+1.0i \\ 0 & 2.0 & 1.0+1.0i & 1.0-1.0i \\ 0 & 0 & 2.0 & 1.0+1.0i \\ 0 & 0 & 0 & 1.0 \end{pmatrix},$$

$$C = \begin{pmatrix} -13.0+9.0i & 2.0+8.0i & -2.0+8.0i & -2.0+5.0i \\ -9.0-1.0i & 0.0+5.0i & -7.0-3.0i & -6.0-0.0i \\ -1.0+1.0i & 4.0+2.0i & 4.0-5.0i & 9.0-5.0i \\ -6.0+6.0i & 9.0+1.0i & -2.0+4.0i & 22.0-8.0i \end{pmatrix}$$

and

$$F = \begin{pmatrix} -6.0+5.0i & 4.0-4.0i & -3.0+11.0i & 3.0-7.0i \\ -5.0+11.0i & 12.0-4.0i & -2.0+2.0i & 0.0+14.0i \\ -5.0-1.0i & 0.0+4.0i & -2.0+10.0i & 3.0-1.0i \\ -6.0-2.0i & 1.0+1.0i & -7.0-3.0i & 4.0+7.0i \end{pmatrix}.$$

9.1 Program Text

```
*     F08YVF Example Program Text
*     Mark 21 Release. NAG Copyright 2004.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MMAX, NMAX
      PARAMETER        (MMAX=8,NMAX=8)
      INTEGER          LDA, LDB, LDC, LDD, LDE
      PARAMETER        (LDA=MMAX,LDB=NMAX,LDC=MMAX,LDD=MMAX,LDE=NMAX)
      INTEGER          LDF, LWORK
      PARAMETER        (LDF=MMAX,LWORK=1)
*     .. Local Scalars ..
      DOUBLE PRECISION DIF, SCALE
      INTEGER          I, IFAIL, IJOB, INFO, J, M, N
*     .. Local Arrays ..
      COMPLEX *16      A(LDA,MMAX), B(LDB,NMAX), C(LDC,NMAX),
     +                 D(LDD,MMAX), E(LDE,NMAX), F(LDF,NMAX),
     +                 WORK(LWORK)
      INTEGER          IWORK(MMAX+NMAX+2)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         X04DBF, ZTGSYL
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F08YVF Example Program Results'
      WRITE (NOUT,*)
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) M, N
      IF (M.LE.MMAX .AND. N.LE.NMAX) THEN
*
*        Read A, B, D, E, C and F from data file
*
         READ (NIN,*) ((A(I,J),J=1,M),I=1,M)
         READ (NIN,*) ((B(I,J),J=1,N),I=1,N)
         READ (NIN,*) ((D(I,J),J=1,M),I=1,M)
         READ (NIN,*) ((E(I,J),J=1,N),I=1,N)
         READ (NIN,*) ((C(I,J),J=1,N),I=1,M)
         READ (NIN,*) ((F(I,J),J=1,N),I=1,M)
*
*        Solve the Sylvester equations
*           A*R - L*B = scale*C and D*R - L*E = scale*F
*        for R and L.
*
         IJOB = 0
         CALL ZTGSYL('No transpose',IJOB,M,N,A,LDA,B,LDB,C,LDC,D,LDD,E,
     +               LDE,F,LDF,SCALE,DIF,WORK,LWORK,IWORK,INFO)
         IF (INFO.GE.1) THEN
            WRITE (NOUT,99999)
            WRITE (NOUT,*)
         END IF
*
*        Print the solution matrices R and L
*
         IFAIL = 0
         CALL X04DBF('General',' ',M,N,C,LDC,'Bracketed','F7.4',
     +               'Solution matrix R','Integer',RLABS,'Integer',
     +               CLABS,80,0,IFAIL)
*
         WRITE (NOUT,*)
         IFAIL = 0
         CALL X04DBF('General',' ',M,N,F,LDF,'Bracketed','F7.4',
     +               'Solution matrix L','Integer',RLABS,'Integer',
     +               CLABS,80,0,IFAIL)
*
         WRITE (NOUT,*)
         WRITE (NOUT,99998) 'SCALE = ', SCALE
      ELSE
         WRITE (NOUT,*) 'MMAX and/or NMAX too small'
      END IF
```

```
      STOP
*
99999 FORMAT (/' (A,D) and (B,E) have common or very close ','eigenval',
     +       'ues.',/' Perturbed values were used to solve ','the equa',
     +       'tions')
99998 FORMAT (1X,A,1P,E10.2)
      END
```

9.2 Program Data

```
F08YVF Example Program Data
  4  4                                                    :Values of M and 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
 (  1.0,  1.0) (  1.0, -1.0) (  1.0,  1.0) (  1.0, -1.0)
 (  0.0,  0.0) (  6.0, -4.0) (  1.0, -1.0) (  1.0,  1.0)
 (  0.0,  0.0) (  0.0,  0.0) (  2.0,  4.0) (  1.0, -1.0)
 (  0.0,  0.0) (  0.0,  0.0) (  0.0,  0.0) (  2.0,  3.0)   :End of matrix D
 (  1.0,  0.0) (  1.0,  1.0) (  1.0, -1.0) (  1.0,  1.0)
 (  0.0,  0.0) (  2.0,  0.0) (  1.0,  1.0) (  1.0, -1.0)
 (  0.0,  0.0) (  0.0,  0.0) (  2.0,  0.0) (  1.0,  1.0)
 (  0.0,  0.0) (  0.0,  0.0) (  0.0,  0.0) (  1.0,  0.0)   :End of matrix E
 (-13.0,  9.0) (  2.0,  8.0) ( -2.0,  8.0) ( -2.0,  5.0)
 ( -9.0, -1.0) (  0.0,  5.0) ( -7.0, -3.0) ( -6.0,  0.0)
 ( -1.0,  1.0) (  4.0,  2.0) (  4.0, -5.0) (  9.0, -5.0)
 ( -6.0,  6.0) (  9.0,  1.0) ( -2.0,  4.0) ( 22.0, -8.0)  :End of matrix C
 ( -6.0,  5.0) (  4.0, -4.0) ( -3.0, 11.0) (  3.0, -7.0)
 ( -5.0, 11.0) ( 12.0, -4.0) ( -2.0,  2.0) (  0.0, 14.0)
 ( -5.0, -1.0) (  0.0,  4.0) ( -2.0, 10.0) (  3.0, -1.0)
 ( -6.0, -2.0) (  1.0,  1.0) ( -7.0, -3.0) (  4.0,  7.0)  :End of matrix F
```

9.3 Program Results

```
 F08YVF Example Program Results

 Solution matrix R
                     1                  2                  3                  4
 1  ( 1.0000, 1.0000) ( 1.0000, 1.0000) ( 1.0000, 1.0000) ( 1.0000, 1.0000)
 2  (-1.0000, 1.0000) ( 2.0000, 1.0000) (-1.0000, 1.0000) (-1.0000, 1.0000)
 3  (-1.0000, 1.0000) ( 1.0000, 1.0000) ( 3.0000, 1.0000) ( 1.0000, 1.0000)
 4  (-1.0000, 1.0000) ( 1.0000, 1.0000) (-1.0000, 1.0000) ( 4.0000, 1.0000)

 Solution matrix L
                     1                  2                  3                  4
 1  ( 4.0000, 1.0000) (-1.0000, 1.0000) ( 1.0000, 1.0000) (-1.0000, 1.0000)
 2  ( 1.0000, 1.0000) ( 3.0000, 1.0000) (-1.0000, 1.0000) ( 1.0000, 1.0000)
 3  (-1.0000, 1.0000) ( 1.0000, 1.0000) ( 2.0000, 1.0000) (-1.0000, 1.0000)
 4  ( 1.0000, 1.0000) (-1.0000, 1.0000) ( 1.0000, 1.0000) ( 1.0000, 1.0000)

 SCALE =   1.00E+00
```