NAG Library Routine Document

F08YHF (DTGSYL)

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

F08YHF (DTGSYL) solves the generalized real quasi-triangular Sylvester equations.

2 Specification

```
SUBROUTINE F08YHF (TRANS, IJOB, M, N, A, LDA, B, LDB, C, LDC, D, LDD, E,        &
                   LDE, F, LDF, SCALE, DIF, WORK, LWORK, IWORK, INFO)

INTEGER            IJOB, M, N, LDA, LDB, LDC, LDD, LDE, LDF, LWORK,             &
                   IWORK(max(1,M+N+6)), INFO
REAL (KIND=nag_wp) A(LDA,*), B(LDB,*), C(LDC,*), D(LDD,*), E(LDE,*),            &
                   F(LDF,*), SCALE, DIF, WORK(max(1,LWORK))
CHARACTER(1)       TRANS
```

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

3 Description

F08YHF (DTGSYL) solves either the generalized real Sylvester equations

$$\begin{aligned} AR - LB &= \alpha C \\ DR - LE &= \alpha F, \end{aligned} \qquad (1)$$

or the equations

$$\begin{aligned} A^{\mathrm{T}}R + D^{\mathrm{T}}L &= \alpha C \\ RB^{\mathrm{T}} + LE^{\mathrm{T}} &= -\alpha F, \end{aligned} \qquad (2)$$

where the pair (A, D) are given m by m matrices in real generalized Schur form, (B, E) are given n by n matrices in real 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

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

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

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

The pair (S, T) are in real generalized Schur form if S is block upper triangular with 1 by 1 and 2 by 2 diagonal blocks on the diagonal and T is upper triangular as returned, for example, by F08XAF (DGGES), or F08XEF (DHGEQZ) 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 1-norm. The 1-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 Sections 2.4.8.3 and 4.11.1.3 of Anderson *et al.* (1999) 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 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 = 'T', solve the 'transposed' system (2).

Constraint: TRANS = 'N' or 'T'.

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 1-norm.

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

Constraint: if TRANS = 'N', $0 \leq \mathrm{IJOB} \leq 4$.

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,$*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: the upper quasi-triangular matrix A.

6: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08YHF (DTGSYL) is called.

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

7: B(LDB,$*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: the upper quasi-triangular matrix B.

8: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08YHF (DTGSYL) is called.

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

9: C(LDC,$*$) – REAL (KIND=nag_wp) array *Input/Output*

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

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

On exit: if $\text{IJOB} = 0$, 1 or 2, C is overwritten by the solution matrix R.

If TRANS = 'N' and $\text{IJOB} = 3$ 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 F08YHF (DTGSYL) is called.

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

11: D(LDD,$*$) – REAL (KIND=nag_wp) array *Input*

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

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 F08YHF (DTGSYL) is called.

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

13: E(LDE,$*$) – REAL (KIND=nag_wp) array *Input*

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

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 F08YHF (DTGSYL) is called.

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

15: F(LDF,$*$) – REAL (KIND=nag_wp) array *Input/Output*

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

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

On exit: if 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 F08YHF (DTGSYL) is called.

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

17: SCALE – REAL (KIND=nag_wp) *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 $= 1$.

18: DIF – REAL (KIND=nag_wp) *Output*

On exit: the estimate of Dif. If IJOB $= 0$, DIF is not referenced.

19: WORK$(\max(1, \mathrm{LWORK}))$ – REAL (KIND=nag_wp) array *Workspace*

On exit: if INFO $= 0$, 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 F08YHF (DTGSYL) 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 $\geq \max(1, 2 \times \mathrm{M} \times \mathrm{N})$;
otherwise LWORK ≥ 1.

21: IWORK$(\max(1, \mathrm{M} + \mathrm{N} + 6))$ – INTEGER array *Workspace*

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 $= -i$, argument i 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 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 $2mn(n+m)$. The Frobenius norm estimate of Dif does not require additional significant computation, but the 1-norm estimate is typically five times more expensive.

The complex analogue of this routine is F08YVF (ZTGSYL).

9 Example

This example solves the generalized Sylvester equations

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

where

$$A = \begin{pmatrix} 4.0 & 1.0 & 1.0 & 2.0 \\ 0 & 3.0 & 4.0 & 1.0 \\ 0 & 1.0 & 3.0 & 1.0 \\ 0 & 0 & 0 & 6.0 \end{pmatrix}, \quad B = \begin{pmatrix} 1.0 & 1.0 & 1.0 & 1.0 \\ 0 & 3.0 & 4.0 & 1.0 \\ 0 & 1.0 & 3.0 & 1.0 \\ 0 & 0 & 0 & 4.0 \end{pmatrix},$$

$$D = \begin{pmatrix} 2.0 & 1.0 & 1.0 & 3.0 \\ 0 & 1.0 & 2.0 & 1.0 \\ 0 & 0 & 1.0 & 1.0 \\ 0 & 0 & 0 & 2.0 \end{pmatrix}, \quad E = \begin{pmatrix} 1.0 & 1.0 & 1.0 & 2.0 \\ 0 & 1.0 & 4.0 & 1.0 \\ 0 & 0 & 1.0 & 1.0 \\ 0 & 0 & 0 & 1.0 \end{pmatrix},$$

$$C = \begin{pmatrix} -4.0 & 7.0 & 1.0 & 12.0 \\ -9.0 & 2.0 & -2.0 & -2.0 \\ -4.0 & 2.0 & -2.0 & 8.0 \\ -7.0 & 7.0 & -6.0 & 19.0 \end{pmatrix} \quad \text{and} \quad F = \begin{pmatrix} -7.0 & 5.0 & 0.0 & 7.0 \\ -5.0 & 1.0 & -8.0 & 0.0 \\ -1.0 & 2.0 & -3.0 & 5.0 \\ -3.0 & 2.0 & 0.0 & 5.0 \end{pmatrix}.$$

9.1 Program Text

```
    Program f08yhfe

!     F08YHF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dtgsyl, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter                :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)                :: dif, scale
      Integer                           :: i, ifail, ijob, info, lda, ldb, ldc, &
                                           ldd, lde, ldf, lwork, m, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable   :: a(:,:), b(:,:), c(:,:), d(:,:),      &
                                           e(:,:), f(:,:), work(:)
      Integer, Allocatable              :: iwork(:)
!     .. Executable Statements ..
      Write (nout,*) 'F08YHF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
```

```
      Read (nin,*)
      Read (nin,*) m, n
      lda = m
      ldb = n
      ldc = m
      ldd = m
      lde = n
      ldf = m
      lwork = 1
      Allocate (a(lda,m),b(ldb,n),c(ldc,n),d(ldd,m),e(lde,n),f(ldf,n), &
        work(lwork),iwork(m+n+6))

!     Read A, B, D, E, C and F from data file

      Read (nin,*)(a(i,1:m),i=1,m)
      Read (nin,*)(b(i,1:n),i=1,n)
      Read (nin,*)(d(i,1:m),i=1,m)
      Read (nin,*)(e(i,1:n),i=1,n)
      Read (nin,*)(c(i,1:n),i=1,m)
      Read (nin,*)(f(i,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

!     The NAG name equivalent of dtgsyl is f08yhf
      Call dtgsyl('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>=1) Then
        Write (nout,99999)
        Write (nout,*)
        Flush (nout)
      End If

!     Print the solution matrices R and L

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call x04caf('General',' ',m,n,c,ldc,'Solution matrix R',ifail)

      Write (nout,*)
      Flush (nout)

      ifail = 0
      Call x04caf('General',' ',m,n,f,ldf,'Solution matrix L',ifail)

      Write (nout,*)
      Write (nout,99998) 'SCALE = ', scale

99999 Format (/' (A,D) and (B,E) have common or very close eigenval', &
        'ues.'/' Perturbed values were used to solve the equations')
99998 Format (1X,A,1P,E10.2)
    End Program f08yhfe
```

9.2 Program Data

```
F08YHF Example Program Data
  4  4                         :Values of M and N
  4.0   1.0   1.0   2.0
  0.0   3.0   4.0   1.0
  0.0   1.0   3.0   1.0
  0.0   0.0   0.0   6.0   :End of matrix A
  1.0   1.0   1.0   1.0
  0.0   3.0   4.0   1.0
  0.0   1.0   3.0   1.0
  0.0   0.0   0.0   4.0   :End of matrix B
  2.0   1.0   1.0   3.0
```

```
  0.0   1.0   2.0   1.0
  0.0   0.0   1.0   1.0
  0.0   0.0   0.0   2.0   :End of matrix D
  1.0   1.0   1.0   2.0
  0.0   1.0   4.0   1.0
  0.0   0.0   1.0   1.0
  0.0   0.0   0.0   1.0   :End of matrix E
 -4.0   7.0   1.0  12.0
 -9.0   2.0  -2.0  -2.0
 -4.0   2.0  -2.0   8.0
 -7.0   7.0  -6.0  19.0   :End of matrix C
 -7.0   5.0   0.0   7.0
 -5.0   1.0  -8.0   0.0
 -1.0   2.0  -3.0   5.0
 -3.0   2.0   0.0   5.0   :End of matrix F
```

9.3 Program Results

```
 F08YHF Example Program Results

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

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

 SCALE =   1.00E+00
```