NAG Library Routine Document

F07CHF (DGTRFS)

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

F07CHF (DGTRFS) computes error bounds and refines the solution to a real system of linear equations $AX = B$ or $A^{\mathrm{T}}X = B$, where A is an n by n tridiagonal matrix and X and B are n by r matrices, using the LU factorization returned by F07CDF (DGTTRF) and an initial solution returned by F07CEF (DGTTRS). Iterative refinement is used to reduce the backward error as much as possible.

2 Specification

```
SUBROUTINE F07CHF (TRANS, N, NRHS, DL, D, DU, DLF, DF, DUF, DU2, IPIV,      &
                   B, LDB, X, LDX, FERR, BERR, WORK, IWORK, INFO)
INTEGER            N, NRHS, IPIV(*), LDB, LDX, IWORK(N), INFO
REAL (KIND=nag_wp) DL(*), D(*), DU(*), DLF(*), DF(*), DUF(*), DU2(*),       &
                   B(LDB,*), X(LDX,*), FERR(NRHS), BERR(NRHS),              &
                   WORK(3*N)
CHARACTER(1)       TRANS
```

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

3 Description

F07CHF (DGTRFS) should normally be preceded by calls to F07CDF (DGTTRF) and F07CEF (DGTTRS). F07CDF (DGTTRF) uses Gaussian elimination with partial pivoting and row interchanges to factorize the matrix A as

$$A = PLU,$$

where P is a permutation matrix, L is unit lower triangular with at most one nonzero subdiagonal element in each column, and U is an upper triangular band matrix, with two superdiagonals. F07CEF (DGTTRS) then utilizes the factorization to compute a solution, $\hat{X}$, to the required equations. Letting $\hat{x}$ denote a column of $\hat{X}$, F07CHF (DGTRFS) computes a *component-wise backward error*, β, the smallest relative perturbation in each element of A and b such that $\hat{x}$ is the exact solution of a perturbed system

$$(A+E)\hat{x} = b + f, \quad \text{with} \quad |e_{ij}| \leq \beta|a_{ij}|, \quad \text{and} \quad |f_j| \leq \beta|b_j|.$$

The routine also estimates a bound for the *component-wise forward error* in the computed solution defined by $\max|x_i - \hat{x}_i| / \max|\hat{x}_i|$, where x is the corresponding column of the exact solution, X.

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 Parameters

1: TRANS – CHARACTER(1) *Input*

On entry: specifies the equations to be solved as follows:

TRANS = 'N'
Solve $AX = B$ for X.

TRANS = 'T' or 'C'
    Solve $A^{\mathrm{T}}X = B$ for X.

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

2: N – INTEGER *Input*

On entry: n, the order of the matrix A.

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

3: NRHS – INTEGER *Input*

On entry: r, the number of right-hand sides, i.e., the number of columns of the matrix B.

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

4: DL($*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: must contain the $(n-1)$ subdiagonal elements of the matrix A.

5: D($*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: must contain the n diagonal elements of the matrix A.

6: DU($*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: must contain the $(n-1)$ superdiagonal elements of the matrix A.

7: DLF($*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: must contain the $(n-1)$ multipliers that define the matrix L of the LU factorization of A.

8: DF($*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: must contain the n diagonal elements of the upper triangular matrix U from the LU factorization of A.

9: DUF($*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: must contain the $(n-1)$ elements of the first superdiagonal of U.

10: DU2($*$) – REAL (KIND=nag_wp) array *Input*

Note: the dimension of the array DU2 must be at least $\max(1, \mathrm{N} - 2)$.

On entry: must contain the $(n-2)$ elements of the second superdiagonal of U.

11: IPIV($*$) – INTEGER array *Input*

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

On entry: must contain the n pivot indices that define the permutation matrix P. At the ith step, row i of the matrix was interchanged with row $\mathrm{IPIV}(i)$, and $\mathrm{IPIV}(i)$ must always be either i or $(i+1)$, $\mathrm{IPIV}(i) = i$ indicating that a row interchange was not performed.

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

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

On entry: the n by r matrix of right-hand sides B.

13: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07CHF (DGTRFS) is called.

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

14: X(LDX, $*$) – REAL (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array X must be at least max(1, NRHS).

On entry: the n by r initial solution matrix X.

On exit: the n by r refined solution matrix X.

15: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07CHF (DGTRFS) is called.

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

16: FERR(NRHS) – REAL (KIND=nag_wp) array *Output*

On exit: estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|\hat{x}_j\|_\infty \leq \mathrm{FERR}(j)$, where $\hat{x}_j$ is the jth column of the computed solution returned in the array X and x_j is the corresponding column of the exact solution X. The estimate is almost always a slight overestimate of the true error.

17: BERR(NRHS) – REAL (KIND=nag_wp) array *Output*

On exit: estimate of the component-wise relative backward error of each computed solution vector $\hat{x}_j$ (i.e., the smallest relative change in any element of A or B that makes $\hat{x}_j$ an exact solution).

18: WORK($3 \times$ N) – REAL (KIND=nag_wp) array *Workspace*

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

20: 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.

7 Accuracy

The computed solution for a single right-hand side, $\hat{x}$, satisfies an equation of the form

$$(A + E)\hat{x} = b,$$

where

$$\|E\|_\infty = O(\epsilon)\|A\|_\infty$$

and ϵ is the ***machine precision***. An approximate error bound for the computed solution is given by

$$\frac{\|\hat{x}-x\|_\infty}{\|x\|_\infty} \leq \kappa(A)\frac{\|E\|_\infty}{\|A\|_\infty},$$

where $\kappa(A) = \|A^{-1}\|_\infty \|A\|_\infty$, the condition number of A with respect to the solution of the linear equations. See Section 4.4 of Anderson *et al.* (1999) for further details.

Routine F07CGF (DGTCON) can be used to estimate the condition number of A.

8 Parallelism and Performance

F07CHF (DGTRFS) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F07CHF (DGTRFS) 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 total number of floating-point operations required to solve the equations $AX = B$ or $A^{\mathrm{T}}X = B$ is proportional to nr. At most five steps of iterative refinement are performed, but usually only one or two steps are required.

The complex analogue of this routine is F07CVF (ZGTRFS).

10 Example

This example solves the equations

$$AX = B,$$

where A is the tridiagonal matrix

$$A = \begin{pmatrix} 3.0 & 2.1 & 0 & 0 & 0 \\ 3.4 & 2.3 & -1.0 & 0 & 0 \\ 0 & 3.6 & -5.0 & 1.9 & 0 \\ 0 & 0 & 7.0 & -0.9 & 8.0 \\ 0 & 0 & 0 & -6.0 & 7.1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 2.7 & 6.6 \\ -0.5 & 10.8 \\ 2.6 & -3.2 \\ 0.6 & -11.2 \\ 2.7 & 19.1 \end{pmatrix}.$$

Estimates for the backward errors and forward errors are also output.

10.1 Program Text

```
    Program f07chfe

!     F07CHF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: dgtrfs, dgttrf, dgttrs, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, ldb, ldx, n, nrhs
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: b(:,:), berr(:), d(:), df(:), dl(:), &
```

```
                                         dlf(:), du(:), du2(:), duf(:),       &
                                         ferr(:), work(:), x(:,:)
      Integer, Allocatable            :: ipiv(:), iwork(:)
!     .. Executable Statements ..
      Write (nout,*) 'F07CHF Example Program Results'
      Write (nout,*)
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, nrhs
      ldb = n
      ldx = n
      Allocate (b(ldb,nrhs),berr(nrhs),d(n),df(n),dl(n-1),dlf(n-1),du(n-1), &
        du2(n-2),duf(n-1),ferr(nrhs),work(3*n),x(ldx,nrhs),ipiv(n),iwork(n))

!     Read the tridiagonal matrix A from data file

      Read (nin,*) du(1:n-1)
      Read (nin,*) d(1:n)
      Read (nin,*) dl(1:n-1)

!     Read the right hand matrix B

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

!     Copy A into DUF, DF and DLF, and copy B into X

      duf(1:n-1) = du(1:n-1)
      df(1:n) = d(1:n)
      dlf(1:n-1) = dl(1:n-1)
      x(1:n,1:nrhs) = b(1:n,1:nrhs)

!     Factorize the copy of the tridiagonal matrix A
!     The NAG name equivalent of dgttrf is f07cdf
      Call dgttrf(n,dlf,df,duf,du2,ipiv,info)

      If (info==0) Then

!       Solve the equations AX = B
!       The NAG name equivalent of dgttrs is f07cef
        Call dgttrs('No transpose',n,nrhs,dlf,df,duf,du2,ipiv,x,ldx,info)

!       Improve the solution and compute error estimates
!       The NAG name equivalent of dgtrfs is f07chf
        Call dgtrfs('No transpose',n,nrhs,dl,d,du,dlf,df,duf,du2,ipiv,b,ldb,x, &
          ldx,ferr,berr,work,iwork,info)

!       Print the solution and the forward and backward error
!       estimates

!       ifail: behaviour on error exit
!              =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
        ifail = 0
        Call x04caf('General',' ',n,nrhs,x,ldx,'Solution(s)',ifail)

        Write (nout,*)
        Write (nout,*) 'Backward errors (machine-dependent)'
        Write (nout,99999) berr(1:nrhs)
        Write (nout,*)
        Write (nout,*) 'Estimated forward error bounds (machine-dependent)'
        Write (nout,99999) ferr(1:nrhs)
      Else
        Write (nout,99998) 'The (', info, ',', info, ')', &
          ' element of the factor U is zero'
      End If

99999 Format ((3X,1P,7E11.1))
99998 Format (1X,A,I3,A,I3,A,A)
    End Program f07chfe
```

10.2 Program Data

```
F07CHF Example Program Data
  5     2                          :Values of N and NRHS
        2.1  -1.0   1.9   8.0
  3.0   2.3  -5.0  -0.9   7.1
  3.4   3.6   7.0  -6.0            :End of matrix A
  2.7   6.6
 -0.5  10.8
  2.6  -3.2
  0.6 -11.2
  2.7  19.1                        :End of matrix B
```

10.3 Program Results

```
 F07CHF Example Program Results

 Solution(s)
             1          2
 1     -4.0000     5.0000
 2      7.0000    -4.0000
 3      3.0000    -3.0000
 4     -4.0000    -2.0000
 5     -3.0000     1.0000

 Backward errors (machine-dependent)
       7.2E-17    5.9E-17

 Estimated forward error bounds (machine-dependent)
       9.4E-15    1.4E-14
```