

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 $A^T X = B$, 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,
1                LDB, X, LDX, FERR, BERR, WORK, IWORK, INFO)
    INTEGER      N, NRHS, IPIV(*), LDB, LDX, IWORK(N), INFO
    double precision DL(*), D(*), DU(*), DLF(*), DF(*), DUF(*), DU2(*),
1                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 $A^T X = B$ for X .

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

- 2: N – INTEGER *Input*
 On entry: n , the order of the matrix A .
 Constraint: $N \geq 0$.

- 3: NRHS – INTEGER *Input*
 On entry: r , the number of right-hand sides, i.e., the number of columns of the matrix B .
 Constraint: NRHS ≥ 0 .

- 4: DL(*) – **double precision** array *Input*
 Note: the dimension of the array DL must be at least $\max(1, N - 1)$.
 On entry: must contain the $(n - 1)$ subdiagonal elements of the matrix A .

- 5: D(*) – **double precision** array *Input*
 Note: the dimension of the array D must be at least $\max(1, N)$.
 On entry: must contain the n diagonal elements of the matrix A .

- 6: DU(*) – **double precision** array *Input*
 Note: the dimension of the array DU must be at least $\max(1, N - 1)$.
 On entry: must contain the $(n - 1)$ superdiagonal elements of the matrix A .

- 7: DLF(*) – **double precision** array *Input*
 Note: the dimension of the array DLF must be at least $\max(1, N - 1)$.
 On entry: must contain the $(n - 1)$ multipliers that define the matrix L of the LU factorization of A .

- 8: DF(*) – **double precision** array *Input*
 Note: the dimension of the array DF must be at least $\max(1, N)$.
 On entry: must contain the n diagonal elements of the upper triangular matrix U from the LU factorization of A .

- 9: DUF(*) – **double precision** array *Input*
 Note: the dimension of the array DUF must be at least $\max(1, N - 1)$.
 On entry: must contain the $(n - 1)$ elements of the first superdiagonal of U .

- 10: DU2(*) – **double precision** array *Input*
 Note: the dimension of the array DU2 must be at least $\max(1, N - 2)$.
 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 $\max(1, N)$.
 On entry: must contain the n pivot indices that define the permutation matrix P . At the i th step, row i of the matrix was interchanged with row IPIV(i), and IPIV(i) must always be either i or $(i + 1)$, IPIV(i) = i indicating that a row interchange was not performed.

- 12: B(LDB,*) – **double precision** array *Input*
Note: the second dimension of the array B must be at least $\max(1, \text{NRHS})$.
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: $\text{LDB} \geq \max(1, N)$.
- 14: X(LDX,*) – **double precision** array *Input/Output*
Note: the second dimension of the array X must be at least $\max(1, \text{NRHS})$.
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: $\text{LDX} \geq \max(1, N)$.
- 16: FERR(NRHS) – **double precision** array *Output*
On exit: estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \text{FERR}(j)$, where $\hat{x}_j$ is the j th 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) – **double precision** 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($3 \times N$) – **double precision** array *Workspace*
- 19: IWORK(N) – INTEGER array *Workspace*
- 20: INFO – INTEGER *Output*
On exit: $\text{INFO} = 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

$\text{INFO} < 0$

If $\text{INFO} = -i$, the i th argument 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 Further Comments

The total number of floating-point operations required to solve the equations $AX = B$ or $A^T X = B$ 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).

9 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.

9.1 Program Text

```
*      F07CHF Example Program Text
*      Mark 21 Release. NAG Copyright 2004.
*      .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, NRHSMX
      PARAMETER        (NMAX=50,NRHSMX=4)
      INTEGER          LDB, LDX
      PARAMETER        (LDB=NMAX,LDX=NMAX)
*      .. Local Scalars ..
      INTEGER          I, IFAIL, INFO, J, N, NRHS
*      .. Local Arrays ..
      DOUBLE PRECISION B(LDB,NRHSMX), BERR(NRHSMX), D(NMAX), DF(NMAX),
+      DL(NMAX-1), DLF(NMAX-1), DU(NMAX-1), DU2(NMAX-2),
+      DUF(NMAX-1), FERR(NRHSMX), WORK(3*NMAX),
+      X(LDX,NRHSMX)
      INTEGER          IPIV(NMAX), IWORK(NMAX)
*      .. External Subroutines ..
      EXTERNAL         DCOPY, DGTRFS, DGTTRF, DGTTRS, F06QFF, X04CAF
*      .. Executable Statements ..
      WRITE (NOUT,*) 'F07CHF Example Program Results'
      WRITE (NOUT,*)
*      Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NRHS
      IF (N.LE.NMAX .AND. NRHS.LE.NRHSMX) THEN
*
*          Read the tridiagonal matrix A from data file
*
          READ (NIN,*) (DU(I),I=1,N-1)
          READ (NIN,*) (D(I),I=1,N)
```

```

      READ (NIN,*) (DL(I),I=1,N-1)
*
*      Read the right hand matrix B
*
      READ (NIN,*) ((B(I,J),J=1,NRHS),I=1,N)
*
*      Copy A into DUF, DF and DLF, and copy B into X
*
      CALL DCOPY(N-1,DU,1,DUF,1)
      CALL DCOPY(N,D,1,DF,1)
      CALL DCOPY(N-1,DL,1,DLF,1)
      CALL F06QFF('General',N,NRHS,B,LDB,X,LDX)
*
*      Factorize the copy of the tridiagonal matrix A
*
      CALL DGTTRF(N,DLF,DF,DUF,DU2,IPIV,INFO)
*
      IF (INFO.EQ.0) THEN
*
*          Solve the equations AX = B
*
*          CALL DGTTRS('No transpose',N,NRHS,DLF,DF,DUF,DU2,IPIV,X,LDX,
+              INFO)
*
*          Improve the solution and compute error estimates
*
*          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 = 0
          CALL X04CAF('General',' ',N,NRHS,X,LDX,'Solution(s)',IFAIL)
*
          WRITE (NOUT,*)
          WRITE (NOUT,*) 'Backward errors (machine-dependent)'
          WRITE (NOUT,99999) (BERR(J),J=1,NRHS)
          WRITE (NOUT,*)
          WRITE (NOUT,*)
+          'Estimated forward error bounds (machine-dependent)'
          WRITE (NOUT,99999) (FERR(J),J=1,NRHS)
          ELSE
              WRITE (NOUT,99998) 'The (', INFO, ', ', INFO, ')',
+              ' element of the factor U is zero'
          END IF
          ELSE
              WRITE (NOUT,*) 'NMAX and/or NRHSMX too small'
          END IF
*
99999 FORMAT ((3X,1P,7E11.1))
99998 FORMAT (1X,A,I3,A,I3,A,A)
      END

```

9.2 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

```

9.3 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
---------	---------
