NAG Library Routine Document

F07UVF (ZTPRFS)

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

F07UVF (ZTPRFS) returns error bounds for the solution of a complex triangular system of linear equations with multiple right-hand sides, $AX = B$, $A^{\mathrm{T}}X = B$ or $A^{\mathrm{H}}X = B$, using packed storage.

2 Specification

```
SUBROUTINE F07UVF (UPLO, TRANS, DIAG, N, NRHS, AP, B, LDB, X, LDX, FERR,    &
                   BERR, WORK, RWORK, INFO)

INTEGER                 N, NRHS, LDB, LDX, INFO
REAL (KIND=nag_wp)      FERR(NRHS), BERR(NRHS), RWORK(N)
COMPLEX (KIND=nag_wp) AP(*), B(LDB,*), X(LDX,*), WORK(2*N)
CHARACTER(1)            UPLO, TRANS, DIAG
```

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

3 Description

F07UVF (ZTPRFS) returns the backward errors and estimated bounds on the forward errors for the solution of a complex triangular system of linear equations with multiple right-hand sides $AX = B$, $A^{\mathrm{T}}X = B$ or $A^{\mathrm{H}}X = B$, using packed storage. The routine handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of F07UVF (ZTPRFS) in terms of a single right-hand side b and solution x.

Given a computed solution x, the routine computes the *component-wise backward error* β. This is the size of the smallest relative perturbation in each element of A and b such that x is the exact solution of a perturbed system

$$(A + \delta A)x = b + \delta b$$
$$\left|\delta a_{ij}\right| \leq \beta\left|a_{ij}\right| \quad \text{and} \quad |\delta b_i| \leq \beta|b_i|.$$

Then the routine estimates a bound for the *component-wise forward error* in the computed solution, defined by:

$$\max_i |x_i - \hat{x}_i| / \max_i |x_i|$$

where $\hat{x}$ is the true solution.

For details of the method, see the F07 Chapter Introduction.

4 References

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Arguments

1: UPLO – CHARACTER(1) *Input*

On entry: specifies whether A is upper or lower triangular.

UPLO = 'U'
A is upper triangular.

UPLO = 'L'
A is lower triangular.

Constraint: UPLO = 'U' or 'L'.

2: TRANS – CHARACTER(1) *Input*

On entry: indicates the form of the equations.

TRANS = 'N'
The equations are of the form $AX = B$.

TRANS = 'T'
The equations are of the form $A^{\mathrm{T}}X = B$.

TRANS = 'C'
The equations are of the form $A^{\mathrm{H}}X = B$.

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

3: DIAG – CHARACTER(1) *Input*

On entry: indicates whether A is a nonunit or unit triangular matrix.

DIAG = 'N'
A is a nonunit triangular matrix.

DIAG = 'U'
A is a unit triangular matrix; the diagonal elements are not referenced and are assumed to be 1.

Constraint: DIAG = 'N' or 'U'.

4: N – INTEGER *Input*

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

Constraint: N ≥ 0.

5: NRHS – INTEGER *Input*

On entry: r, the number of right-hand sides.

Constraint: NRHS ≥ 0.

6: AP$(*)$ – COMPLEX (KIND=nag_wp) array *Input*

Note: the dimension of the array AP must be at least $\max(1, \mathrm{N} \times (\mathrm{N}+1)/2)$.

On entry: the n by n triangular matrix A, packed by columns.

More precisely,

if UPLO = 'U', the upper triangle of A must be stored with element A_{ij} in AP$(i + j(j-1)/2)$ for $i \leq j$;

if UPLO = 'L', the lower triangle of A must be stored with element A_{ij} in AP$(i + (2n-j)(j-1)/2)$ for $i \geq j$.

If DIAG = 'U', the diagonal elements of A are assumed to be 1, and are not referenced; the same storage scheme is used whether DIAG = 'N' or 'U'.

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

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

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

8: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07UVF (ZTPRFS) is called.

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

9: X(LDX, $*$) – COMPLEX (KIND=nag_wp) array *Input*

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

On entry: the n by r solution matrix X, as returned by F07USF (ZTPTRS).

10: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07UVF (ZTPRFS) is called.

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

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

On exit: FERR(j) contains an estimated error bound for the jth solution vector, that is, the jth column of X, for $j = 1, 2, \ldots, r$.

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

On exit: BERR(j) contains the component-wise backward error bound β for the jth solution vector, that is, the jth column of X, for $j = 1, 2, \ldots, r$.

13: WORK($2 \times \mathrm{N}$) – COMPLEX (KIND=nag_wp) array *Workspace*

14: RWORK(N) – REAL (KIND=nag_wp) array *Workspace*

15: 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 bounds returned in FERR are not rigorous, because they are estimated, not computed exactly; but in practice they almost always overestimate the actual error.

8 Parallelism and Performance

F07UVF (ZTPRFS) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F07UVF (ZTPRFS) 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

A call to F07UVF (ZTPRFS), for each right-hand side, involves solving a number of systems of linear equations of the form $Ax = b$ or $A^{\mathrm{H}}x = b$; the number is usually 5 and never more than 11. Each solution involves approximately $4n^2$ real floating-point operations.

The real analogue of this routine is F07UHF (DTPRFS).

10 Example

This example solves the system of equations $AX = B$ and to compute forward and backward error bounds, where

$$A = \begin{pmatrix} 4.78 + 4.56i & 0.00 + 0.00i & 0.00 + 0.00i & 0.00 + 0.00i \\ 2.00 - 0.30i & -4.11 + 1.25i & 0.00 + 0.00i & 0.00 + 0.00i \\ 2.89 - 1.34i & 2.36 - 4.25i & 4.15 + 0.80i & 0.00 + 0.00i \\ -1.89 + 1.15i & 0.04 - 3.69i & -0.02 + 0.46i & 0.33 - 0.26i \end{pmatrix}$$

and

$$B = \begin{pmatrix} -14.78 - 32.36i & -18.02 + 28.46i \\ 2.98 - 2.14i & 14.22 + 15.42i \\ -20.96 + 17.06i & 5.62 + 35.89i \\ 9.54 + 9.91i & -16.46 - 1.73i \end{pmatrix},$$

using packed storage for A.

10.1 Program Text

```
    Program f07uvfe

!     F07UVF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dbf, ztprfs, ztptrs
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
      Character (1), Parameter         :: diag = 'N', trans = 'N'
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, j, ldb, ldx, n, nrhs
      Character (1)                    :: uplo
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: ap(:), b(:,:), work(:), x(:,:)
      Real (Kind=nag_wp), Allocatable  :: berr(:), ferr(:), rwork(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F07UVF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, nrhs
      ldb = n
      ldx = n
      Allocate (ap(n*(n+1)/2),b(ldb,nrhs),work(2*n),x(ldx,n),berr(nrhs),ferr(  &
        nrhs),rwork(n))

!     Read A and B from data file, and copy B to X

      Read (nin,*) uplo
      If (uplo=='U') Then
        Read (nin,*)((ap(i+j*(j-1)/2),j=i,n),i=1,n)
      Else If (uplo=='L') Then
        Read (nin,*)((ap(i+(2*n-j)*(j-1)/2),j=1,i),i=1,n)
      End If
```

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

!     Compute solution in the array X
!     The NAG name equivalent of ztptrs is f07usf
      Call ztptrs(uplo,trans,diag,n,nrhs,ap,x,ldx,info)

!     Compute backward errors and estimated bounds on the
!     forward errors

!     The NAG name equivalent of ztprfs is f07uvf
      Call ztprfs(uplo,trans,diag,n,nrhs,ap,b,ldb,x,ldx,ferr,berr,work,rwork,  &
        info)

!     Print solution

      Write (nout,*)
      Flush (nout)

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call x04dbf('General',' ',n,nrhs,x,ldx,'Bracketed','F7.4','Solution(s)', &
        'Integer',rlabs,'Integer',clabs,80,0,ifail)

      Write (nout,*)
      Write (nout,*) 'Backward errors (machine-dependent)'
      Write (nout,99999) berr(1:nrhs)
      Write (nout,*) 'Estimated forward error bounds (machine-dependent)'
      Write (nout,99999) ferr(1:nrhs)

99999 Format ((5X,1P,4(E11.1,7X)))
    End Program f07uvfe
```

10.2 Program Data

```
F07UVF Example Program Data
  4  2                                                        :Values of N and NRHS
  'L'                                                         :Value of UPLO
 ( 4.78, 4.56)
 ( 2.00,-0.30) (-4.11, 1.25)
 ( 2.89,-1.34) ( 2.36,-4.25) ( 4.15, 0.80)
 (-1.89, 1.15) ( 0.04,-3.69) (-0.02, 0.46) ( 0.33,-0.26)  :End of matrix A
 (-14.78,-32.36) (-18.02, 28.46)
 (  2.98, -2.14) ( 14.22, 15.42)
 (-20.96, 17.06) (  5.62, 35.89)
 (  9.54,  9.91) (-16.46, -1.73)                          :End of matrix B
```

10.3 Program Results

```
 F07UVF Example Program Results

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

 Backward errors (machine-dependent)
         6.2E-17           2.7E-17
 Estimated forward error bounds (machine-dependent)
         2.9E-14           3.2E-14
```