NAG Library Routine Document

F07JVF (ZPTRFS)

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

F07JVF (ZPTRFS) computes error bounds and refines the solution to a complex system of linear equations $AX = B$, where A is an n by n Hermitian positive definite tridiagonal matrix and X and B are n by r matrices, using the modified Cholesky factorization returned by F07JRF (ZPTTRF) and an initial solution returned by F07JSF (ZPTTRS). Iterative refinement is used to reduce the backward error as much as possible.

2 Specification

```
SUBROUTINE F07JVF (UPLO, N, NRHS, D, E, DF, EF, B, LDB, X, LDX, FERR, BERR,     &
                   WORK, RWORK, INFO)

INTEGER                 N, NRHS, LDB, LDX, INFO
REAL (KIND=nag_wp)      D(*), DF(*), FERR(NRHS), BERR(NRHS), RWORK(N)
COMPLEX (KIND=nag_wp)   E(*), EF(*), B(LDB,*), X(LDX,*), WORK(N)
CHARACTER(1)            UPLO
```

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

3 Description

F07JVF (ZPTRFS) should normally be preceded by calls to F07JRF (ZPTTRF) and F07JSF (ZPTTRS). F07JRF (ZPTTRF) computes a modified Cholesky factorization of the matrix A as

$$A = LDL^{\mathrm{H}},$$

where L is a unit lower bidiagonal matrix and D is a diagonal matrix, with positive diagonal elements. F07JSF (ZPTTRS) then utilizes the factorization to compute a solution, $\hat{X}$, to the required equations. Letting $\hat{x}$ denote a column of $\hat{X}$, F07JVF (ZPTRFS) 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.

Note that the modified Cholesky factorization of A can also be expressed as

$$A = U^{\mathrm{H}} DU,$$

where U is unit upper bidiagonal.

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: UPLO – CHARACTER(1) *Input*

On entry: specifies the form of the factorization as follows:

UPLO = 'U'
$A = U^{H}DU$.

UPLO = 'L'
$A = LDL^{H}$.

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

2: N – INTEGER *Input*

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

Constraint: N ≥ 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: 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 of A.

5: E($*$) – COMPLEX (KIND=nag_wp) array *Input*

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

On entry: if UPLO = 'U', E must contain the $(n - 1)$ superdiagonal elements of the matrix A.

If UPLO = 'L', E must contain the $(n - 1)$ subdiagonal elements of the matrix A.

6: 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 diagonal matrix D from the LDL^{T} factorization of A.

7: EF($*$) – COMPLEX (KIND=nag_wp) array *Input*

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

On entry: if UPLO = 'U', EF must contain the $(n - 1)$ superdiagonal elements of the unit upper bidiagonal matrix U from the $U^{H}DU$ factorization of A.

If UPLO = 'L', EF must contain the $(n - 1)$ subdiagonal elements of the unit lower bidiagonal matrix L from the LDL^{H} factorization of A.

8: 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 matrix of right-hand sides B.

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07JVF (ZPTRFS) is called.

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

10: X(LDX,$*$) – COMPLEX (KIND=nag_wp) 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.

11: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07JVF (ZPTRFS) is called.

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

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

On exit: estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|\hat{x}_j\|_\infty \leq \text{FERR}(j)$, 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.

13: 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).

14: WORK(N) – COMPLEX (KIND=nag_wp) array *Workspace*

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

16: 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 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 F07JUF (ZPTCON) can be used to compute the condition number of A.

8 Further Comments

The total number of floating point operations required to solve the equations $AX = B$ is proportional to nr. At most five steps of iterative refinement are performed, but usually only one or two steps are required.

The real analogue of this routine is F07JHF (DPTRFS).

9 Example

This example solves the equations

$$AX = B,$$

where A is the Hermitian positive definite tridiagonal matrix

$$A = \begin{pmatrix} 16.0 & 16.0 - 16.0i & 0 & 0 \\ 16.0 + 16.0i & 41.0 & 18.0 + 9.0i & 0 \\ 0 & 18.0 - 9.0i & 46.0 & 1.0 + 4.0i \\ 0 & 0 & 1.0 - 4.0i & 21.0 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 64.0 + 16.0i & -16.0 - 32.0i \\ 93.0 + 62.0i & 61.0 - 66.0i \\ 78.0 - 80.0i & 71.0 - 74.0i \\ 14.0 - 27.0i & 35.0 + 15.0i \end{pmatrix}.$$

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

9.1 Program Text

```
    Program f07jvfe

!     F07JVF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: nag_wp, x04dbf, zptrfs, zpttrf, zpttrs
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                         :: i, ifail, info, ldb, ldx, n, nrhs
!     .. Local Arrays ..
      Complex (Kind=nag_wp), Allocatable :: b(:,:), e(:), ef(:), work(:), x(:,:)
      Real (Kind=nag_wp), Allocatable  :: berr(:), d(:), df(:), ferr(:),       &
                                          rwork(:)
      Character (1)                   :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F07JVF 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),e(n-1),ef(n-1),work(n),x(ldx,nrhs),berr(nrhs), &
        d(n),df(n),ferr(nrhs),rwork(n))

!     Read the lower bidiagonal part of the tridiagonal matrix A from
!     data file

      Read (nin,*) d(1:n)
      Read (nin,*) e(1:n-1)
```

```
!     Read the right hand matrix B

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

!     Copy A into DF and EF, and copy B into X
      df(1:n) = d(1:n)
      ef(1:n-1) = e(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 zptrrf is f07jrf
      Call zpttrf(n,df,ef,info)

      If (info==0) Then

!       Solve the equations AX = B
!       The NAG name equivalent of zptrrs is f07jsf
        Call zpttrs('Lower',n,nrhs,df,ef,x,ldx,info)

!       Improve the solution and compute error estimates
!       The NAG name equivalent of zptrfs is f07jvf
        Call zptrfs('Lower',n,nrhs,d,e,df,ef,b,ldb,x,ldx,ferr,berr,work,rwork, &
          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 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,*)
        Write (nout,*) 'Estimated forward error bounds (machine-dependent)'
        Write (nout,99999) ferr(1:nrhs)
      Else
        Write (nout,99998) 'The leading minor of order ', info, &
          ' is not positive definite'
      End If

99999 Format ((3X,1P,7E11.1))
99998 Format (1X,A,I3,A)
    End Program f07jvfe
```

9.2 Program Data

```
F07JVF Example Program Data
    4            2                                 :Values of N and NRHS
   16.0          41.0          46.0          21.0 :End of diagonal D
 ( 16.0, 16.0) ( 18.0, -9.0) (  1.0, -4.0)         :End of sub-diagonal E
 ( 64.0, 16.0) (-16.0,-32.0)
 ( 93.0, 62.0) ( 61.0,-66.0)
 ( 78.0,-80.0) ( 71.0,-74.0)
 ( 14.0,-27.0) ( 35.0, 15.0)                       :End of matrix B
```

9.3 Program Results

```
 F07JVF Example Program Results

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

```
 Backward errors (machine-dependent)
       0.0E+00    0.0E+00

 Estimated forward error bounds (machine-dependent)
       9.0E-12    6.1E-12
```