# NAG Library Routine Document

# F07PVF (ZHPRFS)

**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

F07PVF (ZHPRFS) returns error bounds for the solution of a complex Hermitian indefinite system of linear equations with multiple right-hand sides, $AX = B$, using packed storage. It improves the solution by iterative refinement, in order to reduce the backward error as much as possible.

## 2 Specification

```
 SUBROUTINE F07PVF(UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X, LDX, FERR,
1                  BERR, WORK, RWORK, INFO)
 INTEGER           N, NRHS, IPIV(*), LDB, LDX, INFO
 double precision  FERR(NRHS), BERR(NRHS), RWORK(N)
 complex*16        AP(*), AFP(*), B(LDB,*), X(LDX,*), WORK(2*N)
 CHARACTER*1       UPLO
```

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

## 3 Description

F07PVF (ZHPRFS) returns the backward errors and estimated bounds on the forward errors for the solution of a complex Hermitian indefinite system of linear equations with multiple right-hand sides $AX = B$, 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 F07PVF (ZHPRFS) 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* $\beta$. 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

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

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 Parameters

1: UPLO – CHARACTER*1 *Input*

*On entry*: indicates whether the upper or lower triangular part of $A$ is stored and how $A$ is to be factorized.

UPLO = 'U'

The upper triangular part of $A$ is stored and $A$ is factorized as $PUDU^{\mathrm{H}}P^{\mathrm{T}}$, where $U$ is upper triangular.

UPLO = 'L'

The lower triangular part of $A$ is stored and $A$ is factorized as $PLDL^{\mathrm{H}}P^{\mathrm{T}}$, where $L$ is lower triangular.

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

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.

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

4: AP$(*)$ – ***complex*16*** 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 $n$ original Hermitian matrix $A$ as supplied to F07PRF (ZHPTRF).

5: AFP$(*)$ – ***complex*16*** array *Input*

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

*On entry*: the factorization of $A$ stored in packed form, as returned by F07PRF (ZHPTRF).

6: IPIV$(*)$ – INTEGER array *Input*

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

*On entry*: details of the interchanges and the block structure of $D$, as returned by F07PRF (ZHPTRF).

7: B(LDB,$*$) – ***complex*16*** 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 F07PVF (ZHPRFS) is called.

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

9: X(LDX,$*$) – ***complex*16*** array *Input/Output*

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

*On entry*: the $n$ by $r$ solution matrix $X$, as returned by F07PSF (ZHPTRS).

*On exit*: the improved solution matrix $X$.

10: LDX – INTEGER *Input*

*On entry*: the first dimension of the array X as declared in the (sub)program from which F07PVF (ZHPRFS) is called.

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

11: FERR(NRHS) – ***double precision*** array *Output*

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

12: BERR(NRHS) – ***double precision*** array *Output*

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

13: WORK($2 \times$ N) – ***complex*16*** array *Workspace*

14: RWORK(N) – ***double precision*** array *Workspace*

15: 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 $i$th parameter 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 Further Comments

For each right-hand side, computation of the backward error involves a minimum of $16n^2$ real floating-point operations. Each step of iterative refinement involves an additional $24n^2$ real operations. At most five steps of iterative refinement are performed, but usually only 1 or 2 steps are required.

Estimating the forward error involves solving a number of systems of linear equations of the form $Ax = b$; the number is usually 5 and never more than 11. Each solution involves approximately $8n^2$ real operations.

The real analogue of this routine is F07PHF (DSPRFS).

## 9 Example

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

$$A = \begin{pmatrix} -1.36 + 0.00i & 1.58 + 0.90i & 2.21 - 0.21i & 3.91 + 1.50i \\ 1.58 - 0.90i & -8.87 + 0.00i & -1.84 - 0.03i & -1.78 + 1.18i \\ 2.21 + 0.21i & -1.84 + 0.03i & -4.63 + 0.00i & 0.11 + 0.11i \\ 3.91 - 1.50i & -1.78 - 1.18i & 0.11 - 0.11i & -1.84 + 0.00i \end{pmatrix}$$

and

$$B = \begin{pmatrix} 7.79 + 5.48i & -35.39 + 18.01i \\ -0.77 - 16.05i & 4.23 - 70.02i \\ -9.58 + 3.88i & -24.79 - 8.40i \\ 2.98 - 10.18i & 28.68 - 39.89i \end{pmatrix}.$$

Here $A$ is Hermitian indefinite, stored in packed form, and must first be factorized by F07PRF (ZHPTRF).

## 9.1 Program Text

```
*     F07PVF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, NRHMAX, LDB, LDX
      PARAMETER        (NMAX=8,NRHMAX=NMAX,LDB=NMAX,LDX=NMAX)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, INFO, J, N, NRHS
      CHARACTER        UPLO
*     .. Local Arrays ..
      COMPLEX *16      AFP(NMAX*(NMAX+1)/2), AP(NMAX*(NMAX+1)/2),
     +                 B(LDB,NRHMAX), WORK(2*NMAX), X(LDX,NMAX)
      DOUBLE PRECISION BERR(NRHMAX), FERR(NRHMAX), RWORK(NMAX)
      INTEGER          IPIV(NMAX)
      CHARACTER        CLABS(1), RLABS(1)
*     .. External Subroutines ..
      EXTERNAL         F06TFF, X04DBF, ZHPRFS, ZHPTRF, ZHPTRS
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F07PVF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, NRHS
      IF (N.LE.NMAX .AND. NRHS.LE.NRHMAX) THEN
*
*        Read A and B from data file, and copy A to AFP and B to X
*
         READ (NIN,*) UPLO
         IF (UPLO.EQ.'U') THEN
            READ (NIN,*) ((AP(I+J*(J-1)/2),J=I,N),I=1,N)
         ELSE IF (UPLO.EQ.'L') THEN
            READ (NIN,*) ((AP(I+(2*N-J)*(J-1)/2),J=1,I),I=1,N)
         END IF
         READ (NIN,*) ((B(I,J),J=1,NRHS),I=1,N)
         DO 20 I = 1, N*(N+1)/2
            AFP(I) = AP(I)
   20    CONTINUE
         CALL F06TFF('General',N,NRHS,B,LDB,X,LDX)
*
*        Factorize A in the array AFP
*
         CALL ZHPTRF(UPLO,N,AFP,IPIV,INFO)
*
         WRITE (NOUT,*)
         IF (INFO.EQ.0) THEN
*
*           Compute solution in the array X
*
            CALL ZHPTRS(UPLO,N,NRHS,AFP,IPIV,X,LDX,INFO)
*
*           Improve solution, and compute backward errors and
```

```
*           estimated bounds on the forward errors
*
            CALL ZHPRFS(UPLO,N,NRHS,AP,AFP,IPIV,B,LDB,X,LDX,FERR,BERR,
     +                  WORK,RWORK,INFO)
*
*           Print solution
*
            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(J),J=1,NRHS)
            WRITE (NOUT,*)
     +        'Estimated forward error bounds (machine-dependent)'
            WRITE (NOUT,99999) (FERR(J),J=1,NRHS)
         ELSE
            WRITE (NOUT,*) 'The factor D is singular'
         END IF
       END IF
*
99999 FORMAT ((5X,1P,4(E11.1,7X)))
      END
```

## 9.2 Program Data

```
F07PVF Example Program Data
  4  2                                                        :Values of N and NRHS
  'L'                                                         :Value of UPLO
 (-1.36, 0.00)
 ( 1.58,-0.90) (-8.87, 0.00)
 ( 2.21, 0.21) (-1.84, 0.03) (-4.63, 0.00)
 ( 3.91,-1.50) (-1.78,-1.18) ( 0.11,-0.11) (-1.84, 0.00)  :End of matrix A
 ( 7.79,  5.48) (-35.39, 18.01)
 (-0.77,-16.05) (  4.23,-70.02)
 (-9.58,  3.88) (-24.79, -8.40)
 ( 2.98,-10.18) ( 28.68,-39.89)                           :End of matrix B
```

## 9.3 Program Results

```
 F07PVF Example Program Results

 Solution(s)
                     1                   2
 1  ( 1.0000,-1.0000) ( 3.0000,-4.0000)
 2  (-1.0000, 2.0000) (-1.0000, 5.0000)
 3  ( 3.0000,-2.0000) ( 7.0000,-2.0000)
 4  ( 2.0000, 1.0000) (-8.0000, 6.0000)

 Backward errors (machine-dependent)
         8.0E-17           7.0E-17
 Estimated forward error bounds (machine-dependent)
         2.5E-15           3.0E-15
```