

## NAG Library Routine Document

### F07PHF (DSPRFS)

**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

F07PHF (DSPRFS) returns error bounds for the solution of a real symmetric 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 F07PHF (UPLO, N, NRHS, AP, AFP, IPIV, B, LDB, X, LDX, FERR,      &
                  BERR, WORK, IWORK, INFO)
INTEGER                N, NRHS, IPIV(*), LDB, LDX, IWORK(N), INFO
REAL (KIND=nag_wp)    AP(*), AFP(*), B(LDB,*), X(LDX,*), FERR(NRHS),    &
                  BERR(NRHS), WORK(3*N)
CHARACTER(1)          UPLO

```

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

#### 3 Description

F07PHF (DSPRFS) returns the backward errors and estimated bounds on the forward errors for the solution of a real symmetric 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 F07PHF (DSPRFS) 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 Arguments

- 1: UPLO – CHARACTER(1) *Input*  
*On entry:* specifies 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  $PUDU^T P^T$ , where  $U$  is upper triangular.  
 UPLO = 'L'  
 The lower triangular part of  $A$  is stored and  $A$  is factorized as  $PLDL^T P^T$ , where  $L$  is lower triangular.  
*Constraint:* UPLO = 'U' or 'L'.
- 2: N – INTEGER *Input*  
*On entry:*  $n$ , the order of the matrix  $A$ .  
*Constraint:*  $N \geq 0$ .
- 3: NRHS – INTEGER *Input*  
*On entry:*  $r$ , the number of right-hand sides.  
*Constraint:* NRHS  $\geq 0$ .
- 4: AP(\*) – REAL (KIND=nag\_wp) array *Input*  
**Note:** the dimension of the array AP must be at least  $\max(1, N \times (N + 1)/2)$ .  
*On entry:* the  $n$  by  $n$  original symmetric matrix  $A$  as supplied to F07PDF (DSPTRF).
- 5: AFP(\*) – REAL (KIND=nag\_wp) array *Input*  
**Note:** the dimension of the array AFP must be at least  $\max(1, N \times (N + 1)/2)$ .  
*On entry:* the factorization of  $A$  stored in packed form, as returned by F07PDF (DSPTRF).
- 6: IPIV(\*) – INTEGER array *Input*  
**Note:** the dimension of the array IPIV must be at least  $\max(1, N)$ .  
*On entry:* details of the interchanges and the block structure of  $D$ , as returned by F07PDF (DSPTRF).
- 7: 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  $r$  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 F07PHF (DSPRFS) is called.  
*Constraint:* LDB  $\geq \max(1, N)$ .
- 9: 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  $r$  solution matrix  $X$ , as returned by F07PEF (DSPTRS).  
*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 F07PHF (DSPRFS) is called.  
*Constraint:*  $LDX \geq \max(1, N)$ .
- 11: FERR(NRHS) – REAL (KIND=nag\_wp) 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  $j = 1, 2, \dots, r$ .
- 12: BERR(NRHS) – REAL (KIND=nag\_wp) 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  $j = 1, 2, \dots, r$ .
- 13: WORK( $3 \times N$ ) – REAL (KIND=nag\_wp) array *Workspace*
- 14: IWORK( $N$ ) – INTEGER 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

F07PHF (DSPRFS) is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F07PHF (DSPRFS) 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

For each right-hand side, computation of the backward error involves a minimum of  $4n^2$  floating-point operations. Each step of iterative refinement involves an additional  $6n^2$  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 4 or 5 and never more than 11. Each solution involves approximately  $2n^2$  operations.

The complex analogues of this routine are F07PVF (ZHPRFS) for Hermitian matrices and F07QVF (ZSPRFS) for symmetric matrices.

## 10 Example

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

$$A = \begin{pmatrix} 2.07 & 3.87 & 4.20 & -1.15 \\ 3.87 & -0.21 & 1.87 & 0.63 \\ 4.20 & 1.87 & 1.15 & 2.06 \\ -1.15 & 0.63 & 2.06 & -1.81 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} -9.50 & 27.85 \\ -8.38 & 9.90 \\ -6.07 & 19.25 \\ -0.96 & 3.93 \end{pmatrix}.$$

Here  $A$  is symmetric indefinite, stored in packed form, and must first be factorized by F07PDF (DSPTRF).

### 10.1 Program Text

```

Program f07phfe

!      F07PHF Example Program Text

!      Mark 26 Release. NAG Copyright 2016.

!      .. Use Statements ..
Use nag_library, Only: dsprfs, dsptrf, dsptrs, nag_wp, x04caf
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
Integer                     :: i, ifail, info, j, ldb, ldx, len, n, &
                             nrhs
Character (1)               :: uplo
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable :: afp(:), ap(:), b(:,,:), berr(:),      &
                             ferr(:), work(:), x(:,,:)
Integer, Allocatable        :: ipiv(:), iwork(:)
!      .. Executable Statements ..
Write (nout,*) 'F07PHF Example Program Results'
!      Skip heading in data file
Read (nin,*)
Read (nin,*) n, nrhs
ldb = n
ldx = n
len = n*(n+1)/2
Allocate (afp(len),ap(len),b(ldb,nrhs),berr(nrhs),ferr(nrhs),work(3*n), &
         x(ldx,n),ipiv(n),iwork(n))

!      Read A and B from data file, and copy A to AFP and 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)

afp(1:len) = ap(1:len)
x(1:n,1:nrhs) = b(1:n,1:nrhs)

!      Factorize A in the array AFP
!      The NAG name equivalent of dsptrf is f07pdf
Call dsptrf(uplo,n,afp,ipiv,info)

Write (nout,*)
Flush (nout)

```

```

      If (info==0) Then

!       Compute solution in the array X
!       The NAG name equivalent of dsptrs is f07pef
!       Call dsptrs(uplo,n,nrhs,afp,ipiv,x,ldx,info)

!       Improve solution, and compute backward errors and
!       estimated bounds on the forward errors

!       The NAG name equivalent of dsprfs is f07phf
!       Call dsprfs(uplo,n,nrhs,ap,afp,ipiv,b,ldb,x,ldx,ferr,berr,work,iwork, &
!           info)

!       Print solution

!       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,*) 'Estimated forward error bounds (machine-dependent)'
!       Write (nout,99999) ferr(1:nrhs)
!     Else
!       Write (nout,*) 'The factor D is singular'
!     End If

99999 Format ((3X,1P,7E11.1))
      End Program f07phfe

```

## 10.2 Program Data

F07PHF Example Program Data

```

  4  2           :Values of N and NRHS
  'L'           :Value of UPLO
  2.07
  3.87 -0.21
  4.20  1.87  1.15
-1.15  0.63  2.06 -1.81   :End of matrix A
-9.50 27.85
-8.38  9.90
-6.07 19.25
-0.96  3.93           :End of matrix B

```

## 10.3 Program Results

F07PHF Example Program Results

Solution(s)

	1	2
1	-4.0000	1.0000
2	-1.0000	4.0000
3	2.0000	3.0000
4	5.0000	2.0000

Backward errors (machine-dependent)

9.9E-17 8.3E-17

Estimated forward error bounds (machine-dependent)

2.4E-14 3.2E-14

---