NAG Library Routine Document

F07HHF (DPBRFS)

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

F07HHF (DPBRFS) returns error bounds for the solution of a real symmetric positive definite band system of linear equations with multiple right-hand sides, $AX = B$. It improves the solution by iterative refinement, in order to reduce the backward error as much as possible.

2 Specification

```
SUBROUTINE F07HHF (UPLO, N, KD, NRHS, AB, LDAB, AFB, LDAFB, B, LDB, X, LDX,     &
                   FERR, BERR, WORK, IWORK, INFO)

INTEGER            N, KD, NRHS, LDAB, LDAFB, LDB, LDX, IWORK(N), INFO
REAL (KIND=nag_wp) AB(LDAB,*), AFB(LDAFB,*), B(LDB,*), X(LDX,*),                &
                   FERR(NRHS), BERR(NRHS), WORK(3*N)
CHARACTER(1)       UPLO
```

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

3 Description

F07HHF (DPBRFS) returns the backward errors and estimated bounds on the forward errors for the solution of a real symmetric positive definite band system of linear equations with multiple right-hand sides $AX = B$. The routine handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of F07HHF (DPBRFS) 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 Parameters

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 $U^{\mathrm{T}}U$, where U is upper triangular.

UPLO = 'L'

The lower triangular part of A is stored and A is factorized as LL^{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: KD – INTEGER *Input*

On entry: k_d, the number of superdiagonals or subdiagonals of the matrix A.

Constraint: $\mathrm{KD} \geq 0$.

4: NRHS – INTEGER *Input*

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

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

5: AB(LDAB,$*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: the n by n original symmetric positive definite band matrix A as supplied to F07HDF (DPBTRF).

6: LDAB – INTEGER *Input*

On entry: the first dimension of the array AB as declared in the (sub)program from which F07HHF (DPBRFS) is called.

Constraint: $\mathrm{LDAB} \geq \mathrm{KD} + 1$.

7: AFB(LDAFB,$*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: the Cholesky factor of A, as returned by F07HDF (DPBTRF).

8: LDAFB – INTEGER *Input*

On entry: the first dimension of the array AFB as declared in the (sub)program from which F07HHF (DPBRFS) is called.

Constraint: $\mathrm{LDAFB} \geq \mathrm{KD} + 1$.

9: B(LDB,$*$) – REAL (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.

10: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07HHF (DPBRFS) is called.

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

11: X(LDX,$*$) – REAL (KIND=nag_wp) array *Input/Output*

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 F07HEF (DPBTRS).

On exit: the improved solution matrix X.

12: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07HHF (DPBRFS) is called.

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

13: 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 $j = 1, 2, \ldots, r$.

14: 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 $j = 1, 2, \ldots, r$.

15: WORK($3 \times$ N) – REAL (KIND=nag_wp) array *Workspace*

16: IWORK(N) – INTEGER array *Workspace*

17: 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 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 $8nk$ floating point operations. Each step of iterative refinement involves an additional $12nk$ operations. This assumes $n \gg k$. At most five steps of iterative refinement are performed, but usually only one or two 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 $4nk$ operations.

The complex analogue of this routine is F07HVF (ZPBRFS).

9 Example

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

$$A = \begin{pmatrix} 5.49 & 2.68 & 0.00 & 0.00 \\ 2.68 & 5.63 & -2.39 & 0.00 \\ 0.00 & -2.39 & 2.60 & -2.22 \\ 0.00 & 0.00 & -2.22 & 5.17 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 22.09 & 5.10 \\ 9.31 & 30.81 \\ -5.24 & -25.82 \\ 11.83 & 22.90 \end{pmatrix}.$$

Here A is symmetric and positive definite, and is treated as a band matrix, which must first be factorized by F07HDF (DPBTRF).

9.1 Program Text

```
    Program f07hhfe

!     F07HHF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dpbrfs, dpbtrf, dpbtrs, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: zero = 0.0E0_nag_wp
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, j, kd, ldab, ldafb,  &
                                          ldb, ldx, n, nrhs
      Character (1)                    :: uplo
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: ab(:,:), afb(:,:), b(:,:), berr(:),  &
                                          ferr(:), work(:), x(:,:)
      Integer, Allocatable             :: iwork(:)
!     .. Intrinsic Procedures ..
      Intrinsic                        :: max, min
!     .. Executable Statements ..
      Write (nout,*) 'F07HHF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, kd, nrhs
      ldab = kd + 1
      ldafb = kd + 1
      ldb = n
      ldx = n
      Allocate (ab(ldab,n),afb(ldafb,n),b(ldb,nrhs),berr(nrhs),ferr(nrhs), &
        work(3*n),x(ldx,n),iwork(n))

!     Set A to zero to avoid referencing uninitialized elements

      ab(1:kd+1,1:n) = zero

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

      Read (nin,*) uplo
      If (uplo=='U') Then
        Do i = 1, n
          Read (nin,*)(ab(kd+1+i-j,j),j=i,min(n,i+kd))
        End Do
      Else If (uplo=='L') Then
        Do i = 1, n
          Read (nin,*)(ab(1+i-j,j),j=max(1,i-kd),i)
        End Do
      End If
```

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

      afb(1:kd+1,1:n) = ab(1:kd+1,1:n)
      x(1:n,1:nrhs) = b(1:n,1:nrhs)

!     Factorize A in the array AFB
!     The NAG name equivalent of dpbtrf is f07hdf
      Call dpbtrf(uplo,n,kd,afb,ldafb,info)

      Write (nout,*)
      Flush (nout)
      If (info==0) Then

!       Compute solution in the array X
!       The NAG name equivalent of dpbtrs is f07hef
        Call dpbtrs(uplo,n,kd,nrhs,afb,ldafb,x,ldx,info)

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

!       The NAG name equivalent of dpbrfs is f07hhf
        Call dpbrfs(uplo,n,kd,nrhs,ab,ldab,afb,ldafb,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,*) 'A is not positive definite'
      End If

99999 Format ((3X,1P,7E11.1))
    End Program f07hhfe
```

9.2 Program Data

```
F07HHF Example Program Data
  4  1  2                    :Values of N, KD and NRHS
  'L'                         :Value of UPLO
  5.49
  2.68   5.63
        -2.39   2.60
               -2.22   5.17   :End of matrix A
 22.09   5.10
  9.31  30.81
 -5.24 -25.82
 11.83  22.90                 :End of matrix B
```

9.3 Program Results

```
 F07HHF Example Program Results

 Solution(s)
             1          2
 1      5.0000    -2.0000
 2     -2.0000     6.0000
 3     -3.0000    -1.0000
 4      1.0000     4.0000
```

```
 Backward errors (machine-dependent)
       4.4E-17    9.2E-17
 Estimated forward error bounds (machine-dependent)
       2.0E-14    2.9E-14
```
