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,
1                  LDX, FERR, BERR, WORK, IWORK, INFO)
 INTEGER           N, KD, NRHS, LDAB, LDAFB, LDB, LDX, IWORK(N), INFO
 double precision  AB(LDAB,*), AFB(LDAFB,*), B(LDB,*), X(LDX,*),
1                  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: 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 $U^{\mathrm{T}}U$, 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: N ≥ 0.

3: KD – INTEGER *Input*

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

Constraint: KD ≥ 0.

4: NRHS – INTEGER *Input*

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

Constraint: NRHS ≥ 0.

5: AB(LDAB,$*$) – ***double precision*** 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: LDAB $\geq$ KD $+ 1$.

7: AFB(LDAFB,$*$) – ***double precision*** 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: LDAFB $\geq$ KD $+ 1$.

9: B(LDB,$*$) – ***double precision*** array *Input*

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

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

11: X(LDX,$*$) – ***double precision*** 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 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: $\text{LDX} \geq \max(1, \text{N})$.

13: FERR(NRHS) – ***double precision*** 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) – ***double precision*** 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($3 \times \text{N}$) – ***double precision*** array *Workspace*

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

17: INFO – INTEGER *Output*

On exit: $\text{INFO} = 0$ unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

$\text{INFO} < 0$

If $\text{INFO} = -i$, 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

```
*     F07HHF Example Program Text
*     Mark 15 Release. NAG Copyright 1991.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      DOUBLE PRECISION ZERO
      PARAMETER        (ZERO=0.0D0)
      INTEGER          NMAX, NRHMAX, KDMAX, LDAB, LDAFB, LDB, LDX
      PARAMETER        (NMAX=8,NRHMAX=NMAX,KDMAX=8,LDAB=KDMAX+1,
     +                 LDAFB=KDMAX+1,LDB=NMAX,LDX=NMAX)
*     .. Local Scalars ..
      INTEGER          I, IFAIL, INFO, J, KD, N, NRHS
      CHARACTER        UPLO
*     .. Local Arrays ..
      DOUBLE PRECISION AB(LDAB,NMAX), AFB(LDAFB,NMAX), B(LDB,NRHMAX),
     +                 BERR(NRHMAX), FERR(NRHMAX), WORK(3*NMAX),
     +                 X(LDX,NMAX)
      INTEGER          IWORK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         DPBRFS, DPBTRF, DPBTRS, F06QFF, F06QHF, X04CAF
*     .. Intrinsic Functions ..
      INTRINSIC        MAX, MIN
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F07HHF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, KD, NRHS
      IF (N.LE.NMAX .AND. KD.LE.KDMAX .AND. NRHS.LE.NRHMAX) THEN
*
*        Set A to zero to avoid referencing uninitialized elements
*
         CALL F06QHF('General',KD+1,N,ZERO,ZERO,AB,LDAB)
*
*        Read A and B from data file, and copy A to AFB and B to X
*
         READ (NIN,*) UPLO
         IF (UPLO.EQ.'U') THEN
            DO 20 I = 1, N
               READ (NIN,*) (AB(KD+1+I-J,J),J=I,MIN(N,I+KD))
   20       CONTINUE
         ELSE IF (UPLO.EQ.'L') THEN
            DO 40 I = 1, N
               READ (NIN,*) (AB(1+I-J,J),J=MAX(1,I-KD),I)
   40       CONTINUE
```

```
         END IF
         READ (NIN,*) ((B(I,J),J=1,NRHS),I=1,N)
*
         CALL F06QFF('General',KD+1,N,AB,LDAB,AFB,LDAFB)
*
         CALL F06QFF('General',N,NRHS,B,LDB,X,LDX)
*
*        Factorize A in the array AFB
*
         CALL DPBTRF(UPLO,N,KD,AFB,LDAFB,INFO)
*
         WRITE (NOUT,*)
         IF (INFO.EQ.0) THEN
*
*           Compute solution in the array X
*
            CALL DPBTRS(UPLO,N,KD,NRHS,AFB,LDAFB,X,LDX,INFO)
*
*           Improve solution, and compute backward errors and
*           estimated bounds on the forward errors
*
            CALL DPBRFS(UPLO,N,KD,NRHS,AB,LDAB,AFB,LDAFB,B,LDB,X,LDX,
     +                  FERR,BERR,WORK,IWORK,INFO)
*
*           Print solution
*
            IFAIL = 0
*
            CALL X04CAF('General',' ',N,NRHS,X,LDX,'Solution(s)',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,*) 'A is not positive-definite'
         END IF
      END IF
*
99999 FORMAT ((3X,1P,7E11.1))
      END
```

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
```