NAG Library Routine Document

F07GHF (DPPRFS)

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

F07GHF (DPPRFS) returns error bounds for the solution of a real symmetric positive-definite 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 F07GHF(UPLO, N, NRHS, AP, AFP, B, LDB, X, LDX, FERR, BERR,
1                  WORK, IWORK, INFO)
 INTEGER           N, NRHS, LDB, LDX, IWORK(N), INFO
 double precision  AP(*), AFP(*), B(LDB,*), X(LDX,*), FERR(NRHS),
1                  BERR(NRHS), WORK(3*N)
 CHARACTER*1       UPLO
```

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

3 Description

F07GHF (DPPRFS) returns the backward errors and estimated bounds on the forward errors for the solution of a real symmetric positive-definite 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 F07GHF (DPPRFS) 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

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

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 $U^{T}U$, 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: NRHS – INTEGER *Input*

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

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

4: AP$(*)$ – ***double precision*** 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 symmetric positive-definite matrix A as supplied to F07GDF (DPPTRF).

5: AFP$(*)$ – ***double precision*** 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 Cholesky factor of A stored in packed form, as returned by F07GDF (DPPTRF).

6: B(LDB,$*$) – ***double precision*** 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.

7: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F07GHF (DPPRFS) is called.

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

8: X(LDX,$*$) – ***double precision*** 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 F07GEF (DPPTRS).

On exit: the improved solution matrix X.

9: LDX – INTEGER *Input*

On entry: the first dimension of the array X as declared in the (sub)program from which F07GHF (DPPRFS) is called.

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

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

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

12: WORK$(3 \times \mathrm{N})$ – ***double precision*** array *Workspace*

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

14: INFO – INTEGER *Output*

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

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

$\mathrm{INFO} < 0$

If $\mathrm{INFO} = -i$, 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 $4n^2$ 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 analogue of this routine is F07GVF (ZPPRFS).

9 Example

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

$$A = \begin{pmatrix} 4.16 & -3.12 & 0.56 & -0.10 \\ -3.12 & 5.03 & -0.83 & 1.18 \\ 0.56 & -0.83 & 0.76 & 0.34 \\ -0.10 & 1.18 & 0.34 & 1.18 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 8.70 & 8.30 \\ -13.35 & 2.13 \\ 1.89 & 1.61 \\ -4.14 & 5.00 \end{pmatrix}.$$

Here A is symmetric positive-definite, stored in packed form, and must first be factorized by F07GDF (DPPTRF).

9.1 Program Text

```
*     F07GHF 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 ..
      DOUBLE PRECISION AFP(NMAX*(NMAX+1)/2), AP(NMAX*(NMAX+1)/2),
     +                 B(LDB,NRHMAX), BERR(NRHMAX), FERR(NRHMAX),
     +                 WORK(3*NMAX), X(LDX,NMAX)
      INTEGER          IWORK(NMAX)
*     .. External Subroutines ..
      EXTERNAL         DPPRFS, DPPTRF, DPPTRS, F06QFF, X04CAF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'F07GHF 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 F06QFF('General',N,NRHS,B,LDB,X,LDX)
*
*        Factorize A in the array AFP
*
         CALL DPPTRF(UPLO,N,AFP,INFO)
*
         WRITE (NOUT,*)
         IF (INFO.EQ.0) THEN
*
*           Compute solution in the array X
*
            CALL DPPTRS(UPLO,N,NRHS,AFP,X,LDX,INFO)
*
*           Improve solution, and compute backward errors and
*           estimated bounds on the forward errors
*
            CALL DPPRFS(UPLO,N,NRHS,AP,AFP,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

```
F07GHF Example Program Data
  4  2                          :Values of N and NRHS
  'L'                           :Value of UPLO
  4.16
 -3.12   5.03
  0.56  -0.83   0.76
 -0.10   1.18   0.34   1.18   :End of matrix A
  8.70   8.30
-13.35   2.13
  1.89   1.61
 -4.14   5.00                   :End of matrix B
```

9.3 Program Results

```
 F07GHF Example Program Results

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

 Backward errors (machine-dependent)
       5.1E-17    6.1E-17
 Estimated forward error bounds (machine-dependent)
       2.3E-14    2.3E-14
```