NAG Library Routine Document

F07MHF (DSYRFS)

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

F07MHF (DSYRFS) returns error bounds for the solution of a real symmetric indefinite 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 F07MHF (UPLO, N, NRHS, A, LDA, AF, LDAF, IPIV, B, LDB, X, LDX,       &
                   FERR, BERR, WORK, IWORK, INFO)

INTEGER            N, NRHS, LDA, LDAF, IPIV(*), LDB, LDX, IWORK(N), INFO
REAL (KIND=nag_wp) A(LDA,*), AF(LDAF,*), B(LDB,*), X(LDX,*), FERR(NRHS),        &
                   BERR(NRHS), WORK(3*N)
CHARACTER(1)       UPLO
```

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

3 Description

F07MHF (DSYRFS) 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 $AX = B$. The routine handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of F07MHF (DSYRFS) 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

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

Then the routine estimates a bound for the *component-wise forward error* in the computed solution, defined by:

$$\max_i\left|x_i - \hat{x}_i\right| / \max_i\left|x_i\right|$$

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

UPLO = 'L'
: The lower triangular part of A is stored and A is factorized as $PLDL^{\mathrm{T}}P^{\mathrm{T}}$, 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: A(LDA,$*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: the n by n original symmetric matrix A as supplied to F07MDF (DSYTRF).

5: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F07MHF (DSYRFS) is called.

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

6: AF(LDAF,$*$) – REAL (KIND=nag_wp) array *Input*

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

On entry: details of the factorization of A, as returned by F07MDF (DSYTRF).

7: LDAF – INTEGER *Input*

On entry: the first dimension of the array AF as declared in the (sub)program from which F07MHF (DSYRFS) is called.

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

8: 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 F07MDF (DSYTRF).

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 F07MHF (DSYRFS) 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 F07MEF (DSYTRS).

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 F07MHF (DSYRFS) 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$(3 \times \mathrm{N})$ – 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 $4n^2$ 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 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 $2n^2$ operations.

The complex analogues of this routine are F07MVF (ZHERFS) for Hermitian matrices and F07NVF (ZSYRFS) for symmetric matrices.

9 Example

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

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

Here A is symmetric indefinite and must first be factorized by F07MDF (DSYTRF).

9.1 Program Text

```
    Program f07mhfe

!     F07MHF Example Program Text

!     Mark 24 Release. NAG Copyright 2012.

!     .. Use Statements ..
      Use nag_library, Only: dsyrfs, dsytrf, dsytrs, nag_wp, x04caf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                          :: i, ifail, info, lda, ldaf, ldb, ldx, &
                                          lwork, n, nrhs
      Character (1)                    :: uplo
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:,:), af(:,:), b(:,:), berr(:),    &
                                          ferr(:), work(:), x(:,:)
      Integer, Allocatable             :: ipiv(:), iwork(:)
!     .. Executable Statements ..
      Write (nout,*) 'F07MHF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n, nrhs
      lda = n
      ldaf = n
      ldb = n
      ldx = n
      lwork = 64*n
      Allocate (a(lda,n),af(ldaf,n),b(ldb,nrhs),berr(nrhs),ferr(nrhs), &
        work(lwork),x(ldx,n),ipiv(n),iwork(n))

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

      Read (nin,*) uplo
      If (uplo=='U') Then
        Do i = 1, n
          Read (nin,*) a(i,i:n)
          af(i,i:n) = a(i,i:n)
        End Do
      Else If (uplo=='L') Then
        Do i = 1, n
          Read (nin,*) a(i,1:i)
          af(i,1:i) = a(i,1:i)
        End Do
      End If
      Read (nin,*)(b(i,1:nrhs),i=1,n)

      x(1:n,1:nrhs) = b(1:n,1:nrhs)
```

```
!     Factorize A in the array AF
!     The NAG name equivalent of dsytrf is f07mdf
      Call dsytrf(uplo,n,af,ldaf,ipiv,work,lwork,info)

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

!       Compute solution in the array X
!       The NAG name equivalent of dsytrs is f07mef
        Call dsytrs(uplo,n,nrhs,af,ldaf,ipiv,x,ldx,info)

!       Improve solution, and compute backward errors and
!       estimated bounds on the forward errors
!       The NAG name equivalent of dsyrfs is f07mhf
        Call dsyrfs(uplo,n,nrhs,a,lda,af,ldaf,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 f07mhfe
```

9.2 Program Data

```
F07MHF 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
```

9.3 Program Results

```
 F07MHF 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
```