NAG Toolbox for MATLAB

f07fh

1 Purpose

f07fh returns error bounds for the solution of a real symmetric positive-definite 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 Syntax

```
[x, ferr, berr, info] = f07fh(uplo, a, af, b, x, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

f07fh 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 $AX = B$. The function handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of f07fh in terms of a single right-hand side b and solution x.

Given a computed solution x, the function 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

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

Then the function 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

5.1 Compulsory Input Parameters

1: **uplo – string**

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: **a**(**lda,**$*$) – **double array**

The first dimension of the array **a** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The n by n original symmetric positive-definite matrix A as supplied to f07fd.

3: **af**(**ldaf,**$*$) – **double array**

The first dimension of the array **af** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{n})$

The Cholesky factor of A, as returned by f07fd.

4: **b**(**ldb,**$*$) – **double array**

The first dimension of the array **b** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{nrhs\_p})$

The n by r right-hand side matrix B.

5: **x**(**ldx,**$*$) – **double array**

The first dimension of the array **x** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{nrhs\_p})$

The n by r solution matrix X, as returned by f07fe.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The second dimension of the array **a** The second dimension of the array **af**.

n, the order of the matrix A.

Constraint: $\mathbf{n} \geq 0$.

2: **nrhs_p – int32 scalar**

Default: The second dimension of the array **b**.

r, the number of right-hand sides.

Constraint: $\mathbf{nrhs\_p} \geq 0$.

5.3 Input Parameters Omitted from the MATLAB Interface

lda, ldaf, ldb, ldx, work, iwork

5.4 Output Parameters

1: **x**(**ldx,**$*$) – **double array**

The first dimension of the array **x** must be at least $\max(1, \mathbf{n})$

The second dimension of the array must be at least $\max(1, \mathbf{nrhs\_p})$

The improved solution matrix X.

2: **ferr**($*$) – **double array**

Note: the dimension of the array **ferr** must be at least $\max(1, \mathbf{nrhs\_p})$.

ferr(j) contains an estimated error bound for the jth solution vector, that is, the jth column of X, for $j = 1, 2, \ldots, r$.

3: **berr**$(*)$ **– double array**

Note: the dimension of the array **berr** must be at least $\max(1, \mathbf{nrhs\_p})$.

berr(j) contains the component-wise backward error bound β for the jth solution vector, that is, the jth column of X, for $j = 1, 2, \ldots, r$.

4: **info – int32 scalar**

info $= 0$ unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

info $= -i$

If **info** $= -i$, parameter i had an illegal value on entry. The parameters are numbered as follows:

1: **uplo**, 2: **n**, 3: **nrhs_p**, 4: **a**, 5: **lda**, 6: **af**, 7: **ldaf**, 8: **b**, 9: **ldb**, 10: **x**, 11: **ldx**, 12: **ferr**, 13: **berr**, 14: **work**, 15: **iwork**, 16: **info**.

It is possible that **info** refers to a parameter that is omitted from the MATLAB interface. This usually indicates that an error in one of the other input parameters has caused an incorrect value to be inferred.

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 analogue of this function is f07fv.

9 Example

```
uplo = 'L';
a = [4.16, 0, 0, 0;
     -3.12, 5.03, 0, 0;
     0.5600000000000001, -0.83, 0.76, 0;
     -0.1, 1.18, 0.34, 1.18];
b = [8.699999999999999, 8.300000000000001;
     -13.35, 2.13;
     1.89, 1.61;
     -4.14, 5];
[af, info] = f07fd(uplo, a);
[x, info]  = f07fe(uplo, af, b);
[xOut, ferr, berr, info] = f07fh(uplo, a, af, b, x)
```

```
xOut =
    1.0000    4.0000
   -1.0000    3.0000
    2.0000    2.0000
```

```
   -3.0000    1.0000
ferr =
   1.0e-13 *
    0.2334
    0.2263
berr =
   1.0e-15 *
    0.1021
    0.0500
info =
           0
```