

NAG Toolbox for MATLAB

f07nv

1 Purpose

f07nv returns error bounds for the solution of a complex symmetric 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] = f07nv(uplo, a, af, ipiv, b, x, 'n', n, 'nrhs_p',
nrhs_p)
```

3 Description

f07nv returns the backward errors and estimated bounds on the forward errors for the solution of a complex symmetric 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 f07nv 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

$$(\delta a_{ij}) \leq \beta |a_{ij}| \quad \text{and} \quad (\delta b_i) \leq \beta |b_i|.$$

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 $PUDU^T P^T$, where U is upper triangular.

uplo = 'L'

The lower triangular part of A is stored and A is factorized as $PLDL^T P^T$, where L is lower triangular.

Constraint: **uplo** = 'U' or 'L'.

2: **a(lda,*) – complex 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 matrix A as supplied to f07nr.

3: **af(ldaf,*) – complex 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 $\max(1, \mathbf{n})$

Details of the factorization of A , as returned by f07nr.

4: **ipiv(*) – int32 array**

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

Details of the interchanges and the block structure of D , as returned by f07nr.

5: **b(ldb,*) – complex 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 .

6: **x(ldx,*) – complex 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 f07ns.

5.2 Optional Input Parameters

1: **n – int32 scalar**

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

n , the order of the matrix A .

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

2: **nrhs_p – int32 scalar**

Default: The second dimension of the arrays **b**, **x**. (An error is raised if these dimensions are not equal.)

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, rwork

5.4 Output Parameters

1: **x(ldx,*) – complex 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 $\max(1, \text{nrhs_p})$.

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

3: **berr**(*) – double array

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

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

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: **ipiv**, 9: **b**, 10: **ldb**, 11: **x**, 12: **ldx**, 13: **ferr**, 14: **berr**, 15: **work**, 16: **rwork**, 17: **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 $16n^2$ real floating-point operations. Each step of iterative refinement involves an additional $24n^2$ real 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 5 and never more than 11. Each solution involves approximately $8n^2$ real operations.

The real analogue of this function is f07mh.

9 Example

```
uplo = 'L';
a = [complex(-0.39, -0.71), complex(0, 0), complex(0, 0), complex(0, 0);
      complex(5.14, -0.64), complex(8.86, 1.81), complex(0, +0),
      complex(0, 0);
      complex(-7.86, -2.96), complex(-3.52, +0.58), complex(-2.83, -0.03),
      complex(0, 0);
      complex(3.8, +0.92), complex(5.32, -1.59), complex(-1.54, -2.86),
```

```
complex(-0.56, +0.12)];  
b = [complex(-55.64, +41.22), complex(-19.09, -35.97);  
     complex(-48.18, +66), complex(-12.08, -27.02);  
     complex(-0.49, -1.47), complex(6.95, +20.49);  
     complex(-6.43, +19.24), complex(-4.59, -35.53)];  
[af, ipiv, info] = f07nr(uplo, a);  
[x, info] = f07ns(uplo, a, ipiv, b);  
[xOut, ferr, berr, info] = f07nv(uplo, a, af, ipiv, b, x)
```

```
xOut =  
    1.0000 - 1.0000i   -2.0000 - 1.0000i  
   -2.0000 + 5.0000i    1.0000 - 3.0000i  
    3.0000 - 2.0000i    3.0000 + 2.0000i  
   -4.0000 + 3.0000i   -1.0000 + 1.0000i  
ferr =  
    1.0e-13 *  
    0.1144  
    0.1241  
berr =  
    1.0e-16 *  
    0.4028  
    0.7349  
info =  
      0
```