NAG Toolbox for MATLAB

f07hv

1 Purpose

f07hv returns error bounds for the solution of a complex Hermitian 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 Syntax

```
[x, ferr, berr, info] = f07hv(uplo, kd, ab, afb, b, x, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

f07hv returns the backward errors and estimated bounds on the forward errors for the solution of a complex Hermitian positive-definite band 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 f07hv 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 $U^{\mathrm{H}}U$, where U is upper triangular.

uplo = 'L'

The lower triangular part of A is stored and A is factorized as LL^{H}, where L is lower triangular.

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

2: **kd – int32 scalar**

k_d, the number of superdiagonals or subdiagonals of the matrix A.

Constraint: **kd** ≥ 0.

3: **ab**(**ldab,**$*$) **– complex array**

The first dimension of the array **ab** must be at least **kd** $+ 1$

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

The n by n original Hermitian positive-definite band matrix A as supplied to f07hr.

4: **afb**(**ldafb,**$*$) **– complex array**

The first dimension of the array **afb** must be at least **kd** $+ 1$

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

The Cholesky factor of A, as returned by f07hr.

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 f07hs.

5.2 Optional Input Parameters

1: **n – int32 scalar**

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

n, the order of the matrix A.

Constraint: **n** ≥ 0.

2: **nrhs_p – int32 scalar**

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

r, the number of right-hand sides.

Constraint: **nrhs_p** ≥ 0.

5.3 Input Parameters Omitted from the MATLAB Interface

ldab, ldafb, 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 $\max(1, \mathbf{nrhs\_p})$.

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: **kd**, 4: **nrhs_p**, 5: **ab**, 6: **ldab**, 7: **afb**, 8: **ldafb**, 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 $32nk$ real floating-point operations. Each step of iterative refinement involves an additional $48nk$ real 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 5 and never more than 11. Each solution involves approximately $16nk$ real operations.

The real analogue of this function is f07hh.

9 Example

```
uplo = 'L';
kd = int32(1);
ab = [complex(9.390000000000001, +0), complex(1.69, +0), complex(2.65,
+0), complex(2.17, +0);
      complex(1.08, +1.73), complex(-0.04, -0.29), complex(-0.33, -2.24),
complex(0, +0)];
```

```
afb  =  [complex(3.064310689208912,  +0),  complex(1.116713953189507,  +0),
complex(1.606635558731136, +0), complex(0.4289150674026451, +0);
           complex(0.3524446799090123,  +0.5645641631875845),  complex(-
0.03581937870996763, ...
               -0.2596904956472653),    complex(-0.2053981677466558,    -
1.394217865916694), complex(0, +0)];
b = [complex(-12.42, +68.42), complex(54.3, -56.56);
     complex(-9.93, +0.88), complex(18.32, +4.76);
     complex(-27.3, -0.01), complex(-4.4, +9.970000000000001);
     complex(5.31, +23.63), complex(9.43, +1.41)];
x  =  [complex(-0.9999999999999991,  +8.000000000000002),  complex(5,  -
6.000000000000002);
                       complex(2.000000000000001,    -3.000000000000006),
complex(1.999999999999998, +3.000000000000004);
            complex(-3.999999999999982,  -5.000000000000001),  complex(-
8.000000000000009, +4.000000000000001);
             complex(7.000000000000004,  +6.000000000000018),  complex(-
1.000000000000003, -7.000000000000009)];
[xOut, ferr, berr, info] = f07hv(uplo, kd, ab, afb, b, x)
```

```
xOut =
  -1.0000 + 8.0000i   5.0000 - 6.0000i
   2.0000 - 3.0000i   2.0000 + 3.0000i
  -4.0000 - 5.0000i  -8.0000 + 4.0000i
   7.0000 + 6.0000i  -1.0000 - 7.0000i
ferr =
   1.0e-13 *
    0.3590
    0.3267
berr =
   1.0e-15 *
    0.1030
    0.0827
info =
          0
```