NAG Toolbox for MATLAB

f07gv

1 Purpose

f07gv returns error bounds for the solution of a complex Hermitian 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 Syntax

```
[x, ferr, berr, info] = f07gv(uplo, ap, afp, b, x, 'n', n, 'nrhs_p',
nrhs_p)
```

3 Description

f07gv returns the backward errors and estimated bounds on the forward errors for the solution of a complex Hermitian positive-definite system of linear equations with multiple right-hand sides $AX = B$, using packed storage. The function handles each right-hand side vector (stored as a column of the matrix B) independently, so we describe the function of f07gv 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

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

Then the function 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

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: **ap**($*$) – **complex array**

Note: the dimension of the array **ap** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$.

The n by n original Hermitian positive-definite matrix A as supplied to f07gr.

3: **afp**($*$) – **complex array**

Note: the dimension of the array **afp** must be at least $\max(1, \mathbf{n} \times (\mathbf{n}+1)/2)$.

The Cholesky factor of A stored in packed form, as returned by f07gr.

4: **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.

5: **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 f07gs.

5.2 Optional Input Parameters

1: **n** – **int32 scalar**

Default: The first dimension of the array **ap** and the second dimension of the array **ap**. (An error is raised if these dimensions are not equal.)

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

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: **nrhs_p**, 4: **ap**, 5: **afp**, 6: **b**, 7: **ldb**, 8: **x**, 9: **ldx**, 10: **ferr**, 11: **berr**, 12: **work**, 13: **rwork**, 14: **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 f07gh.

9 Example

```
uplo = 'L';
ap = [complex(3.23, +0);
     complex(1.51, +1.92);
     complex(1.9, -0.84);
     complex(0.42, -2.5);
     complex(3.58, +0);
     complex(-0.23, -1.11);
     complex(-1.18, -1.37);
     complex(4.09, +0);
     complex(2.33, +0.14);
     complex(4.29, +0)];
afp = [complex(1.797220075561143, +0);
     complex(0.8401864749527325, +1.068316577423342);
     complex(1.057188279741849, -0.4673885026227l2);
     complex(0.233694251311356, -1.391037210186643);
     complex(1.316353439509685, +0);
     complex(-0.4701749470106329, +0.3130658155999466);
     complex(0.08335250923944192, +0.03676071443037458);
     complex(1.560392977137124, +0);
     complex(0.9359617337923402, +0.9899692192815736);
     complex(0.6603332973655893, +0)];
b = [complex(3.93, -6.14), complex(1.48, +6.58);
     complex(6.17, +9.42), complex(4.65, -4.75);
     complex(-7.17, -21.83), complex(-4.91, +2.29);
     complex(1.99, -14.38), complex(7.64, -10.79)];
x    =    [complex(0.9999999999999895,    -1.000000000000006),    complex(-
0.9999999999999902, +2.00000000000001);
                   complex(6.020812984559171e-17,    +3.000000000000004),
complex(3.000000000000001, -4.000000000000004);
            complex(-3.999999999999996,   -4.999999999999995),   complex(-
2.000000000000004, +2.999999999999992);
         complex(2,  +0.9999999999999915),  complex(3.999999999999996,  -
4.999999999999991)];
[xOut, ferr, berr, info] = f07gv(uplo, ap, afp, b, x)
```

```
xOut =
   1.0000 - 1.0000i  -1.0000 + 2.0000i
  -0.0000 + 3.0000i   3.0000 - 4.0000i
  -4.0000 - 5.0000i  -2.0000 + 3.0000i
   2.0000 + 1.0000i   4.0000 - 5.0000i
ferr =
   1.0e-13 *
    0.6040
    0.7158
berr =
   1.0e-15 *
    0.1128
    0.0526
info =
           0
```