

NAG Toolbox for MATLAB

f07gp

1 Purpose

f07gp uses the Cholesky factorization

$$A = U^H U \quad \text{or} \quad A = LL^H$$

to compute the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n Hermitian positive-definite matrix stored in packed format and X and B are n by r matrices. Error bounds on the solution and a condition estimate are also provided.

2 Syntax

```
[ap, afp, equed, s, b, x, rcond, ferr, berr, info] = f07gp(fact, uplo,
ap, afp, equed, s, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

f07gp performs the following steps:

1. If **fact** = 'E', real diagonal scaling factors, D_S , are computed to equilibrate the system:

$$(D_S A D_S)(D_S^{-1} X) = D_S B.$$

Whether or not the system will be equilibrated depends on the scaling of the matrix A , but if equilibration is used, A is overwritten by $D_S A D_S$ and B by $D_S B$.

2. If **fact** = 'N' or 'E', the Cholesky decomposition is used to factor the matrix A (after equilibration if **fact** = 'E') as $A = U^H U$ if **uplo** = 'U' or $A = LL^H$ if **uplo** = 'L', where U is an upper triangular matrix and L is a lower triangular matrix.
3. If the leading i by i principal minor is not positive-definite, then the function returns with **info** = i . Otherwise, the factored form of A is used to estimate the condition number of the matrix A . If the reciprocal of the condition number is less than *machine precision*, **info** $\geq N + 1$ is returned as a warning, but the function still goes on to solve for X and compute error bounds as described below.
4. The system of equations is solved for X using the factored form of A .
5. Iterative refinement is applied to improve the computed solution matrix and to calculate error bounds and backward error estimates for it.
6. If equilibration was used, the matrix X is premultiplied by D_S so that it solves the original system before equilibration.

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D 1999 *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia URL: <http://www.netlib.org/lapack/lug>

Golub G H and Van Loan C F 1996 *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

Higham N J 2002 *Accuracy and Stability of Numerical Algorithms* (2nd Edition) SIAM, Philadelphia

5 Parameters

5.1 Compulsory Input Parameters

1: **fact** – string

Specifies whether or not the factorized form of the matrix A is supplied on entry, and if not, whether the matrix A should be equilibrated before it is factorized.

fact = 'F'

afp contains the factorized form of A . If **equed** = 'Y', the matrix A has been equilibrated with scaling factors given by **s**. **afp** will not be modified.

fact = 'N'

The matrix A will be copied to **afp** and factorized.

fact = 'E'

The matrix A will be equilibrated if necessary, then copied to **afp** and factorized.

Constraint: **fact** = 'F', 'N' or 'E'.

2: **uplo** – string

If **uplo** = 'U', the upper triangle of A is stored.

If **uplo** = 'L', the lower triangle of A is stored.

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

3: **ap**(*) – complex array

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

The n by n Hermitian matrix A , packed by columns, except if **fact** = 'F' and **equed** = 'Y', **ap** must contain the equilibrated matrix $D_S A D_S$.

More precisely,

if **uplo** = 'U', the upper triangle of A must be stored with element A_{ij} in **ap**($i + j(j - 1)/2$) for $i \leq j$;

if **uplo** = 'L', the lower triangle of A must be stored with element A_{ij} in **ap**($i + (2n - j)(j - 1)/2$) for $i \geq j$.

4: **afp**(*) – complex array

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

If **fact** = 'F', **afp** contains the triangular factor U or L from the Cholesky factorization $A = U^H U$ or $A = L L^H$, in the same storage format as **ap**. If **equed** $\neq$ 'N', **afp** is the factorized form of the equilibrated matrix $D_S A D_S$.

5: **equed** – string

If **fact** = 'N' or 'E', **equed** need not be set.

If **fact** = 'F', **equed** must specify the form of the equilibration that was performed as follows:

if **equed** = 'N', no equilibration;

if **equed** = 'Y', equilibration was performed, i.e., A has been replaced by $D_S A D_S$.

Constraint: if **fact** = 'F', **equed** = 'N' or 'Y'.

6: **s**(*) – double array

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

If **fact** = 'N' or 'E', **s** need not be set.

If **fact** = 'F' and **equed** = 'Y', **s** must contain the scale factors, D_S , for A ; each element of **s** must be positive.

7: **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 .

5.2 Optional Input Parameters

1: **n** – int32 scalar

Default: The dimension of the array **s**.

n , the number of linear equations, i.e., 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, i.e., the number of columns of the matrix B .

Constraint: **nrhs_p** ≥ 0 .

5.3 Input Parameters Omitted from the MATLAB Interface

ldb, ldx, work, rwork

5.4 Output Parameters

1: **ap(*)** – complex array

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

If **fact** = 'F' or 'N', or if **fact** = 'E' and **equed** = 'N', **ap** is not modified.

If **fact** = 'E' and **equed** = 'Y', **ap** contains $D_S A D_S$.

2: **afp(*)** – complex array

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

If **fact** = 'N', **afp** returns the triangular factor U or L from the Cholesky factorization $A = U^H U$ or $A = L L^H$ of the original matrix A .

If **fact** = 'E', **afp** returns the triangular factor U or L from the Cholesky factorization $A = U^H U$ or $A = L L^H$ of the equilibrated matrix A (see the description of **ap** for the form of the equilibrated matrix).

3: **equed** – string

If **fact** = 'F', **equed** is unchanged from entry.

Otherwise, if **info** ≥ 0 , **equed** specifies the form of the equilibration that was performed as specified above.

4: **s(*)** – double array

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

If **fact** = 'F', **s** is unchanged from entry.

Otherwise, if **info** ≥ 0 and **equed** = 'Y', **s** contains the scale factors, D_S , for A ; each element of **s** is positive.

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 $\max(1, \mathbf{nrhs_p})$

If **equed** = 'N', **b** is not modified.

If **equed** = 'Y', **b** contains $D_S B$.

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 $\max(1, \mathbf{nrhs_p})$

If **info** = 0 or **info** $\geq N + 1$, the n by r solution matrix X to the original system of equations. Note that the arrays A and B are modified on exit if **equed** = 'Y', and the solution to the equilibrated system is $D_S^{-1} X$.

7: **rcond** – double scalar

If **info** ≥ 0 , an estimate of the reciprocal condition number of the matrix A (after equilibration if that is performed), computed as $\mathbf{rcond} = 1 / (\|A\|_1 \|A^{-1}\|_1)$.

8: **ferr(*)** – double array

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

If **info** = 0 or **info** $\geq N + 1$, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \mathbf{ferr}(j)$ where $\hat{x}_j$ is the j th column of the computed solution returned in the array **x** and x_j is the corresponding column of the exact solution X . The estimate is as reliable as the estimate for **rcond**, and is almost always a slight overestimate of the true error.

9: **berr(*)** – double array

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

If **info** = 0 or **info** $\geq N + 1$, an estimate of the component-wise relative backward error of each computed solution vector $\hat{x}_j$ (i.e., the smallest relative change in any element of A or B that makes $\hat{x}_j$ an exact solution).

10: **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: **fact**, 2: **uplo**, 3: **n**, 4: **nrhs_p**, 5: **ap**, 6: **afp**, 7: **equed**, 8: **s**, 9: **b**, 10: **ldb**, 11: **x**, 12: **ldx**, 13: **rcond**, 14: **ferr**, 15: **berr**, 16: **work**, 17: **rwork**, 18: **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.

info > 0 and **info** ≤ *N*

If **info** = *i* and *i* ≤ **n**, the leading minor of order *i* of *A* is not positive-definite, so the factorization could not be completed, and the solution has not been computed. **rcond** = 0 is returned.

info = *N* + 1

U is nonsingular, but **rcond** is less than *machine precision*, meaning that the matrix is singular to working precision. Nevertheless, the solution and error bounds are computed because there are a number of situations where the computed solution can be more accurate than the value of **rcond** would suggest.

7 Accuracy

For each right-hand side vector *b*, the computed solution *x* is the exact solution of a perturbed system of equations $(A + E)x = b$, where

$$|E| \leq c(n)\epsilon|U^T||U|,$$

$c(n)$ is a modest linear function of *n*, and ϵ is the *machine precision*. See Section 10.1 of Higham 2002 for further details.

If $\hat{x}$ is the true solution, then the computed solution *x* satisfies a forward error bound of the form

$$\frac{\|x - \hat{x}\|_\infty}{\|\hat{x}\|_\infty} \leq w_c \text{cond}(A, \hat{x}, b),$$

where $\text{cond}(A, \hat{x}, b) = \frac{\| |A|^{-1}(|A||\hat{x}| + |b|) \|_\infty}{\|\hat{x}\|_\infty} \leq \text{cond}(A) = \frac{\| |A|^{-1} |A| \|_\infty}{1} \leq \kappa_\infty(A)$. If $\hat{x}$ is the *j*th column of *X*, then w_c is returned in **berr**(*j*) and a bound on $\|x - \hat{x}\|_\infty / \|\hat{x}\|_\infty$ is returned in **ferr**(*j*). See Section 4.4 of Anderson *et al.* 1999 for further details.

8 Further Comments

The factorization of *A* requires approximately $\frac{4}{3}n^3$ floating-point operations.

For each right-hand side, computation of the backward error involves a minimum of $16n^2$ floating-point operations. Each step of iterative refinement involves an additional $24n^2$ 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 equations of the form $Ax = b$; the number is usually 4 or 5 and never more than 11. Each solution involves approximately $8n^2$ operations.

The real analogue of this function is f07gb.

9 Example

```
fact = 'Equilibration';
uplo = 'U';
ap = [complex(3.23, +0);
      complex(1.51, -1.92);
      complex(3.58, +0);
      complex(1.9, +0.84);
      complex(-0.23, +1.11);
      complex(4.09, +0);
      complex(0.42, +2.5);
      complex(-1.18, +1.37);
      complex(2.33, -0.14);
      complex(4.29, +0)];
afp = complex(zeros(10,1));
equet = ' ';
s = zeros(4,1);
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)];
[apOut, afpOut, equedOut, sOut, bOut, x, rcond, ferr, berr, info] = ...
    f07gp(fact, uplo, ap, afp, equed, s, b)

```

```

apOut =
    3.2300
    1.5100 - 1.9200i
    3.5800
    1.9000 + 0.8400i
   -0.2300 + 1.1100i
    4.0900
    0.4200 + 2.5000i
   -1.1800 + 1.3700i
    2.3300 - 0.1400i
    4.2900
afpOut =
    1.7972
    0.8402 - 1.0683i
    1.3164
    1.0572 + 0.4674i
   -0.4702 - 0.3131i
    1.5604
    0.2337 + 1.3910i
    0.0834 - 0.0368i
    0.9360 - 0.9900i
    0.6603
equedOut =
N
sOut =
    0.5564
    0.5285
    0.4945
    0.4828
bOut =
    3.9300 - 6.1400i    1.4800 + 6.5800i
    6.1700 + 9.4200i    4.6500 - 4.7500i
   -7.1700 -21.8300i   -4.9100 + 2.2900i
    1.9900 -14.3800i    7.6400 -10.7900i
x =
    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
rcond =
    0.0066
ferr =
    1.0e-13 *
    0.6129
    0.7626
berr =
    1.0e-15 *
    0.0965
    0.1053
info =
        0

```