

NAG Toolbox for MATLAB

f07np

1 Purpose

f07np uses the diagonal pivoting factorization to compute the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n symmetric matrix and X and B are n by r matrices. Error bounds on the solution and a condition estimate are also provided.

2 Syntax

```
[af, ipiv, x, rcond, ferr, berr, info] = f07np(fact, uplo, a, af, ipiv,
b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

f07np performs the following steps:

1. If **fact** = 'N', the diagonal pivoting method is used to factor A . The form of the factorization is $A = UDU^T$ if **uplo** = 'U' or $A = LDL^T$ if **uplo** = 'L', where U (or L) is a product of permutation and unit upper (lower) triangular matrices, and D is symmetric and block diagonal with 1 by 1 and 2 by 2 diagonal blocks.
2. If some $d_{ii} = 0$, so that D is exactly singular, 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.
3. The system of equations is solved for X using the factored form of A .
4. Iterative refinement is applied to improve the computed solution matrix and to calculate error bounds and backward error estimates for it.

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 has been supplied.

fact = 'F'

af and **ipiv** contain the factorized form of the matrix A . **a**, **af** and **ipiv** will not be modified.

fact = 'N'

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

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

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: **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 symmetric matrix A .

If **uplo** = 'U', the upper triangular part of A must be stored and the elements of the array below the diagonal are not referenced.

If **uplo** = 'L', the lower triangular part of A must be stored and the elements of the array above the diagonal are not referenced.

4: **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 $\max(1, \mathbf{n})$

If **fact** = 'F', **af** contains the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $\mathbf{a} = UDU^T$ or $\mathbf{a} = LDL^T$ as computed by f07nr.

5: **ipiv(*)** – int32 array

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

If **fact** = 'F', **ipiv** contains details of the interchanges and the block structure of D , as determined by f07nr.

ipiv(k) > 0

Rows and columns k and **ipiv**(k) were interchanged and $D(k, k)$ is a 1 by 1 diagonal block.

uplo = 'U' and **ipiv**(k) = **ipiv**($k - 1$) < 0

Rows and columns $k - 1$ and $-\mathbf{ipiv}(k)$ were interchanged and $D(k - 1 : k, k - 1 : k)$ is a 2 by 2 diagonal block.

uplo = 'L' and **ipiv**(k) = **ipiv**($k + 1$) < 0

Rows and columns $k + 1$ and $-\mathbf{ipiv}(k)$ were interchanged and $D(k : k + 1, k : k + 1)$ is a 2 by 2 diagonal block.

6: **b(lb,*)** – 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 second dimension of the array **a** The second dimension of the array **af** The dimension of the array **ipiv**.

n , the number of linear equations, i.e., the order of the matrix A .

Constraint: $n \geq 0$.

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

lda, ldaf, ldb, ldx, work, lwork, rwork

5.4 Output Parameters

1: **af(ldaf,*)** – complex array

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

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

If **fact** = 'N', **af** returns the block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $a = UDU^T$ or $a = LDL^T$.

2: **ipiv(*)** – int32 array

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

If **fact** = 'N', **ipiv** contains details of the interchanges and the block structure of D , as determined by f07nr.

3: **x(ldx,*)** – complex array

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

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

If **info** = 0 or **info** $\geq N + 1$, the n by r solution matrix X .

4: **rcond** – double scalar

The estimate of the reciprocal condition number of the matrix A . If **rcond** = 0, the matrix may be exactly singular. This condition is indicated by a return code of **info** > 0 $\leq N$. Otherwise, if **rcond** is less than the *machine precision*, the matrix is singular to working precision. This condition is indicated by a return code of **info** $\geq N + 1$.

5: **ferr(*)** – double array

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

If **info** = 0 or **info** $\geq N + 1$, an estimate of the forward error bound for each computed solution vector, such that $\|\hat{x}_j - x_j\|_\infty / \|x_j\|_\infty \leq \text{ferr}(j)$ 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.

6: **berr**(*) – double array

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

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).

7: **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: **a**, 6: **lda**, 7: **af**, 8: **ldaf**, 9: **ipiv**, 10: **b**, 11: **ldb**, 12: **x**, 13: **ldx**, 14: **rcond**, 15: **ferr**, 16: **berr**, 17: **work**, 18: **lwork**, 19: **rwork**, 20: **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** $\leq N$

If **info** $\leq n$, $d(i, i)$ is exactly zero. The factorization has been completed, but the factor D is exactly singular, so the solution and error bounds could not be computed. **rcond** = 0 is returned.

info = $N + 1$

D 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 $\hat{x}$ is the exact solution of a perturbed system of equations $(A + E)\hat{x} = b$, where

$$\|E\|_1 = O(\epsilon)\|A\|_1,$$

where ϵ is the *machine precision*. See Chapter 11 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 $\text{cond}(A, \hat{x}, b) = \frac{\| |A|^{-1}(|A|\|\hat{x}\| + |b|) \|_\infty}{\|\hat{x}\|_\infty} \leq \text{cond}(A) = \| |A|^{-1} |A| \|_\infty \leq \kappa_\infty(A)$. 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 f07mb.

9 Example

```
fact = 'Not factored';
uplo = 'Upper';
a = [complex(-0.56, +0.12), complex(-1.54, -2.86), complex(5.32, -1.59),
      complex(3.8, +0.92);
      complex(0, 0), complex(-2.83, -0.03), complex(-3.52, +0.58),
      complex(-7.86, -2.96);
      complex(0, +0), complex(0, +0), complex(8.86, +1.81), complex(5.14,
      -0.64);
      complex(0, +0), complex(0, 0), complex(0, +0), complex(-0.39, -
      0.71)];
af = complex(zeros(4, 4));
ipiv = zeros(4, 1, 'int32');
b = [complex(-6.43, +19.24), complex(-4.59, -35.53);
      complex(-0.49, -1.47), complex(6.95, +20.49);
      complex(-48.18, +66), complex(-12.08, -27.02);
      complex(-55.64, +41.22), complex(-19.09, -35.97)];
[afOut, ipivOut, x, rcond, ferr, berr, info] = f07np(fact, uplo, a, af,
ipiv, b)
```

```
afOut =
-2.0954 - 2.2011i -0.1071 - 0.3157i -0.4823 + 0.0150i 0.4426 +
0.1936i
0 4.4079 + 5.3991i -0.6078 + 0.2811i 0.5279 -
0.3715i
0 0 -2.8300 - 0.0300i -7.8600 -
2.9600i
0 0 0 -0.3900 -
0.7100i
ipivOut =
1
2
-2
-2
```

```
x =
-4.0000 + 3.0000i -1.0000 + 1.0000i
3.0000 - 2.0000i 3.0000 + 2.0000i
-2.0000 + 5.0000i 1.0000 - 3.0000i
1.0000 - 1.0000i -2.0000 - 1.0000i
```

```
rcond =
0.0486
```

```
ferr =
1.0e-13 *
0.1168
0.1223
```

```
berr =
1.0e-16 *
0.4559
0.5990
```

```
info =
0
```