NAG Toolbox for MATLAB

f07qn

1 Purpose

f07qn computes the solution to a complex system of linear equations

$$AX = B,$$

where A is an n by n symmetric matrix stored in packed format and X and B are n by r matrices.

2 Syntax

```
[ap, ipiv, b, info] = f07qn(uplo, ap, b, 'n', n, 'nrhs_p', nrhs_p)
```

3 Description

f07qn uses the diagonal pivoting method to factor A as $A = UDU^{\mathrm{T}}$ if **uplo** = 'U' or $A = LDL^{\mathrm{T}}$ if **uplo** = 'L', where U (or L) is a product of permutation and unit upper (lower) triangular matrices, D is symmetric and block diagonal with 1 by 1 and 2 by 2 diagonal blocks. The factored form of A is then used to solve the system of equations $AX = B$.

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: **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'.

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 symmetric matrix A, packed by columns.

More precisely,

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

3: **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 $\max(1, \textbf{nrhs\_p})$

Note: To solve the equations $Ax = b$, where b is a single right-hand side, **b** may be supplied as a one-dimensional array with length $\textbf{ldb} = \max(1, \textbf{n})$.

The n by r right-hand side matrix B.

5.2 Optional Input Parameters

1: **n – int32 scalar**

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

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

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: $\textbf{nrhs\_p} \geq 0$.

5.3 Input Parameters Omitted from the MATLAB Interface

ldb

5.4 Output Parameters

1: **ap(*) – complex array**

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

The block diagonal matrix D and the multipliers used to obtain the factor U or L from the factorization $A = UDU^{\mathrm{T}}$ or $A = LDL^{\mathrm{T}}$ as computed by f07qr, stored as a packed triangular matrix in the same storage format as A.

2: **ipiv(*) – int32 array**

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

Details of the interchanges and the block structure of D, as determined by f07qr.

$\textbf{ipiv}(k) > 0$

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

uplo = 'U' and $\textbf{ipiv}(k) = \textbf{ipiv}(k - 1) < 0$

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

uplo = 'L' and $\textbf{ipiv}(k) = \textbf{ipiv}(k + 1) < 0$

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

3: **b(ldb,*) – complex array**

The first dimension of the array **b** must be at least $\max(1, \textbf{n})$

The second dimension of the array must be at least $\max(1, \textbf{nrhs\_p})$

Note: To solve the equations $Ax = b$, where b is a single right-hand side, **b** may be supplied as a one-dimensional array with length $\textbf{ldb} = \max(1, \textbf{n})$.

If $\textbf{info} = 0$, the n by r solution matrix X.

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: **ipiv**, 6: **b**, 7: **ldb**, 8: **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

If **info** $= i$, d_{ii} is exactly zero. The factorization has been completed, but the block diagonal matrix D is exactly singular, so the solution could not be computed.

7 Accuracy

The computed solution for a single right-hand side, $\hat{x}$, satisfies an equation of the form

$$(A + E)\hat{x} = b,$$

where

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

and ϵ is the ***machine precision***. An approximate error bound for the computed solution is given by

$$\frac{\|\hat{x} - x\|_1}{\|x\|_1} \leq \kappa(A)\frac{\|E\|_1}{\|A\|_1},$$

where $\kappa(A) = \left\|A^{-1}\right\|_1 \|A\|_1$, the condition number of A with respect to the solution of the linear equations. See Section 4.4 of Anderson *et al.* 1999 and Chapter 11 of Higham 2002 for further details.

f07qp is a comprehensive LAPACK driver that returns forward and backward error bounds and an estimate of the condition number. Alternatively, f04dj solves $Ax = b$ and returns a forward error bound and condition estimate. f04dj calls f07qn to solve the equations.

8 Further Comments

The total number of floating-point operations is approximately $\frac{4}{3}n^3 + 8n^2r$, where r is the number of right-hand sides.

The real analogue of this function is f07pa.

9 Example

```
uplo = 'U';
ap = [complex(-0.5600000000000001, +0.12);
     complex(-1.54, -2.86);
     complex(-2.83, -0.03);
     complex(5.32, -1.59);
     complex(-3.52, +0.58);
     complex(8.859999999999999, +1.81);
     complex(3.8, +0.92);
     complex(-7.86, -2.96);
```

```
     complex(5.14, -0.64);
     complex(-0.39, -0.71)];
b = [complex(-6.43, +19.24);
     complex(-0.49, -1.47);
     complex(-48.18, +66);
     complex(-55.64, +41.22)];
[apOut, ipiv, bOut, info] = f07qn(uplo, ap, b)
```

```
apOut =
  -2.0954 - 2.2011i
  -0.1071 - 0.3157i
   4.4079 + 5.3991i
  -0.4823 + 0.0150i
  -0.6078 + 0.2811i
  -2.8300 - 0.0300i
   0.4426 + 0.1936i
   0.5279 - 0.3715i
  -7.8600 - 2.9600i
  -0.3900 - 0.7100i
ipiv =
           1
           2
          -2
          -2
bOut =
  -4.0000 + 3.0000i
   3.0000 - 2.0000i
  -2.0000 + 5.0000i
   1.0000 - 1.0000i
info =
           0
```