NAG Toolbox for MATLAB

f01mc

1 Purpose

f01mc computes the Cholesky factorization of a real symmetric positive-definite variable-bandwidth matrix.

2 Syntax

```
[al, d, ifail] = f01mc(a, nrow, 'n', n, 'lal', lal)
```

3 Description

f01mc determines the unit lower triangular matrix L and the diagonal matrix D in the Cholesky factorization $A = LDL^{\mathrm{T}}$ of a symmetric positive-definite variable-bandwidth matrix A of order n. (Such a matrix is sometimes called a 'sky-line' matrix.)

The matrix A is represented by the elements lying within the **envelope** of its lower triangular part, that is, between the first nonzero of each row and the diagonal (see Section 9 for an example). The **width** $\mathbf{nrow}(i)$ of the ith row is the number of elements between the first nonzero element and the element on the diagonal, inclusive. Although, of course, any matrix possesses an envelope as defined, this function is primarily intended for the factorization of symmetric positive-definite matrices with an **average** bandwidth which is small compared with n (also see Section 8).

The method is based on the property that during Cholesky factorization there is no fill-in outside the envelope.

The determination of L and D is normally the first of two steps in the solution of the system of equations $Ax = b$. The remaining step, viz. the solution of $LDL^{\mathrm{T}}x = b$, may be carried out using f04mc.

4 References

Jennings A 1966 A compact storage scheme for the solution of symmetric linear simultaneous equations *Comput. J.* **9** 281–285

Wilkinson J H and Reinsch C 1971 *Handbook for Automatic Computation II, Linear Algebra* Springer–Verlag

5 Parameters

5.1 Compulsory Input Parameters

1: **a(lal) – double array**

The elements within the envelope of the lower triangle of the positive-definite symmetric matrix A, taken in row by row order. The following code assigns the matrix elements within the envelope to the correct elements of the array:

```
k = 0;
for i = 1:n
  for j = i-nrow(i)+1:i
    k = k + 1;
    a(k) = matrix(i,j);
  end
end
```

See also Section 8.

2: **nrow(n) – int32 array**

nrow(i) must contain the width of row i of the matrix A, i.e., the number of elements between the first (leftmost) nonzero element and the element on the diagonal, inclusive.

Constraint: $1 \leq$ **nrow**$(i) \leq i$, for $i = 1, 2, \ldots, n$.

5.2 Optional Input Parameters

1: **n – int32 scalar**

Default: The dimension of the arrays **nrow**, **d**. (An error is raised if these dimensions are not equal.)

n, the order of the matrix A.

Constraint: **n** ≥ 1.

2: **lal – int32 scalar**

Default: The dimension of the arrays **a**, **al**. (An error is raised if these dimensions are not equal.)

Constraint: **lal** $\geq$ **nrow**$(1) +$ **nrow**$(2) + \ldots +$ **nrow**(n).

5.3 Input Parameters Omitted from the MATLAB Interface

None.

5.4 Output Parameters

1: **al(lal) – double array**

The elements within the envelope of the lower triangular matrix L, taken in row by row order. The envelope of L is identical to that of the lower triangle of A. The unit diagonal elements of L are stored explicitly. See also Section 8.

2: **d(n) – double array**

The diagonal elements of the diagonal matrix D. Note that the determinant of A is equal to the product of these diagonal elements. If the value of the determinant is required it should not be determined by forming the product explicitly, because of the possibility of overflow or underflow. The logarithm of the determinant may safely be formed from the sum of the logarithms of the diagonal elements.

3: **ifail – int32 scalar**

0 unless the function detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the function:

ifail $= 1$

On entry, **n** < 1,
or for some i, **nrow**$(i) < 1$ or **nrow**$(i) > i$,
or **lal** $<$ **nrow**$(1) +$ **nrow**$(2) + \ldots +$ **nrow**$($**n**$)$.

ifail $= 2$

A is not positive-definite, or this property has been destroyed by rounding errors. The factorization has not been completed.

ifail = 3

A is not positive-definite, or this property has been destroyed by rounding errors. The factorization has been completed but may be very inaccurate (see Section 7).

7 Accuracy

If **ifail** = 0 on exit, then the **computed** L and D satisfy the relation $LDL^{\mathrm{T}} = A + F$, where

$$\|F\|_2 \leq km^2\epsilon \times \max_i a_{ii}$$

and

$$\|F\|_2 \leq km^2\epsilon \times \|A\|_2,$$

where k is a constant of order unity, m is the largest value of **nrow**(i), and ϵ is the ***machine precision***. See pages 25–27 and 54–55 of Wilkinson and Reinsch 1971. If **ifail** = 3 on exit, then the factorization has been completed although the matrix was not positive-definite. However the factorization may be very inaccurate and should be used only with great caution. For instance, if it is used to solve a set of equations $Ax = b$ using f04mc, the residual vector $b - Ax$ should be checked.

8 Further Comments

The time taken by f01mc is approximately proportional to the sum of squares of the values of **nrow**(i).

The distribution of row widths may be very non-uniform without undue loss of efficiency. Moreover, the function has been designed to be as competitive as possible in speed with functions designed for full or uniformly banded matrices, when applied to such matrices.

9 Example

```
a = [1;
     2;
     5;
     3;
     13;
     16;
     5;
     14;
     18;
     8;
     55;
     24;
     17;
     77];
nrow = [int32(1);
     int32(2);
     int32(2);
     int32(1);
     int32(5);
     int32(3)];
[al, d, ifail] = f01mc(a, nrow)
```

```
al =
    1.0000
    2.0000
    1.0000
    3.0000
    1.0000
    1.0000
    5.0000
    4.0000
    1.5000
```

```
    0.5000
    1.0000
    1.5000
    5.0000
    1.0000
d =
     1
     1
     4
    16
     1
    16
ifail =
           0
```