NAG Library Routine Document

F01MCF

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of ***bold italicised*** terms and other implementation-dependent details.

1 Purpose

F01MCF computes the Cholesky factorization of a real symmetric positive definite variable-bandwidth matrix.

2 Specification

```
SUBROUTINE F01MCF (N, A, LAL, NROW, AL, D, IFAIL)

INTEGER             N, LAL, NROW(N), IFAIL
REAL (KIND=nag_wp) A(LAL), AL(LAL), D(N)
```

3 Description

F01MCF 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 10 for an example). The **width** $\mathrm{NROW}(i)$ 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 routine is primarily intended for the factorization of symmetric positive definite matrices with an **average** bandwidth which is small compared with n (also see Section 9).

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 F04MCF.

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

1: N – INTEGER *Input*

On entry: n, the order of the matrix A.

Constraint: $\mathrm{N} \geq 1$.

2: A(LAL) – REAL (KIND=nag_wp) array *Input*

On entry: 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
      DO 20 I = 1, N
         DO 10 J = I-NROW(I)+1, I
            K = K + 1
            A(K) = matrix (I,J)
10       CONTINUE
20    CONTINUE
```

See also Section 9.

3: LAL – INTEGER *Input*

On entry: the dimension of the arrays A and AL as declared in the (sub)program from which F01MCF is called.

Constraint: $\mathrm{LAL} \geq \mathrm{NROW}(1) + \mathrm{NROW}(2) + \ldots + \mathrm{NROW}(n)$.

4: NROW(N) – INTEGER array *Input*

On entry: $\mathrm{NROW}(i)$ 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: $1 \leq \mathrm{NROW}(i) \leq i$, for $i = 1, 2, \ldots, n$.

5: AL(LAL) – REAL (KIND=nag_wp) array *Output*

On exit: 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 9.

6: D(N) – REAL (KIND=nag_wp) array *Output*

On exit: 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.

7: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, if you are not familiar with this parameter, the recommended value is 0. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

On exit: $\mathrm{IFAIL} = 0$ unless the routine detects an error or a warning has been flagged (see Section 6).

6 Error Indicators and Warnings

If on entry $\mathrm{IFAIL} = 0$ or -1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors or warnings detected by the routine:

$\mathrm{IFAIL} = 1$

On entry, $\mathrm{N} < 1$,
or for some i, $\mathrm{NROW}(i) < 1$ or $\mathrm{NROW}(i) > i$,
or $\mathrm{LAL} < \mathrm{NROW}(1) + \mathrm{NROW}(2) + \ldots + \mathrm{NROW}(\mathrm{N})$.

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 not been completed.

IFAIL $= -99$

An unexpected error has been triggered by this routine. Please contact NAG.

See Section 3.8 in the Essential Introduction for further information.

IFAIL $= -399$

Your licence key may have expired or may not have been installed correctly.

See Section 3.7 in the Essential Introduction for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.6 in the Essential Introduction for further information.

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).

8 Parallelism and Performance

Not applicable.

9 Further Comments

The time taken by F01MCF 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 routine has been designed to be as competitive as possible in speed with routines designed for full or uniformly banded matrices, when applied to such matrices.

Unless otherwise stated in the Users' Note for your implementation, the routine may be called with the same actual array supplied for parameters A and AL, in which case L overwrites the lower triangle of A. However this is not standard Fortran and may not work in all implementations.

10 Example

This example obtains the Cholesky factorization of the symmetric matrix, whose lower triangle is:

$$\begin{pmatrix} 1 & & & & & \\ 2 & 5 & & & & \\ 0 & 3 & 13 & & & \\ 0 & 0 & 0 & 16 & & \\ 5 & 14 & 18 & 8 & 55 & \\ 0 & 0 & 0 & 24 & 17 & 77 \end{pmatrix}.$$

For this matrix, the elements of NROW must be set to 1, 2, 2, 1, 5, 3, and the elements within the envelope must be supplied in row order as:

$$1, 2, 5, 3, 13, 16, 5, 14, 18, 8, 55, 24, 17, 77.$$

10.1 Program Text

```
    Program f01mcfe

!     F01MCF Example Program Text

!     Mark 25 Release. NAG Copyright 2014.

!     .. Use Statements ..
      Use nag_library, Only: f01mcf, nag_wp
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Integer, Parameter              :: nin = 5, nout = 6
!     .. Local Scalars ..
      Integer                         :: i, ifail, k1, k2, lal, n
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable :: a(:), al(:), d(:)
      Integer, Allocatable            :: nrow(:)
!     .. Executable Statements ..
      Write (nout,*) 'F01MCF Example Program Results'
!     Skip heading in data file
      Read (nin,*)
      Read (nin,*) n
      Allocate (d(n),nrow(n))
      Read (nin,*) nrow(1:n)
      lal = 0
      Do i = 1, n
        lal = lal + nrow(i)
      End Do
      Allocate (a(lal),al(lal))
      k2 = 0
      Do i = 1, n
        k1 = k2 + 1
        k2 = k2 + nrow(i)
        Read (nin,*) a(k1:k2)
      End Do

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f01mcf(n,a,lal,nrow,al,d,ifail)

      Write (nout,*)
      Write (nout,*) '  I    D(I)   Row I of unit lower triangle'
      Write (nout,*)
      k2 = 0
      Do i = 1, n
        k1 = k2 + 1
        k2 = k2 + nrow(i)
```

```
        Write (nout,99999) i, d(i), al(k1:k2)
      End Do

99999 Format (1X,I3,7F8.3)
    End Program f01mcfe
```

10.2 Program Data

```
F01MCF Example Program Data
  6                                 : n
  1  2  2  1  5  3                  : nrow
   1.0
   2.0   5.0
   3.0  13.0
  16.0
   5.0  14.0  18.0   8.0  55.0
  24.0  17.0  77.0                  : a
```

10.3 Program Results

```
 F01MCF Example Program Results

   I    D(I)   Row I of unit lower triangle

   1   1.000   1.000
   2   1.000   2.000   1.000
   3   4.000   3.000   1.000
   4  16.000   1.000
   5   1.000   5.000   4.000   1.500   0.500   1.000
   6  16.000   1.500   5.000   1.000
```