NAG Toolbox

nag_matop_real_vband_posdef_fac (f01mc)

1 Purpose

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

2 Syntax

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

3 Description

nag_matop_real_vband_posdef_fac (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 10 for an example). The **width** **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 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 nag_linsys_real_posdef_vband_solve (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**) – REAL (KIND=nag_wp) 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 9.

2: **nrow**(**n**) – INTEGER 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** – INTEGER

Default: the dimension of the array **nrow**.

n, the order of the matrix A.

Constraint: **n** ≥ 1.

2: **lal** – INTEGER

Default: the dimension of the array **a**.

The dimension of the arrays **a** and **al**.

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

5.3 Output Parameters

1: **al**(**lal**) – REAL (KIND=nag_wp) 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 9.

2: **d**(**n**) – REAL (KIND=nag_wp) 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** – INTEGER

ifail $= 0$ unless the function detects an error (see Section 5).

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 $= 3$ (*warning*)

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.

ifail $= -399$

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

ifail $= -999$

Dynamic memory allocation failed.

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 Further Comments

The time taken by nag_matop_real_vband_posdef_fac (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

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:

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

9.1 Program Text

```
function f01mc_example

fprintf('f01mc example results\n\n');

a = [1;
     2;     5;
            3;     13;
```

```
                          16;
     5;    14;     18;      8;     55;
                   24;     17;     77];
nrow = [nag_int(1);  2; 2; 1; 5; 3];

% Factorize
[L, D, ifail] = f01mc(a, nrow);

% Display
fprintf('    D                   L\n');
k = 0;
for i = 1:6
  fprintf(' %6.3f    ',D(i));
  for j = 1:i-nrow(i)
    fprintf('%8s',' ');
  end
  fprintf('%8.3f',L(k+1:k+nrow(i)));
  fprintf('\n');
  k = k + nrow(i);
end
```

9.2 Program Results

```
     f01mc example results

    D                   L
  1.000       1.000
  1.000       2.000   1.000
  4.000               3.000   1.000
 16.000                                1.000
  1.000       5.000   4.000   1.500   0.500   1.000
 16.000                                1.500   5.000   1.000
```