NAG Library Routine Document

F11MEF

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

F11MEF computes the LU factorization of a real sparse matrix in compressed column (Harwell–Boeing), column-permuted format.

2 Specification

```
SUBROUTINE F11MEF (N, IROWIX, A, IPRM, THRESH, NZLMX, NZLUMX, NZUMX, IL,    &
                   LVAL, IU, UVAL, NNZL, NNZU, FLOP, IFAIL)
INTEGER            N, IROWIX(*), IPRM(7*N), NZLMX, NZLUMX, NZUMX,          &
                   IL(7*N+NZLMX+4), IU(2*N+NZUMX+1), NNZL, NNZU, IFAIL
REAL (KIND=nag_wp) A(*), THRESH, LVAL(NZLUMX), UVAL(NZUMX), FLOP
```

3 Description

Given a real sparse matrix A, F11MEF computes an LU factorization of A with partial pivoting, $P_rAP_c = LU$, where P_r is a row permutation matrix (computed by F11MEF), P_c is a (supplied) column permutation matrix, L is unit lower triangular and U is upper triangular. The column permutation matrix, P_c, must be computed by a prior call to F11MDF. The matrix A must be presented in the column permuted, compressed column (Harwell–Boeing) format.

The LU factorization is output in the form of four one-dimensional arrays: integer arrays IL and IU and real-valued arrays LVAL and UVAL. These describe the sparsity pattern and numerical values in the L and U matrices. The minimum required dimensions of these arrays cannot be given as a simple function of the size arguments (order and number of nonzero values) of the matrix A. This is due to unpredictable fill-in created by partial pivoting. F11MEF will, on return, indicate which dimensions of these arrays were not adequate for the computation or (in the case of one of them) give a firm bound. You should then allocate more storage and try again.

4 References

Demmel J W, Eisenstat S C, Gilbert J R, Li X S and Li J W H (1999) A supernodal approach to sparse partial pivoting *SIAM J. Matrix Anal. Appl.* **20** 720–755

Demmel J W, Gilbert J R and Li X S (1999) An asynchronous parallel supernodal algorithm for sparse gaussian elimination *SIAM J. Matrix Anal. Appl.* **20** 915–952

5 Arguments

1: N – INTEGER *Input*

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

Constraint: N ≥ 0.

2: IROWIX($*$) – INTEGER array *Input*

Note: the dimension of the array IROWIX must be at least nnz, the number of nonzeros of the sparse matrix A.

On entry: the row index array of sparse matrix A.

3: A(∗) – REAL (KIND=nag_wp) array *Input*

Note: the dimension of the array A must be at least nnz, the number of nonzeros of the sparse matrix A.

On entry: the array of nonzero values in the sparse matrix A.

4: IPRM($7 \times$ N) – INTEGER array *Input/Output*

On entry: contains the column permutation which defines the permutation P_c and associated data structures as computed by routine F11MDF.

On exit: part of the array is modified to record the row permutation P_r determined by pivoting.

5: THRESH – REAL (KIND=nag_wp) *Input*

On entry: the diagonal pivoting threshold, t. At step j of the Gaussian elimination, if $\left|A_{jj}\right| \geq t\left(\max_{i \geq j}\left|A_{ij}\right|\right)$, use A_{jj} as a pivot, otherwise use $\max_{i \geq j}\left|A_{ij}\right|$. A value of $t = 1$ corresponds to partial pivoting, a value of $t = 0$ corresponds to always choosing the pivot on the diagonal (unless it is zero).

Suggested value: THRESH $= 1.0$. Smaller values may result in a faster factorization, but the benefits are likely to be small in most cases. It might be possible to use THRESH $= 0.0$ if you are confident about the stability of the factorization, for example, if A is diagonally dominant.

Constraint: $0.0 \leq$ THRESH ≤ 1.0.

6: NZLMX – INTEGER *Input*

On entry: indicates the available size of array IL. The dimension of IL should be at least $7 \times$ N + NZLMX + 4. A good range for NZLMX that works for many problems is nnz to $8 \times nnz$, where nnz is the number of nonzeros in the sparse matrix A. If, on exit, IFAIL $= 2$, the given NZLMX was too small and you should attempt to provide more storage and call the routine again.

Constraint: NZLMX ≥ 1.

7: NZLUMX – INTEGER *Input/Output*

On entry: indicates the available size of array LVAL. The dimension of LVAL should be at least NZLUMX.

Constraint: NZLUMX ≥ 1.

On exit: if IFAIL $= 4$, the given NZLUMX was too small and is reset to a value that will be sufficient. You should then provide the indicated storage and call the routine again.

8: NZUMX – INTEGER *Input*

On entry: indicates the available sizes of arrays IU and UVAL. The dimension of IU should be at least $2 \times$ N + NZUMX + 1 and the dimension of UVAL should be at least NZUMX. A good range for NZUMX that works for many problems is nnz to $8 \times nnz$, where nnz is the number of nonzeros in the sparse matrix A. If, on exit, IFAIL $= 3$, the given NZUMX was too small and you should attempt to provide more storage and call the routine again.

Constraint: NZUMX ≥ 1.

9: IL($7 \times$ N + NZLMX + 4) – INTEGER array *Output*

On exit: encapsulates the sparsity pattern of matrix L.

10: LVAL(NZLUMX) – REAL (KIND=nag_wp) array *Output*

On exit: records the nonzero values of matrix L and some of the nonzero values of matrix U.

11: IU($2 \times \mathrm{N} + \mathrm{NZUMX} + 1$) – INTEGER array *Output*

On exit: encapsulates the sparsity pattern of matrix U.

12: UVAL(NZUMX) – REAL (KIND=nag_wp) array *Output*

On exit: records some of the nonzero values of matrix U.

13: NNZL – INTEGER *Output*

On exit: the number of nonzero values in the matrix L.

14: NNZU – INTEGER *Output*

On exit: the number of nonzero values in the matrix U.

15: FLOP – REAL (KIND=nag_wp) *Output*

On exit: the number of floating-point operations performed.

16: IFAIL – INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. If you are unfamiliar with this argument you should refer to Section 3.4 in How to Use the NAG Library and its Documentation 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 argument, 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: 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 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:

IFAIL $= 1$

On entry, N $= \langle value \rangle$.
Constraint: N ≥ 0.

On entry, NZLMX $= \langle value \rangle$.
Constraint: NZLMX ≥ 1.

On entry, NZLUMX $= \langle value \rangle$.
Constraint: NZLUMX ≥ 1.

On entry, NZUMX $= \langle value \rangle$.
Constraint: NZUMX ≥ 1.

On entry, THRESH $= \langle value \rangle$.
Constraint: $0.0 \leq$ THRESH ≤ 1.0.

IFAIL $= 2$

Insufficient NZLMX.

IFAIL $= 3$

Insufficient NZUMX.

IFAIL $= 4$

Insufficient NZLUMX.

IFAIL $= 5$

The matrix is singular – no factorization possible.

IFAIL $= -99$

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

See Section 3.9 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -399$

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

See Section 3.8 in How to Use the NAG Library and its Documentation for further information.

IFAIL $= -999$

Dynamic memory allocation failed.

See Section 3.7 in How to Use the NAG Library and its Documentation for further information.

7 Accuracy

The computed factors L and U are the exact factors of a perturbed matrix $A + E$, where

$$|E| \leq c(n)\epsilon|L||U|,$$

$c(n)$ is a modest linear function of n, and ϵ is the ***machine precision***, when partial pivoting is used. If no partial pivoting is used, the factorization accuracy can be considerably worse. A call to F11MMF after F11MEF can help determine the quality of the factorization.

8 Parallelism and Performance

F11MEF is threaded by NAG for parallel execution in multithreaded implementations of the NAG Library.

F11MEF makes calls to BLAS and/or LAPACK routines, which may be threaded within the vendor library used by this implementation. Consult the documentation for the vendor library for further information.

Please consult the X06 Chapter Introduction for information on how to control and interrogate the OpenMP environment used within this routine. Please also consult the Users' Note for your implementation for any additional implementation-specific information.

9 Further Comments

The total number of floating-point operations depends on the sparsity pattern of the matrix A.

A call to F11MEF may be followed by calls to the routines:

F11MFF to solve $AX = B$ or $A^{\mathrm{T}}X = B$;

F11MGF to estimate the condition number of A;

F11MMF to estimate the reciprocal pivot growth of the LU factorization.

10 Example

This example computes the LU factorization of the matrix A, where

$$A = \begin{pmatrix} 2.00 & 1.00 & 0 & 0 & 0 \\ 0 & 0 & 1.00 & -1.00 & 0 \\ 4.00 & 0 & 1.00 & 0 & 1.00 \\ 0 & 0 & 0 & 1.00 & 2.00 \\ 0 & -2.00 & 0 & 0 & 3.00 \end{pmatrix}.$$

10.1 Program Text

```
    Program f11mefe

!     F11MEF Example Program Text

!     Mark 26 Release. NAG Copyright 2016.

!     .. Use Statements ..
      Use nag_library, Only: f11mdf, f11mef, nag_wp, x04cbf
!     .. Implicit None Statement ..
      Implicit None
!     .. Parameters ..
      Real (Kind=nag_wp), Parameter    :: one = 1.E0_nag_wp
      Integer, Parameter               :: nin = 5, nout = 6
!     .. Local Scalars ..
      Real (Kind=nag_wp)               :: flop, thresh
      Integer                          :: i, ifail, n, nnz, nnzl, nnzu, nzlmx, &
                                          nzlumx, nzumx
      Character (1)                    :: spec
!     .. Local Arrays ..
      Real (Kind=nag_wp), Allocatable  :: a(:), lval(:), uval(:)
      Integer, Allocatable             :: icolzp(:), il(:), iprm(:),           &
                                          irowix(:), iu(:)
      Character (1)                    :: clabs(1), rlabs(1)
!     .. Executable Statements ..
      Write (nout,*) 'F11MEF Example Program Results'
      Flush (nout)
!     Skip heading in data file
      Read (nin,*)

!     Read order of matrix

      Read (nin,*) n

      Allocate (icolzp(n+1),iprm(7*n))

!     Read the matrix A

      Read (nin,*) icolzp(1:n+1)
      nnz = icolzp(n+1) - 1

      Allocate (a(nnz),lval(8*nnz),uval(8*nnz),il(7*n+8*nnz+4),irowix(nnz),    &
        iu(2*n+8*nnz+1))

      Do i = 1, nnz
        Read (nin,*) a(i), irowix(i)
      End Do

!     Calculate COLAMD permutation

      spec = 'M'

!     ifail: behaviour on error exit
!             =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call f11mdf(spec,n,icolzp,irowix,iprm,ifail)

!     Factorise
```

```
      thresh = one
      ifail = 0
      nzlmx = 8*nnz
      nzlumx = 8*nnz
      nzumx = 8*nnz
      Call f11mef(n,irowix,a,iprm,thresh,nzlmx,nzlumx,nzumx,il,lval,iu,uval,   &
        nnzl,nnzu,flop,ifail)

!     Output results

      Write (nout,99999)
      Write (nout,99998) nnzl + nnzu - n
      Flush (nout)

      Call x04cbf('G','X',1,10,lval,1,'F7.2','Factor elements in LVAL','N',    &
        rlabs,'N',clabs,80,0,ifail)
      Call x04cbf('G','X',1,4,uval,1,'F7.2','Factor elements in UVAL','N',     &
        rlabs,'N',clabs,80,0,ifail)

99999 Format (1X,/,1X,'Number of nonzeros in factors (excluding unit',         &
        ' diagonal)')
99998 Format (1X,I8)
    End Program f11mefe
```

10.2 Program Data

```
F11MEF Example Program Data
  5                N
 1
 3
 5
 7
 9
 12   ICOLZP(I) I=1,..,N+1
  2.   1
  4.   3
  1.   1
 -2.   5
  1.   2
  1.   3
 -1.   2
  1.   4
  1.   3
  2.   4
  3.   5    A(I), IROWIX(I) I=1,...,NNZ
```

10.3 Program Results

```
 F11MEF Example Program Results

 Number of nonzeros in factors (excluding unit diagonal)
       14
 Factor elements in LVAL
    -2.00  -0.50   4.00   0.50   2.00   0.50  -1.00   0.50   1.00  -1.00
 Factor elements in UVAL
     1.00   3.00   1.00   1.00
```