1 SUMMARY

This routine forms an **incomplete LU factorization of an $n \times n$ sparse unsymmetric matrix A.** No fill-in is allowed. The entries of **A** are stored by rows. If **A** has zeros on the diagonal, the routine first finds a row permutation **Q** which makes the matrix have nonzeros on the diagonal. The incomplete **LU** factorization of the permuted matrix **QA** is then formed. **L** is lower triangular and **U** is unit upper triangular. The incomplete factorization may be used as a preconditioner when solving the linear system $\mathbf{Ax} = \mathbf{b}$. A second entry performs the preconditioning operations

$\mathbf{y} = \mathbf{Pz}$ and $\mathbf{y} = \mathbf{P}^{\mathrm{T}}\mathbf{z}$,

where $\mathbf{P} = (\mathbf{LU})^{-1}\mathbf{Q}$ is the preconditioner.

ATTRIBUTES — Version: 1.2.0. **Types:** Real (single, double). **Calls:** `FD15`, `MC21`, `MC22`, `MC38`. **Language:** Fortran 77. **Date:** April 1995. **Origin:** N.I.M. Gould and J.A. Scott, Rutherford Appleton Laboratory.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequence

There are three entries:

(a) `MI11I/ID` sets default values for control parameters. It should normally be called once prior to calls to `MI11A/AD`.

(b) `MI11A/AD` first finds a row permutation **Q** which makes the matrix have nonzeros on the diagonal. The ILU(0) factorization of the permuted matrix is then formed.

(c) `MI11B/BD` performs the preconditioning operation $\mathbf{y} = \mathbf{Pz}$ or $\mathbf{y} = \mathbf{P}^{\mathrm{T}}\mathbf{z}$, where $\mathbf{P} = (\mathbf{LU})^{-1}\mathbf{Q}$.

2.1.1 To set default values for the control parameters

The single precision version

```
CALL MI11I(ICNTL,CNTL)
```

The double precision version

```
CALL MI11ID(ICNTL,CNTL)
```

`ICNTL` is an `INTEGER` array of length `4` that need not be set by the user . On return it contains default values (see Section 2.2.1 for details).

`CNTL` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `2` that need not be set by the user . On return it contains default values (see Section 2.2.1 for details).

2.1.2 To form the incomplete factorization

The single precision version

```
CALL MI11A(N,NZ,INDEX,A,IP,JCN,FACTOR,IFPTR,IW,ICNTL,CNTL,INFO)
```

The double precision version

```
CALL MI11AD(N,NZ,INDEX,A,IP,JCN,FACTOR,IFPTR,IW,ICNTL,CNTL,INFO)
```

N is an INTEGER variable that must be set by the user to n, the order of the matrix $\mathbf{A}$. This argument is not altered by the routine. **Restriction**: N $\geq$ 1.

NZ is an INTEGER variable that must be set by the user to the length of the arrays INDEX, A, JCN, and FACTOR. NZ must be at least as large as the number of entries in the matrix $\mathbf{A}$. This argument is not altered by the routine. **Restriction**: NZ $\geq$max(IP(N+1)-1,N).

INDEX is an INTEGER array of length NZ. If ICNTL(4) $=$ 0 (the default), the first IP(N+1)-1 entries must be set by the user to hold the column indices of the entries of the matrix $\mathbf{A}$. The entries must be ordered by rows, with the entries in each row contiguous and those of row I preceding those of row I+1 (I = 1, 2,..., N). The order within each row is arbitrary. If ICNTL(4) $\neq$ 0, the first IP(N+1)-1 entries must be set by the user to hold the row indices of the entries of the matrix $\mathbf{A}$. The entries must be ordered by columns, with the entries in each column contiguous and those of column I preceding those of column I+1 (I = 1, 2,..., N). The order within each column is arbitrary. Out-of-range indices are ignored and the number of such indices is returned in INFO(2) (see Section 2.2.2). This argument is not altered by the routine.

A is a REAL (DOUBLE PRECISION in the D version) array of length NZ that must be set by the user to the values of the entries of the matrix $\mathbf{A}$ corresponding to the indices set in INDEX. If there is more than one entry for a particular position, the values are accumulated. The number of such multiple entries is returned in INFO(3) (see Section 2.2.2). Entries with value zero are ignored and the number of such entries is returned in INFO(4) (see Section 2.2.2). This argument is not altered by the routine.

IP is an INTEGER array of length N+1. If ICNTL(4) $=$ 0, IP(I) must point to the position in the arrays INDEX and FACTOR of the first entry in row I. If ICNTL(4) $\neq$ 0, IP(I) must point to the position in the arrays INDEX and FACTOR of the first entry in column I (I = 1, 2,..., N). IP(N+1)-1 must be set to the number of entries in the matrix $\mathbf{A}$. This argument is not altered by the routine.

JCN is an INTEGER array of length NZ that need not be set by the user . On exit, the leading entries hold the column indices of the matrix $\mathbf{L}+\mathbf{U}$ (excluding the unit diagonal of $\mathbf{U}$) ordered by rows, with the entries in each row ordered by columns.

FACTOR is a REAL (DOUBLE PRECISION in the D version) of length NZ that need not be set by the user . On exit, the leading entries hold the values of the entries of the matrix $\mathbf{L}+\mathbf{U}$ (excluding the unit diagonal of $\mathbf{U}$) corresponding to the column indices set in JCN.

IFPTR is an INTEGER array of length N+1 that need not be set by the user . On exit IFPTR(I) points to the position in the arrays JCN and FACTOR of the first entry in row I of the matrix $\mathbf{L}+\mathbf{U}$ (excluding the unit diagonal of $\mathbf{U}$), I = 1, 2,..., N. IFPTR(N+1) is set to one greater than the number of entries in $\mathbf{L}+\mathbf{U}$ (excluding the unit diagonal of $\mathbf{U}$).

IW is an INTEGER array of length at least 5*N+NZ and is used by the routine as workspace. On exit, IW(I) is the position in the original matrix $\mathbf{A}$ of row I in the permuted matrix $\mathbf{QA}$, I = 1, 2,..., N.

ICNTL is an INTEGER array of length 4 that contains control parameters. Default values for the components may be set by a call to MI11I/ID. Details of the control parameters are given in Section 2.2.1 This argument is not altered by the routine.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 2 that contains control parameters and must be set by the user. Default values for the components may be set by a call to MI11I/ID. Details of the control parameters are given in Section 2.2.1. This argument is not altered by the routine.

INFO is an INTEGER array of length 6 that need not be set by the user . It is used to hold information about the execution of the subroutine. On exit from MI11A/AD, a value for INFO(1) of zero indicates that the subroutine has performed successfully. For nonzero values, see Section 2.3. For information output in the other components, see Section 2.2.2.

2.1.3 To perform preconditioning operations

The single precision version

```
CALL MI11B(TRANS,N,NZ,JCN,FACTOR,IFPTR,IW,Y,Z,W)
```

The double precision version

```
CALL MI11BD(TRANS,N,NZ,JCN,FACTOR,IFPTR,IW,Y,Z,W)
```

TRANS is a LOGICAL variable that must be set by the user . If TRANS=.TRUE., the preconditioning operation $\mathbf{y} = \mathbf{P}^{\mathrm{T}}\mathbf{z}$ is performed and if TRANS=.FALSE., the preconditioning operation $\mathbf{y} = \mathbf{P}\mathbf{z}$ is performed, where $\mathbf{P} = (\mathbf{L}\mathbf{U})^{-1}\mathbf{Q}$ is the ILU(0) preconditioner. This argument is not altered by the routine.

N,NZ,JCN,IFPTR,FACTOR must all be unchanged since the call to MI11A/AD. These arguments are not altered by the routine.

IW is an INTEGER array of length at least 2*N. The first 2*N entries must be unchanged since the call to MI11A/AD. This argument is not altered by the routine.

Y is a REAL (DOUBLE PRECISION in the D version) array of length N that need not be set by the user . On exit, Y holds the preconditioned vector $\mathbf{y}$.

Z is a REAL (DOUBLE PRECISION in the D version) array of length N that must be set by the user to hold the vector $\mathbf{z}$ which is to be preconditioned. This argument is not altered by the routine.

W is a REAL (DOUBLE PRECISION in the D version) array. If TRANS=.TRUE., W must be of length at least N and is used by the routine as workspace. If TRANS=.FALSE., W is not accessed.

2.2 Arrays for control and information

2.2.1 Control parameters

The elements of the arrays ICNTL and CNTL control the action of MI11A/AD. Default values may be set by calling MI11I/ID.

ICNTL(1) is the stream number for error messages and has the default value 6. Printing of error messages is suppressed if ICNTL(1) $\leq$ 0.

ICNTL(2) is the stream number for warning messages and has the default value 6. Printing of warning messages is suppressed if ICNTL(2) $\leq$ 0.

ICNTL(3) controls whether the user wants the input data to be checked for errors. It has default value 0 and, in this case, the input data is checked for errors, including out-of-range indices, duplicated entries, and matrix entries with value zero. If ICNTL(3) is nonzero, only the input arguments N and NZ are checked.

ICNTL(4) has default value 0 and, in this case, the user must input the matrix **A** by column indices and row pointers. Otherwise, the user must input **A** by row indices and column pointers.

CNTL(1),CNTL(2) are threshold tolerances. CNTL(1) has default value 0.0001 and CNTL(2) has default value 1.0. If at any stage k of the elimination process, the diagonal entry $\mathbf{a}_{kk}^{(k)}$ is such that

$$|\mathbf{a}_{kk}^{(k)}| < \texttt{CNTL(1)} * \max_{j>k} |\mathbf{a}_{kj}^{(k)}| \qquad \text{and} \qquad |\mathbf{a}_{kk}^{(k)}| < \texttt{CNTL(1)} * \max_{j>k} |\mathbf{a}_{jk}^{(k)}|,$$

$\mathbf{a}_{kk}^{(k)}$ is increased to $\min(\max_{j>k} |\mathbf{a}_{kj}^{(k)}|, \max_{j>k} |\mathbf{a}_{jk}^{(k)}|)$. If the absolute value of the (increased) diagonal entry is less than CNTL(1), the diagonal entry is given the value CNTL(2). If CNTL(2) is less than $\sqrt{u}$, where u is the relative machine precision, it is reset to the default value 1.0.

2.2.2 Information returned to the user

The array INFO is used to hold information which is returned to the user.

INFO(1) has the value zero if the call was successful, a positive value if a warning was issued, and a negative value in the event of a fatal error (see Section 2.3).

INFO(2) holds the number of out-of-range indices. These indices are ignored by the routine.

INFO(3) holds the number of multiple entries in the input matrix. Such entries are summed.

INFO(4) holds the number of entries in the input matrix with the value zero. These entries are ignored by the routine.

INFO(5) holds the number of entries in the input matrix if the out-of-range and zero entries are ignored and multiple entries are summed. This equal to IFPTR(N+1)-1 on exit.

INFO(6) holds the number of diagonal entries $\mathbf{a}_{kk}^{(k)}$ during the elimination process which were increased in size (see CNTL(1) and CNTL(2) in Section 2.2.1).

2.3 Error diagnostics

If MI11A/AD returns with a negative value of INFO(1), an error has occurred; if MI11A/AD returns with a positive value of INFO(1), a warning has been issued. Error messages are output on unit ICNTL(1) and warnings on unit ICNTL(2). Possible non-zero values of INFO(1) are given below.

-1 N < 1. Immediate return with input parameters unchanged.

-2 NZ $<\max($IP(N+1)-1,N$)$. Immediate return with input parameters unchanged.

-3 The matrix **A** is structurally singular (a permutation matrix **Q** such that **QA** has nonzero diagonal entries cannot be found). Immediate return.

+1 Either out-of-range indices, or multiple entries, or zero entries in the input matrix, or one or more of the diagonal entries $\mathbf{a}_{kk}^{(k)}$ during the elimination process was increased in size. See INFO(I) (I = 2, 3, ..., 6) in Section 2.2.2 for full details.

3 GENERAL INFORMATION

Use of common: None.

Other routines called directly: MI11A/AD calls the internal routine MI11C/CD. In addition, HSL routines FD15A/AD, MC21A/AD, MC22A/AD, and MC38A/AD are called.

Input/output: Error messages are printed on unit ICNTL(1) and warnings on unit ICNTL(2); see Section 2.3.

Restrictions: $\texttt{N} \geq 1$, $\texttt{NZ} \geq \max(\texttt{IP(N+1)-1}, \texttt{N})$.

4 METHOD

The input data is first checked for errors. If a fatal error is found, an error flag is set in INFO(1) and control is returned to the user. If ICNTL(3) $= 0$ (the default), the input matrix is checked for out-of-range indices, duplicated entries, and matrix entries with value zero and, if necessary, a warning flag is set in INFO(1). If the user has input the matrix by row indices and column pointers (ICNTL(4) $\neq 0$), MC38 is used to reorder the matrix by column indices and row pointers. The diagonal entries of **A** are then considered. If there are no zeros on the diagonal, the permutation **Q** is set to the identity. Otherwise, MC21 is used to find a row permutation **Q** such that **QA** has no zeros on the diagonal. If MC21 finds that **A** is structurally singular, an error flag is set in INFO(1) and control is returned to the user. Otherwise, MC22 is used to perform the permutation **QA**. The entries of the rows of **QA** are ordered by columns using an in-place sort.

The incomplete **LU** factorization of **QA** is then performed. No fill-in is allowed so that all fill-in elements are dropped as soon as they are introduced at an elimination step. Thus an entry in the **L** or **U** factor is nonzero only if the corresponding entry in the permuted matrix **QA** is nonzero.

5 EXAMPLE OF USE

We wish to solve the system of equations $\mathbf{Ax} = \mathbf{b}$ where

$$\mathbf{A} = \begin{pmatrix} 2 & 1 & -1 & 0 \\ 2 & 3 & 0 & 1 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & -1 & 2 \end{pmatrix}, \mathbf{b} = \begin{pmatrix} 2 \\ 6 \\ 3 \\ 1 \end{pmatrix},$$

by forming the incomplete LU factorization of **A** and using it as a preconditioner for the bi-conjugate gradient method. The following program, which illustrates the calling sequence for MI11, may be used.

```
C  Form the incomplete LU factorisation of A,
C  and use as a preconditioner to solve Ax = b with bi-conjugate
C  gradient method.

C     .. Parameters ..
      DOUBLE PRECISION ZERO
      PARAMETER (ZERO=0.0D0)
      INTEGER MAXN,MAXNZ
      PARAMETER (MAXN=5,MAXNZ=15)
C     ..
C     .. Local Scalars ..
      DOUBLE PRECISION RESID
```

```
      INTEGER B,I,IACT,IE,IR,IS,J,N,NZ
C     ..
C     .. Local Arrays ..
      DOUBLE PRECISION A(MAXNZ),CNTL(2),CNTL1(5),FACTOR(MAXNZ),
     +                 WORK(MAXN),W(MAXN,7),RSAVE(8)
      INTEGER ICNTL(4),ICNTL1(8),INDEX(MAXNZ),INFO(6),IFPTR(MAXN+1),
     +        IP(MAXN+1),IW(5*MAXN+MAXNZ),JCN(MAXNZ),
     +        JNFO(4),LOCY(2),LOCZ(2),ISAVE(14)
C     ..
C     .. External Subroutines ..
      EXTERNAL MI11AD,MI11BD,MI11ID,MI25AD,MI25ID
C     ..
C
C Read the matrix in (INDEX,IP,A)
      READ (5,FMT=*) N
      READ (5,FMT=*) (IP(I),I=1,N+1)
      NZ = IP(N+1) - 1
      READ (5,FMT=*) (INDEX(I),I=1,NZ)
      READ (5,FMT=*) (A(I),I=1,NZ)
C
      CALL MI11ID(ICNTL,CNTL)

      CALL MI11AD(N,NZ,INDEX,A,IP,JCN,FACTOR,IFPTR,IW,ICNTL,CNTL,INFO)

      IF (INFO(1).LT.0) GO TO 130

C Now wish to solve Ax = b using incomplete LU  preconditioner.

C Read in right-hand side into first column of W
      B = 1
      READ (5,FMT=*) (W(I,B),I=1,N)

C Use bi-conjugate gradient method
      CALL MI25ID(ICNTL1,CNTL1,ISAVE,RSAVE)
C Preconditioning is required
      ICNTL1(3) = 1

      IACT = 0
   90 CONTINUE
      CALL MI25AD(IACT,N,W,MAXN,LOCY,LOCZ,RESID,ICNTL1,CNTL1,JNFO,
     +            ISAVE,RSAVE)

      IF (IACT.LT.0) GO TO 130

      IF (IACT.EQ.2) THEN
C  Perform  matrix-vector products with A and A(T).
         DO 100 I = 1,N
            W(I,LOCY(1)) = ZERO
            W(I,LOCY(2)) = ZERO
```

```
  100    CONTINUE
         DO 120 I = 1,N
            IS = IP(I)
            IE = IP(I+1) - 1
            DO 110 IR = IS,IE
               J = INDEX(IR)
               W(I,LOCY(1)) = W(I,LOCY(1)) + A(IR)*W(J,LOCZ(1))
               W(J,LOCY(2)) = W(J,LOCY(2)) + A(IR)*W(I,LOCZ(2))
  110       CONTINUE
  120    CONTINUE
         GO TO 90
       END IF

       IF (IACT.EQ.3) THEN
C  Use MI11B/BD to perform the preconditioning operations
         CALL MI11BD(.FALSE.,N,NZ,JCN,FACTOR,IFPTR,IW,W(1,LOCY(1)),
     +              W(1,LOCZ(1)),WORK)
         CALL MI11BD(.TRUE.,N,NZ,JCN,FACTOR,IFPTR,IW,W(1,LOCY(2)),
     +              W(1,LOCZ(2)),WORK)
         GO TO 90
       END IF

C  Solution found
       WRITE (6,FMT=9010) JNFO(2), (W(I,2),I=1,N)

  130 CONTINUE

 9010 FORMAT (//I6,' iterations required by MI25 ',//' Solution = ',
     +       / (1P,5D10.2))
       END
```

The input data used is:

```
  4
  1  4  7  9 11
  3  1  2  4  2  1  2  3  4  3
  -1.0   2.0   1.0   1.0   3.0   2.0   2.0   1.0   2.0  -1.0
   2.0   6.0   3.0   1.0
```

This produces the following output:

```
     3 iterations required by MI25

 Solution =
  1.00D+00  1.00D+00  1.00D+00  1.00D+00
```