1 SUMMARY

Given the sparsity pattern of a symmetric matrix **A**, MC47 uses an **approximate minimum degree algorithm** to compute a pivot order that is efficient when used with a sparse Cholesky solver. MC47 optionally allows for the efficient handling of dense or almost dense rows of **A**. At each step, the pivot selected is the one that minimizes an upper-bound on the (external) degree. A permutation corresponding to this ordering is returned, together with information that may assist in the subsequent numerical factorization of the matrix.

The version of the approximate minimum degree algorithm implemented in MC47 is described in detail in the report
An approximate minimum degree algorithm for matrices with dense rows,
P. R. Amestoy, H. S. Dollar, J. K. Reid, J. A. Scott.

ATTRIBUTES — Version: 2.1.0. (31 October 2007) **Types:** Real (single, double). **Calls:** MC34, MC59. **Language:** Fortran 77. **Original date:** September 1995. **Origin:** Version 1: P. R. Amestoy (ENSEEIHT, Toulouse, France), T. A. Davis (University of Florida) and I. S. Duff (Rutherford Appleton Laboratory). Version 2: P. R. Amestoy (ENSEEIHT, Toulouse, France), H. S. Dollar, J. K. Reid and J. A. Scott (Rutherford Appleton Laboratory).

2 HOW TO USE THE PACKAGE

There are three entries to this package:

(a) MC47I/ID may be called to initialize the parameters that control the execution of the package.

(b) MC47A/AD is optional. It performs extensive checking and reordering of the user's data before calling MC47B/BD to compute an approximate minimum degree ordering. Users should note that the time taken for the initial checking and reordering can be comparable to that taken by MC47B/BD.

(c) MC47B/BD computes an approximate minimum degree ordering. No data checking is performed.

2.1.1 The initialization subroutine

To initialize control parameters, the user may make a call of the following form:

The single precision version

```
CALL MC47I(ICNTL)
```

The double precision version

```
CALL MC47ID(ICNTL)
```

ICNTL is an INTEGER array of length 10 that need not be set by the user. This array is used to hold control parameters. On exit, ICNTL contains default values. If the user wishes to use values other than the defaults, the corresponding entries in ICNTL should be reset after the call to MC47I/ID. The default values are as follows:

ICNTL(1) is the stream number for error messages. It has the default value 6. Printing of error messages is suppressed if ICNTL(1) < 0.

ICNTL(2) is the stream number for warning messages. It has the default value 6. Printing of warning messages is suppressed if ICNTL(2) < 0.

ICNTL(3) is the stream number for printing of the matrix data on entry to and exit from MC47A/AD. It has the default value -1. Printing is suppressed if ICNTL(3) < 0.

ICNTL(4) controls the choice of AMD algorithm. The options (which are explained more fully in Section 4) are:
= -1 No dense row detection

= 0 Only rows that are completely full are treated as dense
= 1 Automatic selection of dense rows
The default value is 1.

ICNTL(5) defines the largest integer that can be stored on the machine being used. In Fortran 95, this is huge(1). The default value is 2139062143.

ICNTL(6) to ICNTL(10) are given the default value 0. They are currently not used but may be used in a later release of the code.

2.1.2 Argument lists for main entry

The user may specify the pattern of the matrix either by providing the row indices of the subdiagonal entries of the columns or by providing the row and column indices of each entry in the lower triangle. The diagonal is always assumed to be nonzero and need not be input.

The single precision version

```
CALL MC47A(N, NE, PE, IW, IWLEN, ICNTL, INFO, RINFO)
```

The double precision version

```
CALL MC47AD(N, NE, PE, IW, IWLEN, ICNTL, INFO, RINFO)
```

N is an INTEGER variable that must be set by the user to the order of the matrix **A**. It is not altered.
Restriction: N ≥ 1.

NE is an INTEGER variable that must be set by the user to the number of entries being provided. It is not altered.

PE is an INTEGER array of size N+1 that must be set by the user. If the user is supplying the entries by columns, PE(I) must hold the index in IW of the start of column I, I=1, ..., N and PE(N+1) must be equal to NE+1. If the user is supplying row and column indices for the entries of **A**, PE(1) must be negative. On exit, PE will hold information on the matrix factors (see Section 2.3).

IW is an INTEGER array of length IWLEN that must be set by the user to hold the pattern of the matrix **A**. Duplicates, out-of-range entries, diagonal entries, and entries in upper triangle are ignored. If PE(1) is positive, IW(PE(J)), ..., IW(PE(J+1)–1) must hold the row indices of entries in column J, J = 1, ..., N. The entries within a column need not be in order. If PE(1) is negative, then (IW(k), IW(NE+k)), k = 1, ..., NE, must hold the row and column index of an entry. IW is used as workspace by the subroutine. On exit, the permutation generated by MC47 is held in IW(IWLEN-N+1), ..., IW(IWLEN) and is such that the k th column of the permuted matrix is column IW(IWLEN-N+k) of the original matrix. The inverse permutation is held in positions IWLEN-2*N+1 to IWLEN-N of IW, preceded by further information on the structure of the factors (see Section 2.3).

IWLEN is an INTEGER variable. It must be set by the user to the length of array IW. It is not altered. We recommend IWLEN > 2**nz* + 9*N, where *nz* is number of valid entries in input matrix. **Restriction**: IWLEN ≥ 2**nz* + 8*N.

ICNTL is an INTEGER array of length 10 that must be set by the user to hold control parameters. Default values may be set by a call to MC47I/ID. It is not altered.

INFO is an INTEGER array of length 10 that need not be set by the user. On exit,

INFO(1) has the value zero if the call was successful, a negative value in the event of an error, and a positive value in the event of a warning (see Section 2.2 for details).

INFO(2) holds the number of compresses performed on the array IW. A large value for this indicates that the ordering could be found more quickly if IWLEN were increased.

INFO(3) holds the minimum necessary value for IWLEN for a successful run of MC47A/AD on the same matrix, without the need for any compresses of IW.

INFO(4) holds the number of entries with row or column indices that are out of range. Any such entry is

ignored.

INFO(5) holds the number of duplicate entries. Any such entry is ignored.

INFO(6) holds the number of entries in the upper triangle. Any such entry is ignored.

INFO(7) holds the number of diagonal entries. Any such entry is ignored.

INFO(8) holds the number of restarts performed.

INFO(9) and INFO(10) are currently set to 0.

RINFO is a REAL (DOUBLE PRECISION in the D version) array of length 10 that need not be set by the user. On exit,

RINFO(1) holds the forecast number of reals in the matrix factor.

RINFO(2) holds the forecast number of floating-point operations required to compute the matrix factor.

RINFO(3) to RINFO(10) are currently set to zero.

2.1.2 Argument list for the auxiliary entry.

No data checking is done by this routine. It is therefore imperative that the data is input in exactly the format requested.

The single precision version

```
      CALL MC47B(N, IWLEN, PE, PFREE, LEN, IW, NV, ELEN,
     *           LAST, DEGREE, HEAD, NEXT, W, ICNTL, JNFO, RJNFO)
```

The double precision version

```
      CALL MC47BD(N, IWLEN, PE, PFREE, LEN, IW, NV, ELEN,
     *            LAST, DEGREE, HEAD, NEXT, W, ICNTL, JNFO, RJNFO)
```

N is an INTEGER variable that must be set by the user to the order of the matrix. It is not altered.

IWLEN is an INTEGER variable that must be set to the length of IW. It must be larger than PFREE-2, but, for efficiency should be larger than PFREE+N. It is not altered.

PE is an INTEGER array of length N such that PE(I), I=1, ..., N, is the position of the first entry of column I in IW. PE is used as workspace by the routine and, on exit, holds information on the assembly tree (see Section 2.3).

PFREE is an INTEGER variable that must be set to one more than the number of entries. It is not altered.

LEN is an INTEGER array of length N that must be set so that the number of entries in column I is LEN(I), I=1, ..., N. It is not altered.

IW is an INTEGER array of length IWLEN, whose first PFREE-1 entries must be set to the row indices of the matrix entries. The entries in row I are held in positions IW(PE(I)) to IW(PE(I)+LEN(I)-1). They can be in any order. The matrix must be symmetric and, if entry (I, J) is present, entry (J, I) must also be present. There must be no out-of-range entries, duplicates, or diagonal entries.

NV is an INTEGER array of length N that need not be set on entry. It is used as workspace and, on exit, NV(I) holds either the degree of variable I when eliminated or is zero if the variable is not principal.

ELEN is an INTEGER array of length N that need not be set on entry. It is used as workspace and, on exit, holds the inverse permutation.

LAST is an INTEGER array of length N that need not be set on entry. It is used as workspace and, on exit, holds the permutation.

DEGREE, HEAD, NEXT, and W are all INTEGER arrays of length N that need not be set on entry. They are used as workspace by the routine.

ICNTL is an INTEGER array of length 10 that must be set by the user to hold control parameters. Default values may

be set by a call to MC47I/ID. Only ICNTL(4), ICNTL(5) and ICNTL(6) are used by MC47B/BD. It is not altered.

JNFO is an INTEGER array of length 10 that need not be set by the user. On exit,

JNFO(1) holds the number of compresses performed on the array IW. A large value for this indicates that the ordering could be found more quickly if IWLEN were increased.

JNFO(2) holds the number of restarts performed.

JNFO(3) to JNFO(10) are currently set to 0.

RJNFO is a REAL (DOUBLE PRECISION in the D version) array of length 10 that need not be set by the user. On exit,

RJNFO(1) holds the forecast number of reals in the matrix factor.

RJNFO(2) holds the forecast number of floating-point operations required to compute the matrix factor.

RJNFO(3) to RJNFO(10) are currently set to zero.

2.2 Error diagnostics

A successful return from MC47A/AD is indicated by a value of INFO(1) equal to zero. An error return is indicated by a negative value for INFO(1) on return from MC47A/AD. The errors that are trapped are:

–1 N < 1.

–2 IWLEN < 2*nz + 8*N, where nz is number of valid entries in the input matrix.

–3 Error in PE when input is by columns.

–4 Matrix is null, which is usually caused by the input of the upper triangle.

There is one warning return:

+1 Some input entries have been ignored because they were either out-of-range, duplicates, on the diagonal, or in the upper triangle. See INFO(4-7) for more details. The action taken is to ignore any such entries.

2.3 Information on structure of the factors.

Information on the structure of the matrix factor is returned by MC47A/AD and MC47B/BD. Although many users will not be interested in this, it can be useful for the subsequent factorization, for example if a multifrontal solution technique is being used.

On output, entries IWLEN-3*N+1 to IWLEN-2*N of IW hold the assembly tree. This represents a partial ordering from which the given pivot order can be obtained by a depth-first search. If IW(IWLEN-3*N+I)>0, then I represents a node in the assembly tree. In this case, node I is associated with element I, and IW(IWLEN-3*N+I) holds the true degree of element I at the time it was created (including the diagonal part). For a node I, -PE(I) is its parent, or is zero if Ii is a root. If IW(IWLEN-3*N+I)=0, then I is not a node in the assembly tree, but is a node that has been merged with another. The tree node into which it has been merged is -PE(I).

3 GENERAL INFORMATION

Use of common: none.

Other routines called directly: MC47A/AD calls MC34A/AD and MC59A/AD. MC47A/AD calls MC47B/BD.

Workspace: 8*N INTEGER workspace is required. Some output data is later returned in 3*N components of this array.

Input/output: Error messages, warning messages, and matrix data only. Error messages on unit ICNTL(1), which has default value 6. Warning messages on unit ICNTL(2), which has default value 6. Printing of these messages is suppresed if the appropriate unit number is set to a negative value. Matrix data is output on unit ICNTL(3), which has default value -1. Printing of matrix data is suppressed if ICNTL(3) < 0.

Restrictions: N ≥ 1, IWLEN ≥ 2*nz + 8*N, where nz is number of valid matrix entries input by the user.

Changes between Version 1 and Version 2:

Version 2 incorporates several additional features to those of Version 1. We give information on these in this section.

Automatic detection and efficient handling of dense or almost dense rows is now available. This is controlled using the new control parameter ICNTL(4) whose default is set so that automatic detection and efficient handling of dense or almost dense rows is performed. This requires no extra workspace. The number of restarts performed during the handling of dense or almost dense rows is returned in INFO(8) from MC47A/AD and JNFO(2) from MC47B/BD.

The package now conforms to the practice of newer packages by using ICNTL, JNFO, RINFO, and RJNFO arrays. Additionally, more detailed information is now returned.

The subroutine MC47I/ID has been added so that the user may initialize the control parameters.

4 METHOD

The MC47 method uses a quotient graph representation of the elimination process.

For the degree of nodes in the elimination graph, we use upper bounds that can be easily computed. Pivots with minimum external degree are then selected. Supervariable detection and mass elimination are also used.

If ICNTL(4) is equal to -1, the classical approximate minimum degree method is used. A full account of this method is given by Amestoy, Davis and Duff (An approximate minimum degree ordering, *SIAM J. Matrix Anal. Appl., 17 (1996), pp. 886-905*).

If the matrix **A** has some rows that have a significantly higher number of nonzero entries than the majority of the rows, then the classical approximate minimum degree algorithm may be slow and it is generally better to treat these *dense rows* separately: a dense row is regarded as being either completely full or quasi dense and no attempt is made to keep track of its degree. A row is sparse if it is not dense. If ICNTL(4) is equal to 0, only rows that are completely full will be classed as dense. If ICNTL(4) is equal to 1, the method will automatically select the dense rows: the classical approximate minimum degree method will be used if no dense rows are initially found; otherwise, the method proceeds by eliminating or reclassifying the sparse rows according to their approximate minimum degrees. When only dense rows remain, a restart is made; the degrees are recalculated and a fresh division between sparse, quasi dense and completely full rows is made. On each restart the criterion for determining whether a row is dense is updated.

The version of the approximate minimum degree method implemented in MC47 is described in detail in the report *An approximate minimum degree algorithm for matrices with dense rows,* P. R. Amestoy, H. S. Dollar, J. K. Reid, J. A. Scott.

5 EXAMPLES OF USE

We illustrate the use of MC47 on the following matrix:

$$\mathbf{A} = \begin{pmatrix} 1 & & & \times & & \times & & & & \\ & 2 & & & \times & \times & & & \times & \\ & & 3 & & \times & \times & \times & & & \\ \times & & & 4 & & & \times & \times & & \\ & \times & \times & & 5 & & \times & & \times & \\ \times & \times & \times & & & 6 & & & & \\ & & \times & \times & \times & & 7 & \times & \times & \times \\ & & & \times & & & \times & 8 & \times & \times \\ & \times & & & \times & & \times & \times & 9 & \times \\ & & & & & & \times & \times & \times & 10 \end{pmatrix}$$

Programs

```
C SIMPLE EXAMPLE OF USE OF MC47
C
        PROGRAM TEST
        INTEGER IWLEN, NMAX
        PARAMETER (IWLEN=200, NMAX=10)
        INTEGER N, NE, MP, INFO(10), ICNTL(10)
        INTEGER IW(IWLEN), PE(NMAX+1), I
        DOUBLE PRECISION RINFO(10)
C
C read input data N, NE and IW
        READ(5,*) N,NE
        IF (N.GT.NMAX .OR. IWLEN.LT.2*NE+8*N) THEN
          WRITE(6,*) 'Input problem too large for arrays declared'
          GO TO 100
        ENDIF
        READ(5,*) (IW(I),I=1,NE)
        READ(5,*) (IW(NE+I), I=1,NE)
        PE(1) = -1
        MP = 0
C
C Perform MC47 ordering
        CALL MC47ID(ICNTL)
        CALL MC47AD (N, NE, PE, IW, IWLEN, ICNTL, INFO, RINFO)
        IF (INFO(1).LT.0) THEN
          WRITE (6,*) ' ERROR detected during ordering'
          WRITE (6,*) ' INFO(1) = ',INFO(1)
        ELSE
C Print assembly tree, and ordering
          WRITE (6,'(A/(20I4))') 'Ordering:',(IW(I),I=IWLEN-N+1,IWLEN)
          WRITE (6,'(A/(20I4))') 'Parent array:',(PE(I),I=1,N)
          WRITE (6,'(A/(20I4))') 'Tree array:',
     *                           (IW(I),I=IWLEN-3*N+1,IWLEN-2*N)
        ENDIF
  100   STOP
        END
```

Input Data

```
10 18
 4  6  5  6  9  5  6  7  7  8  7  9  8  9 10  9 10 10
 1  1  2  2  2  3  3  3  4  4  5  5  7  7  7  8  8  9
```

Output data

```
Ordering:
   1   6  10   8   4   3   5   7   9   2
Parent array:
  -6   0  -2  -2  -2  -4  -2  -4  -2  -8
Tree array:
   3   5   0   5   0   4   0   4   0   4
```