Warning: Subroutine MC39 has been superseded by subroutine MC49 which uses improved algorithms; the use of the latter routine is recommended. The superseded routine may be removed from later releases of the library.

1 SUMMARY

This subroutine **sorts the sparsity pattern of a matrix to an ordering by columns.** There are two alternative entries. The subroutine MC39A/AD sorts the sparsity pattern from an arbitrary ordering to an ordering by columns, with an option for ordering the entries within each column by row indices. The subroutine MC39B/BD sorts the sparsity pattern of a matrix ordered by columns, with arbitrary order within each column, to an ordering by columns with ordering by row indices within each column.

ATTRIBUTES — Version: 1.0.0. **Remark:** This subroutine supersedes MC20. **Types:** MC39A, MC39AD. **Original date:** February 1988. **Origin:** I.S.Duff, J.K.Reid, and J.A.Scott, Harwell.

2 HOW TO USE THE PACKAGE

2.1 The alternative entries and the argument lists

There are two alternative entries.

(a) MC39A/AD sorts the entries of a sparse matrix from arbitrary order to column order. An option exists for ordering the entries within each column by row indices.

(b) MC39B/BD sorts the entries of a sparse matrix ordered by columns, with arbitrary order within each column, to an ordering by columns with ordering by row indices within each column.

2.1.1 The argument list for MC39A/AD.

To sort the entries of a sparse matrix from arbitrary order to column order

The single precision version

```
CALL MC39A(IND,NC,NR,NNZ,IRN,JCN,YESA,A,IP,IW)
```

The double precision version

```
CALL MC39AD(IND,NC,NR,NNZ,IRN,JCN,YESA,A,IP,IW)
```

IND is an INTEGER variable which must be set on entry to specify whether ordering by rows within each column is required. If IND is set to 0, the ordering within each column will be arbitrary. If IND is set to 1, the ordering within each column will be by row indices. This argument is not altered by MC39A/AD. **Restriction:** IND=0 or 1.

NC is an INTEGER variable which must be set to the number of columns in the matrix. This argument is not altered by MC39A/AD.

NR is an INTEGER variable which must be set to the number of rows in the matrix. This argument is not altered by MC39A/AD. **Restriction:**

NNZ is an INTEGER variable which must be set to the number of entries in the matrix. This argument is not altered by MC39A/AD.

IRN is an INTEGER array of length NNZ which must be set to contain the row indices of the entries in the matrix. The entries may be in any order. On exit, the row indices are reordered so that the entries of a single column are contiguous with column J preceding column J+1 (J=1,...,N-1), with no space between columns. If IND=0, the order within each column is arbitrary; if IND=1, the order within each column is by row indices.

JCN is an INTEGER array of length NNZ. On entry, JCN(K) must be set to the column index of the entry held in IRN(K) (K=1,2,...,NNZ). If IND=1 and YESA=.FALSE., the contents of this array are destroyed by the subroutine; otherwise, on exit, the array will have been permuted in the same way as the array IRN.

YESA is a LOGICAL variable. If set to .TRUE., the values of the entries in the matrix **A** are sorted. If YESA is .FALSE., the array A is not accessed. This argument is not altered by MC39A/AD.

A is a REAL (DOUBLE PRECISION in the D version) array. If YESA is .FALSE., the array is not accessed. If YESA is .TRUE., the array must be of length NNZ and, on entry, must be set so that A(K) holds the value of the entry in IRN(K) (K=1,2,...,NNZ). On exit, the array will have been permuted in the same way as the array IRN.

IP is an INTEGER array. If IND=0, the array must be of length NC+1; if IND=1, the array must be of length max(NC,NR)+1. On entry, the array is not required to be set. On exit, IP(I) contains the position in the array IRN of the first entry in column I (I=1,2,...,NC), and IP(NC+1) is set to NNZ+1.

IW is an INTEGER array. If IND=0, the array must be of length NC+1; if IND=1, the array must be of length NR+1. This array is used as workspace.

2.1.2 The argument list for MC39B/BD.

To sort the entries within each column of a sparse matrix ordered by columns

The single precision version

```
CALL MC39B(NC,NR,NNZ,IRN,YESA,A,IP,IWORK,IW)
```

The double precision version

```
CALL MC39BD(NC,NR,NNZ,IRN,YESA,A,IP,IWORK,IW)
```

NC is an INTEGER variable which must be set to the number of columns in the matrix. This argument is not altered by MC39B/BD.

NR is an INTEGER variable which must be set to the number of rows in the matrix. This argument is not altered by MC39B/BD. **Restriction:**

NNZ is an INTEGER variable which must be set to the number of entries in the matrix. This argument is not altered by MC39B/BD.

IRN is an INTEGER array of length NNZ. On entry, the array must be set to contain the row indices of the entries in the matrix, ordered so that the entries of a single column are contiguous with column J preceding column J+1 (J=1,...,N-1), with no space between columns. On exit, the entries are reordered so that the order within each column is by row indices.

YESA is a LOGICAL variable. If set to .TRUE., the values of the entries in the matrix **A** are sorted. If YESA is .FALSE., the array A is not accessed. This argument is not altered by MC39B/BD.

A is a REAL (DOUBLE PRECISION in the D version) array. If YESA is .FALSE., the array is not accessed. If YESA is .TRUE., the array must be of length NNZ and, on entry, must be set so that A(K) holds the value of the entry in IRN(K) (K=1,2,...,NNZ). On exit, the array will have been permuted in the same way as the array IRN.

IP is an INTEGER array of length NC+1. On entry, IP(I) must be set to the position in the array IRN of the first entry in column I (I=1,2,...,NC), and IP(NC+1) must be set to NNZ+1. The array is not altered by MC39B/BD.

IWORK is an INTEGER array of length NNZ. This array is used as workspace.

IW is an INTEGER array of length NR+1. This array is used as workspace.

3 GENERAL INFORMATION

Workspace: The array IW is used by MC39A/AD and MC39B/BD as workspace; see §2.1. In addition, the array IWORK is used by MC39B/BD as workspace.

Use of common: None.

Other routines called directly: The subroutine MC39A/AD calls the internal subroutine MC39C/CD and, if IND=1, the internal subroutine MC39D/DD. The subroutine MC39B/BD calls the internal subroutine MC39D/DD.

Input/output: None

Restrictions:

IND=0 or 1,

If these restrictions are violated the subroutine immediately returns without changing any of the input parameters.

4 METHOD

To sort the entries into column order, with arbitrary order within each column, the following procedure is used. The number of entries in each column is counted. The space needed by each column is calculated and pointers are set to the first entry in each column. Each column is considered in turn, and each entry in the column which has not already been allocated is, in turn, made the current entry and examined to see if it is in place. If not, it is put into the first free location in its correct column, the pointer for that column is increased by one, and the displaced entry is made the current entry. This chain of displacing entries continues until an entry which belongs to the column of the first entry in the chain is found. This entry is stored in the position vacated by the first entry in the chain and the next item in that column is then examined.

To sort the entries from arbitrary order to column order, with ordering by row indices within each column, the above procedure is followed using rows in place of columns to obtain an ordering by rows, with arbitrary order within each row. The number of entries in each column is then obtained by a counting pass. The final position of each entry is determined by examining each row in turn using a similar algorithm to that discussed above. Finally the entries are permuted into these positions.

To sort the entries from column order, with arbitrary order within in each column, to column order with ordering by row indices within each column, the algorithm outlined above for ordering from row order to column order is followed with columns in place of rows. This gives an ordering by rows, with ordering by column indices within each row. By interchanging the roles of rows and columns and repeating this step, the final ordering is obtained.

5 EXAMPLE OF USE

The following example illustrates the use of MC39A. Using the program:

```
INTEGER IRN(9),JCN(9),IP(5),IW(5),IND,NC,NR,NNZ
REAL      A(1)
LOGICAL YESA
DATA IND /1/
DATA YESA /.FALSE./
READ(5,*)NR,NC,NNZ
READ(5,*)(JCN(I),I=1,NNZ)
READ(5,*)(IRN(I),I=1,NNZ)
CALL MC39A(IND,NC,NR,NNZ,IRN,JCN,YESA,A,IP,IW)
WRITE(6,*)(IP(I),I=1,NC+1)
```

```
      WRITE(6,*)(IRN(I),I=1,NNZ)
      STOP
      END
```

on the data

```
5    4    9
1    3    4    1    2    4    2    4    2
1    3    3    3    1    1    2    4    5
```

produces the output:

```
1    3    6    7   10
1    3    1    2    5    3    1    3    4
```