1 SUMMARY

This subroutine **calculates a symmetric permutation that reduces the profile of a sparse matrix with a symmetric sparsity pattern.** The profile of a matrix is the total number of coefficients in the lower triangle when any zero ahead of the first entry in its row is excluded. For an $n \times n$ $\mathbf{A} = \{a_{ij}\}$ the profile is

$$\sum_{i=1}^{n} \max_{\substack{a_{ij} \neq 0 \\ j<i}} |i+1-j| \ .$$

ATTRIBUTES — **Version:** 1.0.0. **Types:** MC40A, MC40AD. **Calls:** MC34, MC49. **Original date:** February 1988. **Origin:** J.A.Scott, Harwell and S.W.Sloan, University of Newcastle, New South Wales.

2 HOW TO USE THE PACKAGE

2.1 Argument List

The single precision version

```
CALL MC40A(ITYPE,N,NNZ,IRN,JCN,ICPTR,IPERM,IW,IPROF,IFLAG)
```

The double precision version

```
CALL MC40AD(ITYPE,N,NNZ,IRN,JCN,ICPTR,IPERM,IW,IPROF,IFLAG)
```

`ITYPE` is an `INTEGER` variable which must be set by the user to `0` or `1` according to the method chosen to input the sparsity pattern of the matrix **A**. The subroutine allows the user to input the entries in the strict lower triangle of **A** in one of two ways. If `ITYPE`$=$`0`, the entries must be ordered by columns. If `ITYPE`$=$`1`, the entries may be input in any order. This argument is not altered by the routine. **Restriction:** `ITYPE=0` or `1`.

`N` is an `INTEGER` variable which must be set by the user to the order of the matrix **A**. This argument is not altered by the routine. **Restriction:** `N`$\geq$`1`.

`NNZ` is an `INTEGER` variable which must be set by the user to the number of entries in the strict lower triangle of the matrix **A**. This argument is not altered by the routine. **Restriction:** `NNZ`$\geq$`0`.

`IRN` is an `INTEGER` array of length `2*NNZ`. The first `NNZ` entries must be set by the user to the row indices of the entries in the strict lower triangle of the matrix **A**. If `ITYPE`$=$`0`, the entries of a single column must be contiguous. The entries of column `J` must precede those of column `J+1` (`J=1,...,N-1`), and there must be no wasted space between the columns. Row indices within a column may be in any order. If `ITYPE`$=$`1`, the entries in `IRN` may be in any order. On exit, the array will be changed to hold the row indices of the upper and lower triangular parts of the matrix **A**, with the diagonal entries excluded. The entries will be ordered by columns, with the entries of column `J` preceding those of column `J+1` (`J = 1,...,N-1`), and no wasted space between the columns.

`JCN` is an `INTEGER` array, which need only be set by the user if `ITYPE`$=$`1`. If so, `JCN` must have length `NNZ` and must be set to the column indices of the entries in the strict lower triangle of **A**, in the same order as in the array `IRN`. The contents of this array are altered by the subroutine. If `ITYPE`$=$`0`, the array is not accessed. If `ITYPE=1`, the user may conserve storage by equivalencing `JCN(1)` to `IRN(K)`, `K`$>$`NNZ`.

`ICPTR` is an `INTEGER` array of length `N+1`, which need only be set by the user if `ITYPE`$=$`0`. If so, it must be set so that `ICPTR(J)` points to the position in the array `IRN` of the first entry in column `J` of the strict lower triangle of **A** (`J=1,...,N`), and `ICPTR(N+1)` must be set to `NNZ+1`. On exit, the array is set by the subroutine so that `ICPTR(J)` points to the position in the array `IRN` of the first entry in column `J` of **A** (`J = 1,...,N`), and

ICPTR(N+1) is set to one greater than the number of entries in **A** (diagonal entries excluded).

IPERM is an INTEGER array of length N. This array need not be set on entry. On exit, the new ordering is contained in IPERM. The new index for variable I is given by IPERM(I) (I = 1,...,N).

IW is an INTEGER array of length at least 3*N+2. The array will be used by the subroutine as workspace.

IPROF is an INTEGER array of length 2 which need not be set by the user. On exit, the first entry in the array will contain the profile corresponding to the original ordering, and the second entry will contain the profile corresponding to the new ordering.

IFLAG is an INTEGER variable which need not be set by the user. On exit, a value of zero indicates that the subroutine has performed successfully. Negative values indicate a fatal error; a positive value indicates a warning. For nonzero values see § 2.3.

2.2 Common blocks

One common block is used. The common block is:

The single precision version

```
COMMON/ MC40I/ LP,MP
```

The double precision version

```
COMMON/ MC40ID/ LP,MP
```

where the parameters are given default values by a block data subprogram MC40K/KD. These parameters are not altered by the subroutine.

LP is an INTEGER variable used as the unit number of the device to which error messages are sent. The default value is 6. Error messages can be suppressed by setting LP = 0.

MP is an INTEGER variable used as the unit number of the device to which warning messages are sent. The default value is 6. Warning messages can be suppressed by setting MP = 0.

2.3 Errors and diagnostic messages

A successful return from the subroutine is indicated by a value of IFLAG equal to zero. Possible nonzero values of IFLAG are given below. In each case an identifying message is output on unit LP (errors) or MP (warnings).

A negative value of IFLAG is associated with an error message, which will be output on unit LP.

–1 – N < 1. Immediate return with input parameters unchanged.
–2 – NNZ < 0. Immediate return with input parameters unchanged.
–3 – The user has violated the restriction ITYPE=0 or 1. Immediate return with input parameters unchanged.

A positive value of IFLAG is associated with a warning message, which will be output on unit MP.

+1 – The user has input entries which belong to the diagonal of the matrix **A**. The subroutine ignores these entries.
+2 – The user has input entries which belong to the upper triangle of the matrix **A**. The subroutine ignores these entries. If the user has input entries belonging to the upper triangle and to the diagonal of **A**, IFLAG = +2 is returned.
+3 – The user has input row/ column indices I/ J outside the range 1 ≤ I, J ≤ N. The subroutine ignores these entries. If row or column indices outside the range are input together with entries belonging to the upper triangle or diagonal of **A**, IFLAG = +3 is returned.

3 GENERAL INFORMATION

Workspace: The array IW of length 3*N+2 is used as workspace.

Use of common: The subroutine uses common block MC40I/ID; see § 2.2.

Other routines called directly: MC40A/AD calls internal subroutines MC40B/BD, MC40C/CD, MC40D/DD, MC40E/ED, MC40F/FD, MC40G/GD, and MC40H/HD, and the subroutine MC34A/AD. MC40A/AD also calls a block data subprogram MC40K/KD. In addition, if ITYPE = 1, MC40A/AD calls MC49A/AD. None of these subroutines need be called directly by the user.

Input/output: Error messages on unit LP (LP = 0 suppresses them). Warning messages on unit MP (MP = 0 suppresses them).

Restrictions:

N ≥ 1,

NNZ ≥ 0,

ITYPE=0 or 1.

4 METHOD

The algorithm is comprised of two distinct steps.

The graph associated with the $n \times n$ matrix **A** has n nodes and an edge between nodes i and j for each entry a_{ij}. In the first step of algorithm, for each component of the graph (set of connected nodes), a pair of well-separated nodes (pseudo-peripheral nodes) are selected using a procedure which is a modification of that given by Gibbs et al. (1976).

In the second step, the nodes in each component are renumbered to obtain a smaller profile. The pseudo-peripheral nodes found in step one serve as the first and last nodes for the labelling within each component. Each successive node is chosen to minimize a weighted average of the distance in the graph from the last node and the number of neighbouring nodes that have not already been chosen. Once all the variables have been assigned new indices, the code checks that the corresponding profile is less than the initial profile. If this is not the case, the initial ordering is retained.

The computed profiles include the diagonal terms.

The algorithm was developed by Scott Sloan and is described in detail by Sloan (1986).

References

Gibbs, N.E., Poole, W.G., Stockmeyer, P.K. (1976) An algorithm for reducing the profile and bandwidth of a sparse matrix. SIAM J. Numer. Anal. **13**, 236-250.

Sloan, S.W. (1986) An algorithm for profile and wavefront reduction of sparse matrices. Inter. J. Numer. Meth. Engng **23**, 239-251.

5 EXAMPLE OF USE

The following program provides an example of the use of MC40. We wish to reduce the profile of a matrix with the following sparsity pattern.

$$\mathbf{A} = \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{pmatrix}.$$

The input data will be in column-wise format. Using the program:

```
      INTEGER IRN(8),JCN(1),ICPTR(6),IPERM(5),IW(17),IPROF(2),
     *        N,NNZ,ITYPE,IFLAG,LP,MP
```

```
      COMMON /MC40I/ LP,MP
C
C Read in data
      READ (5,*) N,NNZ
      READ (5,*) (IRN(I),I = 1,NNZ)
      READ (5,*) (ICPTR(I),I = 1,N+1)

C Prepare to call MC40
      ITYPE = 0
      CALL MC40A(ITYPE,N,NNZ,IRN,JCN,ICPTR,IPERM,IW,IPROF,IFLAG)

C Check for an error return
      IF (IFLAG.LT.0) GO TO 100

C Write out the profile
      WRITE (6,210) IPROF(1)
      IF (IPROF(1).EQ.IPROF(2))THEN
         WRITE (6,200)
      ELSE
         WRITE (6,220) IPROF(2)
      END IF
100   STOP
C
200   FORMAT(/5X,'The algorithm did not reduce the profile')
210   FORMAT(/5X,'The profile initially is',I3)
220   FORMAT(/5X,'The reduced profile is',I3)
      END
```

on the data

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

produces the output:

```
      The profile initially is 15

      The reduced profile is  9
```

The pattern of the reordered matrix is:

$$\begin{pmatrix} \times & & & \times & \\ & \times & & \times & \\ & & \times & \times & \\ \times & \times & \times & \times & \times \\ & & & \times & \times \end{pmatrix}.$$