1 SUMMARY

Given a **sparse matrix** $\mathbf{A}=\{a_{ij}\}_{n\times n}$ and a row permutation matrix **P** and a column permutation matrix **Q**, this subroutine **performs the permutation** $\tilde{\mathbf{A}}=\mathbf{PAQ}$.

The nonzero elements of **A** are stored by rows in a compact form and the user defines the permutation matrices **P** and **Q** by index vectors of length *n*.

The code is described in I.S.Duff, Harwell report R.8730 (1977).

ATTRIBUTES — Version: 1.0.0. (12 July 2004) **Types:** Real (single, double), Complex (single, double). **Remark:** Supersedes ME22. **Original date:** November 1976. **Origin:** I. S. Duff, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Argument list

The single precision version

```
CALL MC22A(N,ICN,A,NZ,LENROW,IP,IQ,IW,IW1)
```

The double precision version

```
CALL MC22AD(N,ICN,A,NZ,LENROW,IP,IQ,IW,IW1)
```

The complex version

```
CALL MC22AC(N,ICN,A,NZ,LENROW,IP,IQ,IW,IW1)
```

The double precision complex version

```
CALL MC22AZ(N,ICN,A,NZ,LENROW,IP,IQ,IW,IW1)
```

`N` is an `INTEGER` variable which must be set by the user to the order of the matrix. This argument is not altered.

`ICN` is an `INTEGER` array of length at least `NZ`, the number of matrix nonzeros. The first `NZ` entries of this array must be set by the user to contain the column indices of the nonzeros of the original matrix. Those belonging to a single row must be contiguous but the ordering of column indices within each row is unimportant. The nonzeros of row `I` must precede those of row `I+1` (`I` = 1, ..., `N`–1), and no wasted space is allowed between the rows. On output the column indices of **PAQ** are held in positions 1 to `NZ`, again without any wasted space between the rows and without ordering within each row.

`A` is a `REAL` (`DOUBLE PRECISION` in the `D` version, `COMPLEX` in the `C` version, or `COMPLEX*16` in the `Z` version) array of length at least `NZ`, whose elements must be set by the user to the values of the nonzero entries of the matrix in the corresponding positions in `ICN`. On output, the entries of `A` will be permuted in an exactly similar fashion to those of `ICN`.

`NZ` is an `INTEGER` variable which must be set by the user to the number of nonzeros in the matrix.

`LENROW` is an `INTEGER` array of length `N`. On input, `LENROW(I)` should be set by the user to be the number of nonzeros in row `I` (`I` = 1, ..., `N`) of the original matrix. On output `LENROW` will be permuted so that `LENROW(I)` is the number of nonzeros in row `I` of **PAQ**.

`IP` is an `INTEGER` array of length `N`, which must be set by the user so that row |`IP(I)`| of the original matrix **A** is row `I` of **PAQ** (`I` = 1, ..., `N`). The sign of `IP(I)` is immaterial. `IP` is not altered by the routine.

`IQ` is an `INTEGER` array of length `N`, which must be set by the user so that column |`IQ(I)`| of the original matrix **A** is column `I` of **PAQ** (`I` = 1, ..., `N`). The sign of `IQ(I)` is immaterial. `IQ` is not altered by the routine.

IW is an INTEGER array of length 2*N, which is used as workspace.

IW1 is an INTEGER array of length at least NZ, which is used as workspace.

2.2 Parameter usage summary

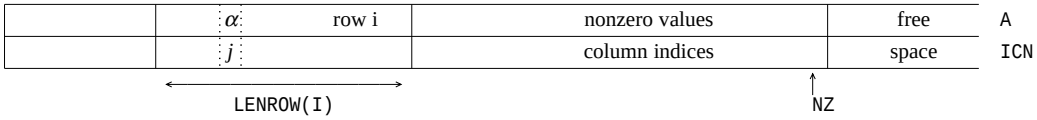
INPUT unchanged by MC22: N, NZ, IP(N), IQ(N).

INPUT changed by MC22: ICN(NZ), A(NZ), LENROW(N).

WORK ARRAYS: IW(N), IW1(NZ).

OUTPUT from MC22: ICN, A, LENROW.

2.3 Data structure



Value of entry (i,j) of **A** is α.

2.4 Errors and diagnostic messages

There are no error returns, but if NZ or N are less than or equal to zero, the subroutine immediately returns control to the calling program.

3 GENERAL INFORMATION

Workspace: Provided by user, see arguments IW and IW1.

Use of common: None.

Other routines called directly: None.

Input/output: None.

Restrictions: None.

4 METHOD

MC22 uses an in-place sort algorithm which performs the sort in O(NZ) operations.

A preliminary pass permutes LENROW, sets up a work array to identify which old row is in each new position, and calculates the amount (offset) by which each row must be moved to achieve the desired permutation. Each position in array ICN is then examined in turn to see whether it contains the nonzero which should be there in the final form. If this is not the case, its entry value and column index are stored temporarily and the entry which should be in this position (accessed through the work array and corresponding offset) is placed there. The position from which this new entry came now becomes the active position and the process continues in this chain-like fashion, until it is found that the entry which was in the original active position is required. At this stage, the information is taken from the temporary storage and the chain is complete. At each stage in the chain a flag is set in the work array to ensure the position is not processed during a subsequent scan, and when an entry is placed in its final position, its column index is changed according to the array IQ.

For further details, see I.S.Duff, MA28 – A set of Fortran subroutines for sparse unsymmetric linear equations, Harwell Report R 8730 (1977).

5 EXAMPLE OF USE

An example to permute the rows and columns of a sparse matrix

In the example code shown below, we read in the entries of the sparse matrix to be permuted (in any order) and the permutation arrays IP() and IQ() which represent the permutation matrices **P** and **Q** respectively. The entries are sorted into row order. We then call the routine MC22A/AD to compute the permuted matrix.

```
      INTEGER MAXN, MAXNZ, LICN
      PARAMETER( MAXN = 50 , MAXNZ = MAXN*MAXN ,
     +           LICN = 4*MAXNZ )
      INTEGER ICN(LICN), JNUM(MAXNZ), N, NZ, I, JPTR(MAXN+1),
     +        LENROW(MAXN), IP(MAXN), IQ(MAXN), IW(2*MAXN), IW1(MAXNZ)
      INTEGER JDISP, IPOS, J, ICNT59(10), INFO59(10)
      DOUBLE PRECISION A(LICN)

      READ(5,*) N, NZ
      IF ( (N .GT. MAXN) .OR. (NZ .GT. MAXNZ) ) THEN
         WRITE(6,550)
         STOP
      END IF

      READ(5, * ) ( ICN(I) , JNUM(I) , A(I) , I=1,NZ )
      READ(5, * ) ( IP(I) , IQ(I) , I=1,N )
      JDISP = 0
C     SORT THE INPUT MATRIX INTO ROW ORDER (MC22A/AD REQUIRES THIS)
      ICNT59(1) = 1
      ICNT59(2) = 0
      ICNT59(3) = 0
      ICNT59(4) = -1
      ICNT59(5) = -1
      ICNT59(6) = 0
      CALL MC59AD( ICNT59, N, N, NZ, JNUM, LICN, ICN, LICN, A, N+1,
     +             JPTR, N+1, IW, INFO59 )
C     JPTR(I) POINTS TO THE START OF THE I'TH ROW OF THE MATRIX
C     LENROW(I) HOLDS THE NUMBER OF ENTRIES IN THE I'TH ROW
      DO 100 I = 1,NZ
        ICN(I) = JNUM(I)
  100 CONTINUE
      DO 150 I = 1,N - 1
        LENROW(I) = JPTR(I+1) - JPTR(I)
  150 CONTINUE
      LENROW(N) = NZ - JPTR(N) + 1
      CALL MC22AD( N, ICN, A, NZ, LENROW, IP, IQ, IW, IW1)
      WRITE(6,570)
      IPOS = 1
      DO 250 I = 1 , N
        DO 200 J = IPOS , IPOS + LENROW(I) - 1
          WRITE(6,600) I , ICN(J) , A(J)
 200    CONTINUE
      IPOS = IPOS + LENROW(I)
 250  CONTINUE
 550  FORMAT( / ' ERROR IN THE INPUT DATA FOR N AND/OR NZ'//)
 570  FORMAT( / ' THE PERMUTED MATRIX IS:' //
     +         '      ROW   COLUMN          ENTRY' / )
 600  FORMAT( 2I8 , 8X , F13.5 )
      STOP
      END
```

To permute the following matrix:

$$\begin{pmatrix} 1 & 0 & 0 & 4 \\ 0 & 0 & 7 & 8 \\ 9 & 0 & 0 & 12 \\ 0 & 14 & 0 & 16 \end{pmatrix}$$

using the following permutation matrices

$$\mathbf{P} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \ \mathbf{Q} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

we could have as input

```
4        8
4        4           16.
1        1            1.
4        2           14.
2        3            7.
3        1            9.
2        4            8.
3        4           12.
1        4            4.
3        4
1        1
4        2
2        3
```

and we would get the following output

```
 THE PERMUTED MATRIX IS:

    ROW   COLUMN          ENTRY

     1        1            12.00000
     1        2             9.00000
     2        1             4.00000
     2        2             1.00000
     3        1            16.00000
     3        3            14.00000
     4        1             8.00000
     4        4             7.00000
```

which represents the following matrix

$$\tilde{\mathbf{A}} = \begin{pmatrix} 12 & 9 & 0 & 0 \\ 4 & 1 & 0 & 0 \\ 16 & 0 & 14 & 0 \\ 8 & 0 & 0 & 7 \end{pmatrix}$$