1 SUMMARY

This subroutine takes an m by n sparse matrix **A**, whose entries are stored by rows, and reorders it to be stored by columns.

ATTRIBUTES — **Version:** 1.0.0. (12 July 2004) **Types:** Real (single, double). **Original date:** January 1989. **Origin:** N.I.M. Gould, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Calling sequence

The single precision version

```
CALL MC46A ( NR, NC, LA, YESA, INDEX, A, IPR, JPC, IW, INFORM )
```

The double precision version

```
CALL MC46AD( NR, NC, LA, YESA, INDEX, A, IPR, JPC, IW, INFORM )
```

`NR` is an `INTEGER` variable set by the user to the number of rows, m, of the matrix **A**. It is unchanged by the subroutine. **Restriction:** `NR` ≥ 1.

`NC` is an `INTEGER` variable set by the user to the number of columns, n, of the matrix **A**. It is unchanged by the subroutine. **Restriction:** `NC` ≥ 1.

`LA` is an `INTEGER` variable which must be set by the user to be at least as large as the number of entries in the matrix **A**. This argument is unchanged by the subroutine. **Restriction:** `LA` ≥ 0.

`YESA` is a `LOGICAL` variable which must be set by the user. If only the sparsity pattern of **A** is required to be stored by columns on output from `MC46A/AD`, `YESA` should be set to `.FALSE.` on entry. If, in addition, the numerical values of the nonzero entries stored by columns are to be output, `YESA` should be set to `.TRUE.`. This argument is unchanged by the subroutine.

`INDEX` is an `INTEGER` array of length `LA`, which must be set by the user to the column indices of the nonzero entries of **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, ..., $`NR`). The ordering within each row is unimportant. On exit, `INDEX` contains the row indices of the entries of **A**. The entries will be ordered by columns, with the entries in each column contiguous and those of column j preceding those of column j+1 ($j = 1, ..., $`NC`). The entries are unordered within each column. **N.B.** If the entries are required to be ordered within each column, the user should call `MC59A/AD` after the call to `MC46A/AD`.

`A` is a `REAL` (`DOUBLE PRECISION` in the D version) assumed-size array which must be set by the user when `YESA=.TRUE.` to the values of the entries of the matrix **A** corresponding to the column indices set in `INDEX`. On exit, if `YESA=.TRUE.`, `A` contains the values of the entries of the matrix **A** corresponding to the column indices set in `INDEX`. This argument need not be set if `YESA=.FALSE.` and is in this case unchanged by the subroutine.

`IPR` is an `INTEGER` array of length `NR+1`, which must be set by the user so that `IPR(i)` points to the positions in the arrays `INDEX` and `A` of the first entry in row i ($i = 1, ..., $`NR`). `IPR(NR+1)` must be set to the number of entries in **A** plus one. This argument is unchanged by the subroutine.

`JPC` is an `INTEGER` array of length `NC+1`, which need not be set by the user on entry. On exit, it will be set so that `JPC(j)` points to the positions in the arrays `INDEX` and `A` of the first entry in column j ($j = 1, ..., $`NC`). `IPR(NC+1)` will be set to the number of entries in **A** plus one.

IW is an INTEGER array of length at least NC which is used for workspace and which need not be set by the user.

INFORM is an INTEGER variable which need not be set on input. On output, the value of INFORM is used to signal an error in the input data or to indicate a successful call to the subroutine. Possible values of INFORM and their consequences are as follows:

- 0 A successful call has been made to MC46A/AD and the matrix **A** is output in a column oriented fashion using the arrays INDEX, JPC and, optionally, A.
- –1 One of the parameters NR, NC or LA is too small.
- –2 The column index of one of the components of **A**, input in INDEX, is smaller than 1 or larger than NC.

2.2 Error returns

If some error in the input data is detected by the subroutines, they return immediately to the calling program with the error flag INFORM having a negative value. See §2.1 for details.

2.3 Converting from storage-by-columns to storage-by-rows

MC46 may be used to convert a matrix stored by columns to one stored by rows by applying the subroutine to the matrix $\mathbf{A}^T$.

3 GENERAL INFORMATION

Use of common: none.

Workspace: provided by the user through the argument IW (INTEGER array of length NC).

Other routines called directly: none.

Input/output: none.

Restrictions: NR ≥ 1, NC ≥ 1 and LA ≥ IPR(NC+1)-1.

4 METHOD

MC46A/AD is an in-place sort algorithm which handles each item to be sorted exactly twice, so its computational complexity is of the order of the number of nonzeros of **A**. The number of entries in each column is first obtained by a counting pass. The space needed by each column is allocated. Each entry in turn is made the *current entry* and examined to see if it is in place. If not, it is put into the next location allotted for the column it occurs in, and the entry displaced made the current entry. This chain of displacing entry continues until the first entry examined in the chain is located and stored. Then the next item is examined.

5 EXAMPLE OF USE

As a simple example, suppose we have the sparse matrix

$$\begin{pmatrix} 1.1 & 1.2 & & 1.4 & & & \\ & 2.2 & & 2.4 & 2.5 & & \\ & & 3.3 & & & 3.6 & \\ & & & 4.4 & & & \\ & 5.2 & & & 5.5 & & \\ 6.1 & & 6.3 & & & & 6.7 \end{pmatrix}$$

stored by rows but actually wish it to be stored by columns. Then we could use the following piece of code.

```
      PROGRAM MAIN
      INTEGER INFORM, J, K, K1, K2, LA, NR, NC, NCP1, NRP1
      PARAMETER ( NR = 6, NC = 7, LA = 14 )
      PARAMETER ( NCP1 = NC + 1, NRP1 = NR + 1 )
      INTEGER INDEX( LA ), JPC( NCP1 ), IPR( NRP1 ), IW( NC )
      DOUBLE PRECISION A( LA )
      LOGICAL YESA
      EXTERNAL MC46AD
      DATA IPR / 1, 4, 7, 9, 10, 12, 15 /
      DATA INDEX / 1, 2, 4, 2, 4, 5, 3, 6, 4, 2, 5, 1, 3, 7 /
      DATA A / 1.1D+0, 1.2D+0, 1.4D+0, 2.2D+0, 2.4D+0, 2.5D+0, 3.3D+0,
     *         3.6D+0, 4.4D+0, 5.2D+0, 5.5D+0, 6.1D+0, 6.3D+0, 6.7D+0 /
      YESA = .TRUE.
      CALL MC46AD( NR, NC, LA, YESA, INDEX, A, IPR, JPC, IW, INFORM )
      WRITE( 6, 2000 ) INFORM
      IF ( INFORM .NE. 0 ) STOP
      DO 20 J = 1, NC
         K1   = JPC( J )
         K2   = JPC( J + 1 ) - 1
         IF ( K1 .LE. K2 ) THEN
            WRITE( 6, 2010 ) J
            DO 10 K = K1, K2
               WRITE( 6, 2020 ) INDEX( K ), A( K )
   10       CONTINUE
         END IF
   20 CONTINUE
      STOP
 2000 FORMAT( ' INFORM = ', I2, ' ON EXIT FROM MC46A ', / )
 2010 FORMAT( ' COLUMN ', I2 )
 2020 FORMAT( '    ROW ', I2, ' VALUE ', F4.1 )
      END
```

This produces the following output:

```
 INFORM =  0 ON EXIT FROM MC46A

 COLUMN  1
    ROW  1 VALUE  1.1
    ROW  6 VALUE  6.1
 COLUMN  2
    ROW  1 VALUE  1.2
    ROW  2 VALUE  2.2
    ROW  5 VALUE  5.2
 COLUMN  3
    ROW  6 VALUE  6.3
    ROW  3 VALUE  3.3
 COLUMN  4
```

```
      ROW  1 VALUE  1.4
      ROW  2 VALUE  2.4
      ROW  4 VALUE  4.4
   COLUMN  5
      ROW  2 VALUE  2.5
      ROW  5 VALUE  5.5
   COLUMN  6
      ROW  3 VALUE  3.6
   COLUMN  7
      ROW  6 VALUE  6.7
```