## 1 SUMMARY

Given a **sparse matrix A** stored in a compact form and two vectors **x** and **y**, the routine evaluates either of the **matrix-vector products** $\mathbf{y}+\mathbf{Ax}$ or $\mathbf{y}+\mathbf{A}^T\mathbf{x}$.

**ATTRIBUTES** — **Version:** 1.0.0. **Types:** `MC09A`, `MC09AD`. **Original date:** February 1972. **Origin:** J. K. Reid, Harwell.

## 2 HOW TO USE THE PACKAGE

### 2.1 Argument list

*The single precision version*

```
CALL MC09A(M,N,A,X,Y,TRANS,IRN,IP)
```

*The double precision version*

```
CALL MC09AD(M,N,A,X,Y,TRANS,IRN,IP)
```

`M` is an `INTEGER` variable set by the user to $m$ the number of rows in the matrix. It is not altered by the subroutine.

`N` is an `INTEGER` variable set by the user to $n$ the number of columns in the matrix. It is not altered by the subroutine.

`A` is a `REAL` (`DOUBLE PRECISION` in the D version) array holding the nonzero matrix elements. These are stored by columns, e.g. $a_{11,}a_{13,}a_{19,}a_{21,}a_{22},...$ . It is not altered by the subroutine.

`X` is a `REAL` (`DOUBLE PRECISION` in the D version) array that must be set to contain the vector **x**. It is not altered by the subroutine.

`Y` is a `REAL` (`DOUBLE PRECISION` in the D version) array that must be set to contain the vector **y** and is overwritten by the result.

`TRANS` is a `LOGICAL` variable which should be set to `.TRUE.` if $\mathbf{y}+\mathbf{A}^T\mathbf{x}$ is required and to `.FALSE.` if $\mathbf{y}+\mathbf{Ax}$ is required. It is not altered by the subroutine.

`IRN,IP` are `INTEGER` arrays used to describe the sparsity structure of **A** and must be set by the user. The nonzeros are stored by columns and `IP(J)` must point to the first nonzero of the `J`-th column, unless this column is null in which case `IP(J)` must equal `IP(J+1)`; `IP(N+1)-1` must equal the number of nonzeros. `IRN(K)` must hold the row number of the `K`-th nonzero. `IP` has dimension $n$+1 and `IRN` has dimension equal to the number of nonzeros in **A**.

## 3 GENERAL INFORMATION

**Use of common:** None.

**Other routines called directly:** None.

**Input/output:** None.

**Workspace:** None.

## 5 EXAMPLE OF USE

The following code reads the entries of a sparse matrix (in any order) and the vectors **x** and **y**. The matrix is then sorted, MC09 is called, and the result is printed.

```
      REAL A(1000),X(100),Y(100)
      INTEGER IRN(1000),ICN(1000),IP(101)
C READ ORDER AND NUMBER OF NONZEROS
      READ(5,*) N,NZ
C CHECK THAT N AND NZ ARE WITHIN BOUNDS
      IF(N.LE.0.OR.N.GT.100) GO TO 40
      IF(NZ.LE.0.OR.NZ.GT.1000) GO TO 40
C READ MATRIX NONZEROS AND VECTORS
      READ(5,*) (IRN(I),ICN(I),A(I),I=1,NZ),(X(I),I=1,N),(Y(I),I=1,N)
C SORT THE MATRIX NONZEROS BY COLUMNS
      CALL MC20A(N,NZ,A,IRN,IP,ICN,0)
      IP(N+1)=NZ+1
C FORM PRODUCT
      CALL MC09A(N,N,A,X,Y,.FALSE.,IRN,IP)
C WRITE PRODUCT
      WRITE(6,10)(Y(I),I=1,N)
   10 FORMAT(' PRODUCT IS '/5F10.5)
   40 STOP
      END
```

For the data

$$\mathbf{A}=\begin{pmatrix}1&0&0&4\\0&6&0&8\\9&0&11&0\\0&14&0&16\end{pmatrix},\quad \mathbf{x}=\begin{pmatrix}1\\2\\3\\4\end{pmatrix},\quad \mathbf{y}=\begin{pmatrix}0\\0\\0\\0\end{pmatrix}.$$

we could have as input

```
        4       8
        4       4             16.
        1       1              1.
        4       2             14.
        2       2              6.
        3       1              9.
        2       4              8.
        3       3             11.
        1       4              4.
        1.      2.     3.      4.
        0.      0.     0.      0.
```

and we would get the following output

```
 PRODUCT IS
  17.00000  44.00000  42.00000  92.00000
```