1 SUMMARY

Given a sparse matrix **A**, this subroutine computes the sparse matrix $\mathbf{A}^T\mathbf{A}$.

Three forms of data storage are permitted for the input matrix: storage by columns, where the row indices and column pointers describe the matrix; storage by rows, where the column indices and row pointers describe the matrix; and the coordinate scheme, where both row and column indices describe the position of entries in the matrix.

ATTRIBUTES — **Version:** 1.0.0. (12 July 2004) **Types:** Real (single, double). **Calls:** MC59. **Original date:** April 2001. **Remark:** The MC26 entries are threadsafe versions of MC35. **Origin:** I.S.Duff, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Initialization

The MC26I/ID entry must be called prior to the first call to the MC26A/AD entry to initialize the control array.

The single precision version

```
CALL MC26I(ICNTL)
```

The double precision version

```
CALL MC26ID(ICNTL)
```

ICNTL is an INTEGER array of length 5, see Section 2.3.

2.2 Argument List

The single precision version

```
CALL MC26A(M,N,ITYPE,NZA,A,IND,LATA,IATA,ATA,JPTR,IPTR,
*                          NZATA,IW,ICNTL,INFO)
```

The double precision version

```
CALL MC26AD(M,N,ITYPE,NZA,A,IND,LATA,IATA,ATA,JPTR,IPTR,
*                          NZATA,IW,ICNTL,INFO)
```

M is an INTEGER variable which must be set by the user to the number of rows in the matrix **A**. This argument is not altered. **Restriction:** M > 0.

N is an INTEGER variable which must be set by the user to the number of columns in the matrix **A**. This argument is not altered. **Restriction:** N > 0.

ITYPE is an INTEGER variable which must be set by the user to the value corresponding to the type of input used. The permitted values are:

1 for the input matrix being stored by columns,
2 for the input matrix being stored by rows,
3 for the input matrix being stored by the coordinate scheme.

This argument is not altered.

NZA is an INTEGER variable which must be set by the user to the number of entries in matrix **A**. This argument is not altered.

A is a REAL (DOUBLE PRECISION in the D version) array of length NZA, which must be set by the user to the

values of the entries in matrix **A**. The order depends upon the value of ITYPE. For ITYPE = 1, they are ordered by columns, with the entries in each column contiguous and those of column J preceding those of column J+1 (J=1,...,N). The ordering within columns is unimportant. For ITYPE = 2, they are ordered by rows, with the entries in each row contiguous, and those of row I preceding those of row I+1 (I=1,...,M). The ordering within rows is unimportant. For ITYPE = 3 they can be in any order. In all cases, the array is set by the subroutine to hold the entries of **A** in row order.

IND is an INTEGER array of length NZA, which must be set by the user . For ITYPE = 1 and 3 it must be set to the row indices of the corresponding entries in A. For ITYPE = 2, the array must be set to the column indices of the corresponding entries in A. On exit, the array is set by the subroutine to the column indices of the corresponding entries in A.

LATA is an INTEGER variable which must be set by the user to the length of array ATA. This argument is not altered. **Restriction:** LATA ≥ NZA+N.

IATA is an INTEGER array of length LATA, which need only be set by the user if ITYPE = 3. If so, the first NZA values must be the column numbers of the entries of matrix **A**, in the same order as arrays A and IND. It will be set by the subroutine so that the first NZATA values are the row numbers of the entries in the lower triangle of $\mathbf{A}^T\mathbf{A}$, ordered by columns.

ATA is a REAL (DOUBLE PRECISION in the D version) array of length LATA, which need not be set by the user. It will be set by the subroutine to contain the values of the entries in the lower triangle of $\mathbf{A}^T\mathbf{A}$. It is stored by columns with the entries of each column contiguous, and the entries of column I preceding those of column I+1 (I=1,...,M).

JPTR is an INTEGER array of length N+1, which must be set by the user only if ITYPE = 1. If so, it must be set so that JPTR(J) points to the position in arrays A and IND of the first entry in column J (J=1,...,N). JPTR(N+1) must be set to NZA+1. On exit, it is set by the subroutine so that JPTR(J) points to the position in arrays ATA and IATA of the first entry in column J of the lower triangle of $\mathbf{A}^T\mathbf{A}$. (J=1,...,N). JPTR(N+1) will be set to 1 greater than the number of entries in $\mathbf{A}^T\mathbf{A}$.

IPTR is an INTEGER array of length M+1, which must be set by the user only if ITYPE = 2. If so, it must be set so that IPTR(I) points to the position in arrays A and IND of the first entry in row I (I=1,...,M). IPTR(M+1) must be set to NZA+1. On exit it is set by the subroutine so that IPTR(I) points to the position in arrays A and IND of the first entry in row I of **A** (I=1,...,M). IPTR(M+1) will be set to NZA+1.

NZATA is an INTEGER variable which need not be set by the user. It will be set by the subroutine to the actual number of entries in the lower triangle of $\mathbf{A}^T\mathbf{A}$, or the length necessary to perform the computation if error return 4 is invoked.

IW is an INTEGER array of length M+N, which is used as workspace.

ICNTL is an INTEGER array of length 5 whose elements can optionally be set by the user, see Section 2.3.

INFO is an INTEGER array of length 5 which is used to return information to the user, see Section 2.3.

2.3 The control and information arrays

The ICNTL array argument which must be of length 5 can be used to pass optional control values to the routine.

ICNTL(1) specifies the unit number to be used to output error messages, see INFO below. It has a default value of 6 which can be reset by the user to another unit or set negative if messages are to be suppressed.

ICNTL(2) specifies the unit number to be used to output warning messages, see INFO below. It has a default value of 6 which can be reset by the user to another unit or set negative if messages are to be suppressed.

ICNTL(3) specifies whether checks for out-of-range indices and duplicate entries are made when ITYPE = 3. If ICNTL(3) = 1 and ITYPE = 3, such checks are made. If ICNTL(3) = 0 (the default) or ITYPE ≠ 3, no such checks are made.

ICNTL(4) to ICNTL(5) should not be altered and at present are not used by MC26.

The INFO array argument which must be of length 5 is used to return information to the user.

INFO(1) is used as an error and warning indicator. It is set to a negative value when errors prevented the calculation from completing, to a value greater than zero when unusual conditions were present of which the user should be informed and a zero value when there are no errors or warnings. The possible values are listed:

- 0 – Input data correct. The calculation is performed as described.
- –1 – M ≤ 0. Immediate return with input values unchanged and INFO(2) set to M.
- –2 – N ≤ 0. Immediate return with input values unchanged and INFO(2) set to N.
- –3 – LATA < NZA+N. Immediate return with input values unchanged and INFO(2) set to NZA + N.
- –4 – LATA is less than the number of entries in $\mathbf{A}^T\mathbf{A}$, so arrays ATA and IATA are too small to hold the result. The smallest value for LATA that will work is given in NZATA. Subroutine returns with arrays A, IND, and IPTR computed to their output form, but ATA, IATA, and JPTR corrupted.
- –5 – ITYPE is out of range. Immediate return with input values unchanged and INFO(2) set to ITYPE.
- –6 – with ITYPE = 3 at least one of the row and column indices in IND and IATA is out of range. The total number of indices out of range is returned in INFO(2).
- –7 – MC59 has returned with an unexpected error and its error code can be found in INFO(2).

The following warnings are possible when calling with ITYPE = 3.

- +1 – One or more duplicates are present in the user-supplied matrix. All additional ones are ignored and execution continued.
- +2 – One or more row indices in IND are out of range. The entry is removed and execution continued.
- +4 – One or more column indices in IATA are out of range. The entry is removed and execution continued.

Return values of 3, 5, 6 and 7 are also possible indicating a combination of two or more of the conditions occurring, i.e. the individual warning values are summed.

INFO(2) is set to supplementary information when there is an error, see INFO(1).

INFO(3) is set to the number of duplicate entries.

INFO(4) is set to the number of out-of-range indices in IND.

INFO(5) is set to the number of out-of-range indices in IATA.

3 GENERAL INFORMATION

Workspace: array IW of length M+N is used as workspace.

Use of common: none.

Other routines called directly: MC26A/AD calls MC59A/AD and MC26B/BD, which never needs to be called directly by the user.

Input/output: error messages on unit ICNTL(1) (ICNTL(1)=0 suppresses them).

Restrictions: M > 0, N > 0, LATA ≥ NZA+N, 1 ≤ ITYPE ≤ 3.

4 METHOD

MC26A/AD first carries out the checks for errors in the input. Control is immediately returned to the calling subroutine if an error is found. The input is then permuted so that the input matrix is ordered by rows with the columns in order within each row. MC26B/BD is then called to affect the construction of $\mathbf{A}^T\mathbf{A}$.

MC26B/BD computes the lower triangle of $\mathbf{A}^T\mathbf{A}$ from $\mathbf{A}$ and $\mathbf{A}^T$ by forming each row of $\mathbf{A}^T\mathbf{A}$ as a linear combination of the appropriate rows in $\mathbf{A}$. We scan each row of $\mathbf{A}^T$ in turn to find which rows of $\mathbf{A}$ contribute to that same row of $\mathbf{A}^T\mathbf{A}$, the values for the rows being accumulated in ATA. We can avoid generating any entries in the lower triangular

part because the entries in $\mathbf{A}^T$ are ordered before entry to MC26B/BD. Finally, the output arrays are updated to the desired form, and a return is made to the calling program. If arrays IATA and ATA are found to be too short, then calculation of these arrays is stopped, the minimum required size is accumulated in NZATA, and an error return made.

5 EXAMPLE OF USE

The following program gives an example of the use of MC26. We input the matrix

$$\mathbf{A} = \begin{pmatrix} 1.0 & 4.0 \\ 0.0 & 3.0 \\ 5.0 & 0.0 \end{pmatrix}$$

in coordinate form in arrays

```
      A = 1.0, 3.0, 5.0, 4.0
    IND =  1 ,  2 ,  3 ,  1
   IATA =  1 ,  2 ,  1 ,  2
```

and receive as output the matrix,

$$\mathbf{A}^T\mathbf{A} = \begin{pmatrix} 26.0 & \\ 4.0 & 25.0 \end{pmatrix} .$$

Thus the program:

```
      DOUBLE PRECISION A(4),ATA(6)
      INTEGER JPTR(3),IPTR(4)
      INTEGER IND(4),IATA(6),IW(5),ICNTL(5),INFO(5)
      INTEGER M,N,NZA,LATA,NZATA,KK,ITYPE
      DATA A/1.0D0,3.0D0,5.0D0,4.0D0/
      DATA IND/1,2,3,1/
      DATA IATA/1,2,1,2,0,0/
      ITYPE=3
      M=3
      N=2
      NZA=4
      LATA=6
      CALL MC26ID(ICNTL)
      CALL MC26AD(M,N,ITYPE,NZA,A,IND,LATA,IATA,ATA,JPTR,IPTR,
     *            NZATA,IW,ICNTL,INFO)
      WRITE (6,100) (ATA(KK),KK=1,3),(IATA(KK),KK=1,3),JPTR,
     1              A,IND,IPTR
  100 FORMAT (' ATA is given by the arrays:'//
     1        '  ATA = ',3F5.1/' IATA = ',3(I3,2X)/
     2        ' JPTR = ',3(I3,2X)///
     3        ' with A (permuted) held in arrays:'//
     4        '    A = ',4F5.1/'  IND = ',4(I3,2X)/
     5        ' IPTR = ',4(I3,2X))
      STOP
      END
```

produces the following output:

```
 ATA is given by the arrays:

  ATA =  26.0  4.0 25.0
 IATA =    1    2    2
 JPTR =    1    3    4


 with A (permuted) held in arrays:

    A =   1.0  4.0  3.0  5.0
  IND =    1    2    2    1
 IPTR =    1    3    4    5
```