Warning: Subroutine MA28 has been superseded by subroutine MA48 which uses improved algorithms; the use of the latter routine is recommended. The superseded routine may be removed from later releases of the library.

1 SUMMARY

To **solve a sparse system of linear equations.** Given a sparse matrix $\mathbf{A}=\{a_{ij}\}_{n\times n}$ this subroutine solves $\mathbf{Ax}=\mathbf{b}$ (or optionally $\mathbf{A}^T\mathbf{x}=\mathbf{b}$). It will solve another system having the same sparsity pattern taking much less processing time. The matrix **A** is also allowed to be singular or rectangular. **Remark:** Short-integer version available.

The method is a variant of Gaussian elimination for sparse systems and is discussed by Duff and Reid, ACM Trans. Math. Software **5** (1979), 18-35.

ATTRIBUTES — **Version:** 1.0.0. **Types:** MA28A, MA28AD. **Calls:** MA30, MC20, MC22, MC23, MC24. **Remark:** supersedes MA18. **Original date:** April 1977 (modified to add MA28I/ID, February 1983). **Origin:** I. S. Duff, Harwell.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequences

There are four entries:

(a) MA28A/AD decomposes **A** into factors using a pivotal strategy designed to compromise between maintaining sparsity and controlling loss of accuracy through roundoff.

(b) MA28B/BD factorizes a new matrix **A** of the same pattern, using the pivotal sequence determined by an earlier entry to MA28A/AD. It also permits nonzeros in positions that filled in during the previous factorization.

(c) MA28C/CD uses the factors produced by MA28A/AD (or MA28B/BD) to solve **Ax**=**b** or $\mathbf{A}^T$**x**=**b**.

(d) MA28I/ID uses the factors produced by MA28A/AD (or MA28B/BD) to solve **Ax**=**b** or $\mathbf{A}^T$**x**=**b**. This entry differs from MA28C/CD inasmuch as iterative refinement is performed and an inaccurate factorization from MA28A/AD can be used. MA28I/ID provides an estimate of the error, but does not perform double-length accumulation of inner products because it is intended to be used with drop tolerances (see TOL in section 2.2) rather than to achieve full machine accuracy from ill-conditioned systems.

MA28B/BD is much faster than MA28A/AD. In some applications it is expected that there will be many calls to MA28B/BD for each call to MA28A/AD. Also, it is expected that MA28C/CD or MA28I/ID may be called with many different vectors for the same matrix **A**.

To use MA28I/ID, the user must save the input matrix. It is, however, possible to use TOL (see section 2.2) to reduce the storage for the matrix factors. MA28I/ID also provides an estimate of the error.

We first describe the argument list for MA28A/AD. Reference should be made to this description for information on parameters which are common to MA28A/AD and MA28B/BD or MA28C/CD.

To decompose a matrix

The single precision version

```
CALL MA28A(N,NZ,A,LICN,IRN,LIRN,ICN,U,IKEEP,IW,W,IFLAG)
```

The double precision version

```
CALL MA28AD(N,NZ,A,LICN,IRN,LIRN,ICN,U,IKEEP,IW,W,IFLAG)
```

`N` is an `INTEGER` variable which must be set by the user to the order n of the matrix **A**. It is not altered by the subroutine. **Restriction:** `N` ≥ 1.

`NZ` is an `INTEGER` variable which must be set by the user to the number of nonzeros in the matrix **A**. It is not altered by the subroutine. **Restriction:** `NZ` ≥ 1.

`A` is a `REAL` array (`DOUBLE PRECISION` in the D version) of length `LICN`. `A(k), k=1,NZ` must be set by the user to hold the nonzero entries of the matrix **A**. On exit, `A` holds the nonzero entries in the factors of the matrix **A**. It should be preserved by the user between calls to this routine and `MA28C/CD` or `MA28I/ID`.

`LICN` is an `INTEGER` variable which must be set by the user to the length of arrays `A` and `ICN`. Since the decomposition is returned in `A` and `ICN`, `LICN` should be large enough to accommodate this and should ordinarily be 2 to 4 times as large as `NZ` (see `MINICN` in section 2.2). It is not altered by the subroutine. **Restriction:** `LICN` $\geq$ `NZ`.

`IRN` is an `INTEGER` array of length `LIRN`. On entry to `MA28A/AD`, `IRN(k)` must hold the row index of the nonzero stored in `A(k), k=1,NZ`. It is used as workspace by `MA28A/AD`, is altered by `MA28A/AD`, and need not be preserved for any subsequent calls.

`LIRN` is an `INTEGER` variable which must be set by the user to the length of array `IRN`. `LIRN` need not be as large as `LICN`; normally it will not need to be very much greater than `NZ`. It is not altered by the subroutine. **Restriction:** `LIRN` $\geq$ `NZ`.

`ICN` is an `INTEGER` array of length `LICN`. On entry `ICN(k)` must hold the column index of the nonzero stored in `A(k), k=1,NZ`. On output, it holds the column indices of the factors of the matrix **A**. These entries should be unaltered by the user between a call to this subroutine and subsequent calls to `MA28B/BD`, `MA28C/CD` or `MA28I/ID`.

`U` is a `REAL` variable (`DOUBLE PRECISION` in the D version). On input to `MA28A/AD`, the user should set `U` to a value between zero and one to control the choice of pivots. A value of 0.1 has been found to work well on test examples. The subroutine will not fail if `U` is outside the above range; values of `U` less than zero are treated as zero and values of `U` greater than one are treated as one. It is unaltered by the subroutine.

`IKEEP` is an `INTEGER` array of length $5n$. It need never be referenced by the user and should be preserved between calls to this subroutine and `MA28B/BD`, `MA28C/CD` or `MA28I/ID`. If preordering to block triangular form is suppressed (see `LBLOCK` in section 2.2 and section 2.6), then only the first $4n+1$ locations in `IKEEP` are referenced.

`IW` is an `INTEGER` array of length $8n$. It is used as workspace by the subroutine. If `NSRCH` (see section 2.2) has been set to a value less than or equal to n, then `IW` need only be of length $7n$.

`W` is a `REAL` array (`DOUBLE PRECISION` in the D version) of length n. `W` is used as workspace and an estimate of the largest entry encountered during LU decomposition (see `GROW` in section 2.2) is output in `W(1)` and the largest entry in the input matrix is output in `W(2)` (see also `THEMAX` in section 2.2). If such an estimate is not requested, then `W` is not used at all by `MA28A/AD`. Note that it is possible to get an exact value for the growth by setting `LBIG` (see section 2.2) to `.TRUE.`.

`IFLAG` is an `INTEGER` variable. On exit from `MA28A/AD`, a value of zero indicates that the subroutine has performed successfully. For nonzero values, see section 2.3.

To decompose a matrix which has a similar structure to that previously decomposed by MA28A/AD

The single precision version

```
CALL MA28B(N,NZ,A,LICN,IVECT,JVECT,ICN,IKEEP,IW,W,IFLAG)
```

The double precision version

```
CALL MA28BD(N,NZ,A,LICN,IVECT,JVECT,ICN,IKEEP,IW,W,IFLAG)
```

The user must input the matrix in the same way as the original matrix was input to MA28A/AD. Note that if MA28A/AD was run with a positive value of TOL (see section 2.2) and entries were dropped from the data structure, a call to MA28B/BD will fail. The parameters are as follows:

N INTEGER variable equal to the order of the matrix. It is not altered by the subroutine. **Restriction:** $N \geq 1$.

NZ INTEGER variable equal to number of nonzeros in the matrix. It is not altered by the subroutine. **Restriction:** $NZ \geq 1$.

A REAL array (DOUBLE PRECISION in the D version) of length LICN. The user must set A(k), k=1,NZ to hold the nonzero entries of the matrix **A**. On exit, A holds the nonzero entries of the factors of the matrix **A**. It must be preserved by the user between calls to this subroutine and MA28C/CD or MA28I/ID.

LICN INTEGER variable equal to length of arrays A and ICN. It is not altered by the subroutine.

IVECT,JVECT INTEGER arrays of length NZ. IVECT(k) and JVECT(k) must contain respectively the row and column index of the nonzero stored in A(k), k=1,NZ. They are not altered by MA28B/BD.

The other parameters are as follows:

ICN,IKEEP are the INTEGER arrays (of lengths LICN, and 5*n*, respectively) of the same names as in the previous call to MA28A/AD. They should be unchanged since this earlier call and they are not altered by MA28B/BD. If preordering to block triangular form is suppressed (see LBLOCK in section 2.2 and section 2.6), then only the first 4*n*+1 locations in IKEEP are referenced.

IW is an INTEGER array of length 5*n* used as workspace by MA28B/BD.

W is a REAL array (DOUBLE PRECISION in the D version) of length *n*. It is used as workspace and an estimate of the largest entry encountered during LU decomposition (see GROW in section 2.2) is output in W(1) and the largest entry of the original matrix (see also THEMAX in section 2.2) will be output in W(2). If an exact value for the growth is required, then LBIG (see section 2.2) must be set to .TRUE..

IFLAG is an INTEGER variable which will be set to zero on successful exit from MA28B/BD, otherwise it will have a nonzero value (see section 2.3).

To solve equations Ax=b or A^Tx=b, using the factors of A from MA28A/AD or MA28B/BD without iterative refinement

The single precision version

```
CALL MA28C(N,A,LICN,ICN,IKEEP,RHS,W,MTYPE)
```

The double precision version

```
CALL MA28CD(N,A,LICN,ICN,IKEEP,RHS,W,MTYPE)
```

Information about the factors of **A** is communicated to this subroutine via the parameters N, A, LICN, ICN and IKEEP where:

N INTEGER variable equal to the order of the matrix. It is not altered by the subroutine.

A REAL array (DOUBLE PRECISION in the D version) of length LICN. It must be unchanged since the last call to MA28A/AD or MA28B/BD. It is not altered by the subroutine.

LICN is an INTEGER variable equal to the length of arrays A and ICN. It is not altered by the subroutine.

ICN,IKEEP are the INTEGER arrays (of lengths LICN and 5*n*, respectively) of the same names as in the previous call to MA28A/AD. They should be unchanged since this earlier call and they are not altered by MA28C/CD. If preordering to block triangular form is suppressed (see LBLOCK in section 2.2 and section 2.6), then only the first 4*n*+1 locations in IKEEP are referenced.

The other parameters are as follows:

RHS is a REAL array (DOUBLE PRECISION in the D version) of length *n*. The user must set RHS(*i*) to contain the value of the *i*-th component of the right hand side b_i, *i*=1,*n*. On exit, RHS(*i*) contains the *i*-th component of the solution vector x_i, *i*=1,*n*.

W is a REAL array (DOUBLE PRECISION in the D version) of length *n*. It is used as workspace by MA28C/CD.

MTYPE is an INTEGER variable which the user must set to determine whether MA28C/CD will solve **Ax**=**b** (MTYPE equal to 1) or $\mathbf{A}^T\mathbf{x}=\mathbf{b}$ (MTYPE≠1, zero say). It is not altered by MA28C/CD.

To solve equations Ax=b or $A^Tx=b$, using the factors of A from MA28A/AD or MA28B/BD with iterative refinement.

The single precision version

```
      CALL MA28I(N,NZ,AORG,IRNORG,ICNORG,LICN,A,ICN,IKEEP,
     * RHS,X,R,W,MTYPE,PREC,IFLAG)
```

The double precision version

```
      CALL MA28ID(N,NZ,AORG,IRNORG,ICNORG,LICN,A,ICN,IKEEP,
     * RHS,X,R,W,MTYPE,PREC,IFLAG)
```

N INTEGER variable equal to the order of the matrix. It is not altered by the subroutine.

NZ INTEGER variable equal to the number of entries in the original matrix. It is not altered by the subroutine.

For this entry the original matrix must have been saved in AORG, IRNORG, ICNORG where entry AORG(k) is in row IRNORG(k) and column ICNORG(k), k=1,...NZ.

AORG REAL array (DOUBLE PRECISION in the D version) of length NZ. Not altered by MA28I/ID.

IRNORG INTEGER array of length NZ. Not altered by MA28I/ID.

ICNORG INTEGER array of length NZ. Not altered by MA28I/ID.

Information about the factors of **A** is communicated to this subroutine via the parameters LICN, A, ICN and IKEEP where:

LICN is an INTEGER variable equal to the length of arrays A and ICN. It is not altered by the subroutine.

A REAL array (DOUBLE PRECISION in the D version) of length LICN. It must be unchanged since the last call to MA28A/AD or MA28B/BD. It is not altered by the subroutine.

ICN,IKEEP are the INTEGER arrays (of lengths LICN and 5*n*, respectively) of the same names as in the previous call to MA28A/AD. They should be unchanged since this earlier call and they are not altered by MA28I/ID. If preordering to block triangular form is suppressed (see LBLOCK in section 2.2 and section 2.6), then only the first 4*n*+1 locations in IKEEP are referenced.

The other parameters are as follows:

RHS is a REAL array (DOUBLE PRECISION in the D version) of length *n*. The user must set RHS(*i*) to contain the value of the *i*-th component of the right hand side b_i, *i*=1,*n*. It is not altered by MA28I/ID.

X is a REAL array (DOUBLE PRECISION in the D version) of length *n*. It need not be set unless an initial guess of the solution is given (see ISTART in section 2.2), in which case the user must set X(*i*) to contain the value of the

i-th component of the estimated solution x_i, *i*=1,*n*. On exit, X(*i*) contains the *i*-th component of the solution vector x_i, *i*=1,*n*.

R is a REAL array (DOUBLE PRECISION in the D version) of length *n*. It need not be set on entry. On exit, R(*i*) contains the *i*-th component of an estimate of the error if unless MAXIT (see section 2.2) has been set to zero.

W is a REAL array (DOUBLE PRECISION in the D version) of length *n*. It is used as workspace by MA28I/ID.

MTYPE is an INTEGER variable which the user must set to determine whether MA28I/ID will solve **Ax**=**b** (MTYPE equal to 1) or $\mathbf{A}^T\mathbf{x}=\mathbf{b}$ (MTYPE≠1, zero say). It is not altered by MA28I/ID.

PREC is a REAL variable (DOUBLE PRECISION in the D version). It should be set by the user to the relative accuracy required. The iterative refinement will terminate if the magnitude of the largest component of the estimated error relative to the largest component in the solution is less than PREC. It is not altered by MA28I/ID.

IFLAG is an INTEGER variable which will be set to zero on successful exit from MA28I/ID, otherwise it will have a nonzero value (see section 2.3).

2.2 Common blocks used

The single precision version

```
      COMMON/MA28E/LP,MP,LBLOCK,GROW
      COMMON/MA28F/EPS,RMIN,RESID,IRNCP,ICNCP,MINIRN,MINICN,IRANK,
     *  ABORT1,ABORT2
      COMMON/MA28G/IDISP(2)
      COMMON/MA28H/TOL,THEMAX,BIG,DXMAX,ERRMAX,DRES,CGCE,NDROP,
     *  MAXIT,NOITER,NSRCH,ISTART,LBIG
```

The double precision version

```
      COMMON/MA28ED/LP,MP,LBLOCK,GROW
      COMMON/MA28FD/EPS,RMIN,RESID,IRNCP,ICNCP,MINIRN,MINICN,IRANK,
     *  ABORT1,ABORT2
      COMMON/MA28GD/IDISP(2)
      COMMON/MA28HD/TOL,THEMAX,BIG,DXMAX,ERRMAX,DRES,CGCE,NDROP,
     *  MAXIT,NOITER,NSRCH,ISTART,LBIG
```

LP,MP are INTEGER variables used by the subroutine as the unit numbers for its warning and diagnostic messages. The default value for both is 6. The user can either reset them to a different stream number or suppress the output by setting them to zero. While LP directs the output of error diagnostics from the principal and internally called subroutines, MP controls only the output of a message which warns the user that the input has two or more nonzeros A(i), . . , A(k) with the same row and column indices. The action taken in this case is to proceed using the numerical value A(i) + . . . + A(k). In the absence of other errors, IFLAG will equal –14 on exit (see section 2.3).

LBLOCK is a LOGICAL variable which controls an option of first preordering the matrix to block lower triangular form (using Harwell subroutine MC23A/AD) (see section 2.6). The preordering is performed if LBLOCK is equal to its default value of .TRUE.. If LBLOCK is set to .FALSE., the option is not invoked and the space allocated to IKEEP can be reduced to 4*n*+1.

GROW is a LOGICAL variable. If it is left at its default value of .TRUE., then on return from MA28A/AD or MA28B/BD, W(1) will give an estimate (an upper bound) of the size of the largest entry encountered during the decomposition. If the matrix is well scaled (see section 2.7), then a high value for W(1), relative to the largest entry in the input matrix, indicates that the LU decomposition could be inaccurate and the user should be wary of the results and perhaps increase U for subsequent runs. We would like to emphasise that this value only relates to the accuracy of our LU decomposition and gives no indication as to the singularity of the matrix or the accuracy of the solution. This upper bound can be a significant overestimate particularly if the matrix is

badly scaled. If an accurate value for the growth is required, LBIG (q.v.) should be set to .TRUE..

EPS,RMIN are REAL variables (DOUBLE PRECISION in the D version) that monitor the stability of successive factorizations. EPS has the default value 10^{-4}. If the ratio of a pivot to the largest entry in its row is less than EPS, IFLAG is returned holding the index of the row. On return, RMIN holds the smaller of EPS and the smallest of these ratios. If EPS≥ 1.0, no such checks are made. If RMIN is very small and W(1) from MA28B/BD is also very large, it may be advisable to perform a new decomposition using MA28A/AD.

RESID is a REAL variable (DOUBLE PRECISION in the D version) which on exit from MA28C/CD gives the value of the maximum residual

$$\max_i |b_i - \sum_j a_{ij} x_j|$$

over all the equations unsatisfied because of dependency (zero pivots).

IRNCP,ICNCP are INTEGER variables which record the number of data compressions and thereby monitor the adequacy of 'elbow room' in IRN and A/ICN respectively. If either is quite large (say greater than $n/10$), it will probably pay to increase the size of the corresponding array for subsequent runs. If either is very low or zero then one can perhaps save storage by reducing the size of the corresponding array.

MINIRN,MINICN are INTEGER variables which, in the event of a successful return (IFLAG ≥ 0 or IFLAG = –14) give the minimum size of IRN and A/ICN respectively which would enable a successful run on an identical matrix. On an exit with IFLAG equal to –5, MINICN gives the minimum value of ICN for success on subsequent runs on an identical matrix. In the event of failure with IFLAG= –6, –4, –3, –2, or –1, then MINICN and MINIRN give the minimum value of LICN and LIRN respectively which would be required for a successful decomposition up to the point at which the failure occurred.

IRANK is an INTEGER variable which gives an estimate (actually an upper bound) of the rank of the matrix.

ABORT1 is a LOGICAL variable with default value .TRUE.. If ABORT1 is set to .FALSE. then MA28A/AD will decompose structurally singular matrices (including rectangular ones).

ABORT2 is a LOGICAL variable with default value .TRUE.. If ABORT2 is set to .FALSE. then MA28A/AD will decompose numerically singular matrices.

IDISP is an INTEGER array of length 2. On output from MA28A/AD, the indices of the diagonal blocks of the factors lie in positions IDISP(1) to IDISP(2) of A/ICN. This array must be preserved between a call to MA28A/AD and subsequent calls to MA28B/BD, MA28C/CD or MA28I/ID.

TOL is a REAL variable (DOUBLE PRECISION in the D version). If it is set to a positive value, then MA28A/AD will drop from the factors any nonzero whose modulus is less than TOL. The factorization will then require less storage but will be inaccurate. After a run of MA28A/AD where entries have been dropped, MA28B/BD will fail and the user is recommended to use MA28I/ID to obtain the solution. The default value for TOL is 0.0.

THEMAX is a REAL variable (DOUBLE PRECISION in the D version). On exit from MA28A/AD, it will hold the largest entry of the original matrix.

BIG is a REAL variable (DOUBLE PRECISION in the D version). If LBIG has been set to .TRUE., BIG will be set to the largest entry encountered during the factorization by MA28A/AD or MA28B/BD.

DXMAX is a REAL variable (DOUBLE PRECISION in the D version). On exit from MA28I/ID, DXMAX will be set to the largest component of the solution.

ERRMAX is a REAL variable (DOUBLE PRECISION in the D version). On exit from MA28I/ID, ERRMAX will be set to the largest component in the estimate of the error if MAXIT is positive and to zero otherwise.

DRES is a REAL variable (DOUBLE PRECISION in the D version). On exit from MA28I/ID, DRES will be set to the largest component of the residual if MAXIT is positive and to zero otherwise.

CGCE is a REAL variable (DOUBLE PRECISION in the D version) which is used to monitor the convergence of the

iterative refinement. If successive corrections do not decrease by a factor of at least CGCE, convergence is deemed to be too slow and MA28I/ID terminates with IFLAG set to –16. CGCE, which has default value 0.5 should be set to a value between zero and one.

NDROP is an INTEGER variable. If TOL has been set positive, NDROP will be set by MA28A/AD to the number of entries dropped from the data structure.

MAXIT is an INTEGER variable. It is the maximum number of iterations performed by MA28I/ID. It has a default value of 16.

NOITER is an INTEGER variable. It is set by MA28I/ID to the number of iterative refinement iterations actually used.

NSRCH is an INTEGER variable with default value 32768. If NSRCH $\leq n$, the pivot search is limited to NSRCH rows. This may result in different fill-in and execution time for MA28A/AD. If NSRCH $\leq n$, the workspace array IW (see section 2.1) can be reduced in length.

ISTART is an INTEGER variable. If ISTART is set to a value other than zero, then the user must supply an estimate of the solution to MA28I/ID (see X in section 2.1). The default value for ISTART is zero.

LBIG is a LOGICAL variable. If LBIG is set to .TRUE., the value of the largest entry encountered in the factorization by MA28A/AD or MA28B/BD is returned in BIG. Setting LBIG to .TRUE. will increase the time for MA28A/AD marginally and that for MA28B/BD by about 20 percent. The default value for LBIG is .FALSE..

2.3 Error diagnostics

A successful return from MA28A/AD, MA28B/BD or MA28I/ID is indicated by a value of IFLAG equal to zero. There are no error returns from MA28C/CD. Possible nonzero values for IFLAG are given below.

–17 Convergence of iterative refinement is deemed too slow (see CGCE in section 2.2). (MA28I/ID entry only).

–16 More than MAXIT iterations (see section 2.2) required for convergence of iterative refinement. (MA28I/ID entry only).

–15 A call to MA28B/BD follows a call to MA28A/AD in which entries were dropped. (MA28B/BD entry only).

Values –14 to –8: Error in user's input matrix. The nature is specified in an output message.

–14 More than one nonzero in same position in matrix. Action taken is to proceed with value equal to the sum of the multiple entries. (See common block variable MP in section 2.2.) (MA28A/AD and MA28B/BD entries.)

–13 Nonzero was not present during the previous call to MA28A/AD (MA28B/BD entry only).

–12 Row or column index out-of-range (MA28A/AD and MA28B/BD entries).

–11 $1 \leq$ N ≤ 32767 violated (MA28A/AD and MA28B/BD entries).

–10 NZ ≤ 0 (MA28A/AD and MA28B/BD entries).

–9 LICN $<$ NZ (MA28A/AD and MA28B/BD entries).

–8 LIRN $<$ NZ (MA28A/AD entry only).

–7 Error encountered during block triangularization phase (MA28A/AD entry only).

Values –6 to –3: Storage allocation for decomposition is insufficient (see common block variables MINICN and MINIRN, section 2.2) (MA28A/AD entry only).

–6 LIRN and LICN too small....information available from MINICN (see section 2.2).

–5 LICN too small....increase to at least value given by common block variable MINICN (see section 2.2).

–4 LICN far too small. No useful information in MINICN.

–3 LIRN too small.

–2 Matrix numerically singular. (MA28A/AD and MA28B/BD entries).

–1 Matrix structurally singular. This means that the nonzero pattern is such that the matrix will be singular for all possible numerical values of the nonzeros (MA28A/AD only).

+1 Successful decomposition on a structurally singular matrix (MA28A/AD only).

+2 Successful decomposition on a numerically singular matrix (MA28A/AD only).

+i (*i*=1,2,..,*n*) Warning. Very small pivot in row *i* (MA28B/BD only).

2.4 Parameter usage summary

MA28A/AD

Input	N, NZ, A(LICN), LICN, IRN(LIRN), LIRN, ICN(LICN), U
Unchanged by MA28A/AD	N, NZ, LICN, LIRN, U
Output	A, ICN, IKEEP(5*N), W(1) and W(2)† , IFLAG
Work-arrays	IW(8*N), W(N)†.

MA28B/BD

Input (by user)	NZ, A(LICN), IVECT(NZ), JVECT(NZ)
Input (from MA28A/AD)	N, ICN(LICN), LICN, IKEEP(5*N)
Unchanged by MA28B/BD	N, NZ, ICN, LICN, IVECT, JVECT, IKEEP
Output	A, IFLAG, W(1) and W(2)
Work-arrays	IW(5*N), W(N).

MA28C/CD

Input (by user)	RHS(N), MTYPE
Input (from MA28A/AD)	N, ICN(LICN), LICN, IKEEP(5*N)
Input (from MA28A/AD or MA28B/BD)	A(LICN)
Unchanged by MA28C/CD	N, ICN, LICN, IKEEP, A, MTYPE
Output	RHS
Work-array	W(N)

MA28I/ID

Input (by user)	NZ, AORG(NZ), IRNORG(NZ), ICNORG(NZ), RHS(N), X(N)†, MTYPE, PREC
Input (from MA28A/AD)	N, ICN(LICN), LICN, IKEEP(5*N)
Input (from MA28A/AD or MA28B/BD)	A(LICN)
Unchanged by MA28I/ID	N, NZ, AORG, IRNORG, ICNORG, ICN, LICN, IKEEP, A, RHS, MTYPE, PREC
Output	X, R(N), IFLAG
Work-array	W(N)

† Optional ..... see sections 2.1 and 2.2.

2.5 Singular or rectangular matrices

Singular and rectangular matrices can be handled by resetting common block variables ABORT1 and/or ABORT2 (see section 2.2). Additionally, the user must set N to the largest dimension of the system. RESID (see section 2.2) will give useful information on the consistency of the equations.

2.6 Block lower triangular form

Many large unsymmetric matrices can be permuted to the form

$$\mathbf{PAQ} = \begin{pmatrix} \mathbf{A}_{11} & & & & & \\ . & \mathbf{A}_{22} & & & & \\ . & . & \mathbf{A}_{33} & & & \\ . & . & . & . & & \\ . & . & . & . & . & \\ \mathbf{A}_{k1} & . & . & . & . & \mathbf{A}_{kk} \end{pmatrix}$$

whereupon the system

$\mathbf{Ax}=\mathbf{b}$ (or $\mathbf{A}^T\mathbf{x}=\mathbf{b}$)

can be solved by block forward-substitution (or back-substitution) giving a saving in storage and execution time if the matrices $\mathbf{A}_{ii}$ are much smaller than $\mathbf{A}$.

This facility is invoked as a default and information on the preordering to this form is given by the variables in the common block MC23B/BD:

The single precision version

```
COMMON/MC23B/LP,NUMNZ,NUM,LARGE,ABORT
```

The double precision version

```
COMMON/MC23BD/LP,NUMNZ,NUM,LARGE,ABORT
```

where LP is the stream for error messages and the INTEGER variables NUMNZ, NUM, LARGE give the structural rank, number of diagonal blocks (*k*, above), and the order of the largest block respectively. ABORT is a logical variable set by MA28A/AD to the value of ABORT1 (section 2.2).

If the user wishes to suppress this block triangularization option, the common block variable LBLOCK (see section 2.2) should be set to .FALSE.. The length of IKEEP can then be reduced to $4n+1$.

2.7 Badly-scaled systems

If the user's input matrix has entries differing widely in magnitude, then an inaccurate solution may be obtained and the estimate of the increase in entry size given by MA28A/AD or MA28B/BD will not be very meaningful. In such cases, the user is advised to first use MC19A/AD to obtain scaling factors for the matrix and then explicitly scale it prior to calling MA28A/AD. Thereafter, both left and right hand sides should be scaled as indicated in the code following the example in section 5.2. Additionally, a better indication of the accuracy of the factorization can be obtained from BIG by setting LBIG to .TRUE. (see section 2.2).

3 GENERAL INFORMATION

Use of common: the subroutine uses common blocks MA28E/ED, MA28F/FD, MA28G/GD, MA28H/HD and MC23B/BD, see sections 2.2 and 2.6. The MA28 common blocks are initialized by block data subprogram MA28J/JD.

Workspace:

W of length N (all entries, but optional in MA28A/AD entry, see section 2.1)

IW of length 8*n* in MA28A/AD and length 5*n* in MA28B/BD.

Other routines called directly: MA28D/DD, MA28J/JD (block data), MC20A/AD, MC22A/AD, MC23A/AD, MC24A/AD, MA30A/AD.

Input/output: Errors and warning messages only. Error messages on unit LP, warning messages on unit MP. Both have default value 6, and output is suppressed if they are set to zero.

Restrictions:

N ≥ 1,
LICN ≥ NZ,
LIRN ≥ NZ.

4 METHOD

These subroutines are really only data management routines. A full description of the package is given by Duff (Harwell Report R.8730, 1977). The method used is a sparse variant of Gaussian elimination. Subroutine MA28I/ID was added to the original package after consultations with Schaumburg, Wasniewski, and Zlatev from Copenhagen.

5 EXAMPLE OF USE

5.1 An example to solve sparse equations without iterative refinement

In the example code shown below, we first decompose a matrix and use information from this decomposition to solve a set of linear equations. Then we factorize a matrix of a similar sparsity pattern and solve another set of equations without iterative refinement in either case.

```
      DOUBLE PRECISION A(500), W(50), X(50), U
      INTEGER ICN(500), IRN(300), IVECT(250),
     +        IW(250), JVECT(250),IKEEP(250)
      LICN=500
      LIRN=300
      MTYPE=1
C     READ IN INPUT MATRIX.
      READ(5, * ) N,NZ
      IF (N.GT.50 .OR. NZ.GT.250) THEN
         WRITE(6,550)
         STOP
      END IF
      READ(5, * ) (IRN(I), ICN(I), A(I), I=1,NZ)
C     COPY INDEX INFORMATION FOR USE IN SUBSEQUENT MA28BD CALL.
      DO 300 I=1,NZ
        IVECT(I)=IRN(I)
 300    JVECT(I)=ICN(I)
      U=0.10D0
C     DECOMPOSE MATRIX INTO ITS FACTORS.
      CALL MA28AD(N,NZ,A,LICN,IRN,LIRN,ICN,U,IKEEP,IW,W,IFLAG)
      IF (IFLAG.LT.0) THEN
```

```
         WRITE(6,600) IFLAG
         STOP
      END IF
C     READ IN RIGHT HAND SIDE.
      READ(5, * ) (X(I),I=1,N)
C     SOLVE LINEAR SYSTEM.
      CALL MA28CD(N,A,LICN,ICN,IKEEP,X,W,MTYPE)
C     PRINT OUT SOLUTION VECTOR.
      WRITE(6,450)
      WRITE(6,400) (X(I),I=1,N)
C     READ IN, DECOMPOSE, AND SOLVE SYSTEM OF SIMILAR EQUATIONS.
      READ(5, * ) (A(I),I=1,NZ)
      READ(5, * ) (X(I),I=1,N)
      CALL MA28BD(N,NZ,A,LICN,IVECT,JVECT,ICN,IKEEP,IW,W,IFLAG)
      IF (IFLAG.LT.0) THEN
         WRITE(6,650) IFLAG
         STOP
      END IF
      CALL MA28CD(N,A,LICN,ICN,IKEEP,X,W,MTYPE)
      WRITE(6,450)
      WRITE(6,400) (X(I),I=1,N)
 400  FORMAT( 5D13.5 )
 450  FORMAT( / ' THE SOLUTION VECTOR X() IS:'/)
 550  FORMAT( ' ERROR IN INPUT DATA FOR N AND/OR NZ' )
 600  FORMAT( ' ERROR RETURN FROM MA28AD WITH IFLAG =',I2 )
 650  FORMAT( ' ERROR RETURN FROM MA28BD WITH IFLAG =',I2 )
      STOP
      END
```

Thus if, in this example, we wish to solve:

$$\begin{pmatrix} 3.14 & 7.5 & \\ 4.1 & 3.2 & 0.3 \\ & 1.0 & 4.1 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 1.0 \\ 2.0 \\ 3.0 \end{pmatrix}$$

followed by the system:

$$\begin{pmatrix} 4.7 & 6.2 & \\ 3.2 & 0.0 & 0.31 \\ & 3.1 & 0.0 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 1.1 \\ 2.1 \\ 3.1 \end{pmatrix}$$

we have as input

```
3     7
1     1     3.14
2     3     0.30
3     3     4.1
2     1     4.1
1     2     7.5
3     2     1.0
2     2     3.2
1.0       2.0          3.0
4.7       0.31         0.0          3.2          6.2
3.1       0.0
1.1       2.1          3.1
```

and output would be

```
THE SOLUTION VECTOR X() IS:
 0.48858D+00 -0.71219D-01  0.74908D+00
THE SOLUTION VECTOR X() IS:
-0.10851D+01  0.10000D+01  0.17975D+02
```

5.2 Scaling

In the example code shown below we scale the matrix and right-hand vector (see section 2.7) prior to decomposition and solution of the linear equations.

```
      DOUBLE PRECISION A(500),W(50),X(50),U
C
C     NOTE THE ARRAYS R() AND C(), WHICH HOLD THE SCALING FACTORS,
C     ARE OF TYPE REAL FOR BOTH SINGLE AND DOUBLE PRECISION PROGRAMS
C
      REAL R(50), C(50)
      INTEGER ICN(500), IRN(300), IW(250), IKEEP(250)
      LICN=500
      LIRN=300
      MTYPE=1
C     READ IN INPUT MATRIX.
      READ(5, * ) N,NZ
      IF (N.GT.50 .OR. NZ.GT.250) THEN
         WRITE(6,550)
         STOP
      END IF
      READ(5, * ) (IRN(I), ICN(I), A(I), I=1,NZ)
      U=0.10D0
C     SCALE INPUT MATRIX

      CALL MC19AD(N,NZ,A,IRN,ICN,R,C,A(NZ+1))
      DO 340 I=1,N
        R(I)=EXP(R(I))
        C(I)=EXP(C(I))
 340  CONTINUE
      DO 350 II=1,NZ
        I=IRN(II)
        J=ICN(II)
        A(II)=A(II)*R(I)*C(J)
 350  CONTINUE

C     DECOMPOSE MATRIX INTO ITS FACTORS.
      CALL MA28AD(N,NZ,A,LICN,IRN,LIRN,ICN,U,IKEEP,IW,W,IFLAG)
      IF (IFLAG.LT.0) THEN
         WRITE(6,600) IFLAG
         STOP
      END IF
C     READ IN RIGHT HAND SIDE.
      READ(5, * ) (X(I), I=1,N)
C     SCALE THE RIGHT HAND SIDE VECTOR BY ROW WEIGHTS

      DO 425 I=1,N
        X(I)=X(I)*R(I)
 425  CONTINUE

C     SOLVE LINEAR SYSTEM.
      CALL MA28CD(N,A,LICN,ICN,IKEEP,X,W,MTYPE)
C     SCALE THE RIGHT HAND SIDE VECTOR BY COLUMN WEIGHTS

      DO 475 I=1,N
        X(I)=X(I)*C(I)
 475  CONTINUE

C     PRINT OUT SOLUTION VECTOR.
      WRITE(6,450)
```

```
      WRITE(6,400) (X(I),I=1,N)
  400 FORMAT( 5D13.5 )
  450 FORMAT( / ' THE SOLUTION VECTOR X() IS:' / )
  550 FORMAT( ' ERROR IN INPUT DATA FOR N AND/OR NZ' )
  600 FORMAT( ' ERROR RETURN FROM MA28AD WITH IFLAG =',I2 )
      STOP
      END
```

Thus if, in this example we wish to solve:

$$\begin{pmatrix} 3.14E5 & 7.5E1 & \\ 4.1 & 3.2E{-}3 & 0.3 \\ & 1.0 & 4.1E2 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 3.1415E5 \\ 5.0064E0 \\ 1.232E3 \end{pmatrix}$$

we have as input

```
      3     7
      1     1       3.14000D+05
      2     3       0.30
      3     3       4.10000D+02
      2     1       4.1
      1     2       7.50000D+01
      3     2       1.00000D+00
      2     2       3.20000D-03
   0.31415D+06  0.50064D+01  0.12320D+04
```

and output would be

```
  THE SOLUTION VECTOR X() IS:

   0.10000D+01  0.20000D+01  0.30000D+01
```

5.3 An example to solve sparse equations with iterative refinement

In the example code shown below, we first decompose a matrix, allowing MA28AD to drop entries from the matrix, and then use this decomposition to solve a set of linear equations using iterative refinement.

```
      DOUBLE PRECISION A(500), W(50), X(50), U, TOL, AORG(300), THEMAX,
     *                 BIG, DXMAX, ERRMAX, DRES, CGCE, RHS(50), R(50),
     *                 PREC
      INTEGER IW(250)
      LOGICAL LBIG
      INTEGER ICN(500), IRN(300), IRNORG(250), ICNORG(250), IKEEP(250)

      COMMON /MA28HD/TOL,THEMAX,BIG,DXMAX,ERRMAX,DRES,CGCE,
     *              NDROP,MAXIT,NOITER,NSRCH,ISTART,LBIG

      LICN=500
      LIRN=300
C     READ IN INPUT MATRIX.
      READ(5, * ) N,NZ
      IF (N.GT.50 .OR. NZ.GT.250) THEN
         WRITE(6,550)
         STOP
      END IF
      READ(5, * ) (IRN(I),ICN(I),A(I),I=1,NZ)
C     COPY INDEX INFORMATION FOR USE IN SUBSEQUENT MA28BD CALL.
      DO 300 I=1,NZ
        AORG(I)=A(I)
        IRNORG(I)=IRN(I)
```

```
        ICNORG(I)=ICN(I)
 300  CONTINUE
      MTYPE=1
      U=0.5D0
      TOL=0.5D-1
      PREC=1.0D-12
C     DECOMPOSE MATRIX INTO ITS FACTORS.
      CALL MA28AD(N,NZ,A,LICN,IRN,LIRN,ICN,U,IKEEP,IW,W,IFLAG)
      IF (IFLAG.LT.0) THEN
         WRITE(6,600) IFLAG
         STOP
      END IF
      WRITE(6,700) NDROP
C     READ IN RIGHT HAND SIDE.
      READ(5, * ) (RHS(I),I=1,N)
C     SOLVE LINEAR SYSTEM USING ITERATIVE REFINMENT
      CALL MA28ID(N,NZ,AORG,IRNORG,ICNORG,LICN,A,ICN,IKEEP,
     *           RHS,X,R,W,MTYPE,PREC,IFLAG)
      IF (IFLAG.LT.0) THEN
         WRITE(6,650) IFLAG
         STOP
      END IF
      WRITE(6,750) NOITER
C     PRINT OUT SOLUTION VECTOR, ESTIMATE OF ERROR AND RESIDUAL.
      WRITE(6,450)
      WRITE(6,400) (X(I),I=1,N)
      WRITE(6,800) ERRMAX,DRES
 400  FORMAT( 5D13.5 )
 450  FORMAT( / ' THE SOLUTION VECTOR X() IS:'/)
 550  FORMAT( ' ERROR IN INPUT DATA FOR N AND/OR NZ' )
 600  FORMAT( ' ERROR RETURN FROM MA28AD WITH IFLAG =',I2 )
 650  FORMAT( ' ERROR RETURN FROM MA28ID WITH IFLAG =',I2 )
 700  FORMAT( / ' NUMBER OF ENTRIES DROPPED DURING FACTORISATION ',I3 )
 750  FORMAT( / ' NUMBER OF ITERATION STEPS ',I3 )
 800  FORMAT( / ' VALUES OF ERRMAX AND DRES ARE ', 2D13.5 )
      STOP
      END
```

Thus if, in this example, we wish to solve:

$$\begin{pmatrix} 5.00 & & 2.00 \\ 5.00 & 1.00 & 2.01 \\ & 1.00 & 3.00 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 11.00 \\ 13.03 \\ 11.00 \end{pmatrix}$$

we have as input

```
      3     7
      1     1     5.00
      2     1     5.00
      3     3     3.00
      2     2     1.00
      1     3     2.00
      3     2     1.00
      2     3     2.01
    11.00    13.03        11.00
```

and output would be

```
NUMBER OF ENTRIES DROPPED DURING FACTORISATION   1

NUMBER OF ITERATION STEPS   6

THE SOLUTION VECTOR X() IS:

 0.10000D+01  0.20000D+01  0.30000D+01

VALUES OF ERRMAX AND DRES ARE   0.12212D-13  0.11546D-13
```