1 SUMMARY

This subroutine **solves a system of linear equations whose matrix is banded, symmetric and positive definite.** It uses symmetric triangular decomposition without interchanges and takes advantage of any variation in bandwidth. For very large systems, it uses scratch space on disk, but if given sufficient workspace in core it makes no use of the disk. The secondary entry MA36B/BD allows further systems with the same matrix but different right-hand sides to be solved economically.

ATTRIBUTES — Version: 1.0.0. **Remark:** Supersedesd by MA36. **Types:** MA36A, MA36AD. **Original date:** October 1987. **Origin:** J.K.Reid, Harwell.

2 HOW TO USE THE PACKAGE

2.1 The argument list and calling sequence (primary call)

The primary call decomposes a given matrix and solves a corresponding set of equations.

The single precision version

```
CALL MA36A(N,A,KA,B,M,IERR,ROW,ND,LRECL,IP)
```

The double precision version

```
CALL MA36AD(N,A,KA,B,M,IERR,ROW,ND,LRECL,IP)
```

N is an INTEGER variable that must be set by the user to the order n of the matrix. This argument is not altered by the subroutine. **Restriction:** $N > 0$.

A is a REAL (DOUBLE PRECISION in the D version) array of length KA that need not be set by the the user and is used to hold the matrix decomposition. It must be unaltered between a call to MA36A/AD and a subsequent call to MA36B/BD unless the disk is used to hold the decomposition (on exit, IP(1) is negative if the disk has been used). See §2.3 for advice on the length needed.

KA is an INTEGER variable that the user must set to the length of array A. It is not altered by the subroutine.

B is a REAL (DOUBLE PRECISION in the D version) array of length N that the user must set to the right-hand-side vector. It is overwritten by the solution.

M is an INTEGER variable that the user must set to a bound on the number of elements in a row of the matrix between the diagonal and the last nonzero in the row (including the diagonal and any embedded zeros). It is not altered by the subroutine. Setting the value too large (for example, n) will waste storage in A and may lead to more i/o when disk storage is in use, but is not otherwise inefficient. **Restriction:** $1 \leq M \leq N$.

IERR is an INTEGER variable that the user need not set. The subroutine sets it to zero after a successful call and to an error number if it fails (see §2.4 for details).

ROW is a subroutine that the user must write. It must have the form

```
SUBROUTINE ROW(R,M,I,L)
```

R is a REAL (DOUBLE PRECISION in the D version) array of length M into which ROW must place the elements of row I of the matrix, starting at the diagonal and ending at the last nonzero of the row. Any embedded zeros must be set explicitly.

M is an INTEGER variable that specifies the maximum number of elements in a row (as passed to MA36A/AD). It must not be altered by ROW.

`I` is an `INTEGER` variable that specifies which row is wanted. It must not be altered by `ROW`.

`L` is an `INTEGER` variable that `ROW` must set to the number of elements in the row between the diagonal and the last nonzero (including the diagonal and any embedded zeros).

`ND` is an `INTEGER` variable that the user must set to the unit number of a direct-access file to be used to hold the matrix factors if there is insufficient room in array `A`, or to zero if no such file is ever to be used. See §2.3 for advice on the size of the file and its records. `ND` is not altered by the subroutine. **Restriction:** `ND` ≥ 0.

`LRECL` is an `INTEGER` variable that the user must set to the length of the records (in reals for `MA36A` and in double precision variables for `MA36AD`) on the direct-access file `ND`. This argument is not altered by the subroutine. **Restriction:** `LRECL` $\geq$ `M`.

`IP` is an `INTEGER` array of length `N` that need not be set by the user and is used to hold pointers to the ends of the rows. It must be unaltered between a call to `MA36A/AD` and a subsequent call to `MA36B/BD`.

2.2 The argument list and calling sequence (secondary call)

The secondary call (`MA36B/BD`) solves a further set of equations involving the same matrix but a different right-hand side vector.

The single precision version

```
CALL MA36B(N,A,KA,B,M,IERR,ND,LRECL,IP)
```

The double precision version

```
CALL MA36BD(N,A,KA,B,M,IERR,ND,LRECL,IP)
```

`N` is an `INTEGER` variable that must be as set in the previous call to `MA36A/AD` and holds the order n of the matrix. This argument is not altered by the subroutine.

`A` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `KA`. If the direct-access file has not been used then `A` must be as left by `MA36A/AD` and is not altered. If the file has been used then `A` is used for workspace. `NREC` (see §2.5) indicates whether the disk file has been needed (`NREC` > 0) or not (`NREC` $= 0$).

`KA` is an `INTEGER` variable that must be as set in the previous call to `MA36A/AD` and holds the length of array `A`. It is not altered by the subroutine.

`B` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `N` that the user must set to the right-hand vector. It is overwritten by the solution.

`M` is an `INTEGER` variable that must be as set in the previous call to `MA36A/AD`. It is not altered by the subroutine.

`IERR` is an `INTEGER` variable that must be as left by `MA36A/AD`. If it is not zero on entry, an error message is printed and a return is then executed. It is not altered by the subroutine unless there is an error when reading the disk file, in which case it is set to 8.

`ND` is an `INTEGER` variable that must be as set in the previous call to `MA36A/AD` and holds the unit number of the disk file or zero. `ND` is not altered by the subroutine.

`LRECL` is an `INTEGER` variable that must be as set in the previous call to `MA36A/AD` and holds the length of the disk records. `LRECL` is not altered by the subroutine.

`IP` is an `INTEGER` array of length `N` which must be as left on return from `MA36A/AD`. It is not altered by the subroutine.

2.3 The sizes of array A and file ND

If array `A` is large enough to hold all the rows of the factor $\mathbf{L}^T$ (row i starts at the diagonal and ends at the last column for which $\mathbf{A}$ has a nonzero in rows 1, 2,..., i) then the direct-access file is not used even if `ND` has been given a positive value.

If array A is not large enough for this and ND > 0 then file ND is used. In this case, array A must be large enough to hold any M consecutive matrix rows together with LRECL reals (or double precision variables in MA36AD/BD). The minimum permissible length for A is set in MINKA, see §2.5. Rows are not split between records, so the record length must be at least M. We recommend that the record length be at least long enough for ten average rows. The file length must be long enough to hold all the rows, bearing in mind that there is some wastage (less than M entries) at the end of each record.

2.4 Error conditions

A successful entry to MA36A/AD is indicated by IERR having the value zero. Unsuccessful entries are indicated by the printing of a warning message on output stream LP (see §2.5) and IERR being set to one of the following values.

1 The array A is too small. MINKA (see §2.5) holds the minimum size of array A for a guaranteed successful entry.

2 Subroutine ROW returned L less than 1 or greater than N-I+1.

3 A zero pivot has been found.

4 M has been given too small or too large a value.

5 LRECL has been given too small a value.

6 ND has been set to zero, but a direct access file is needed.

7 Error when writing to file ND.

8 Error when reading file ND.

< 0 Row I=-IERR of the matrix has a negative pivot. The results may be unreliable.

2.5 Common

The subroutines contain a common block of the form

The single precision version

```
COMMON/MA36C/LP, NREC, MINKA, IOS
```

The double precision version

```
COMMON/MA36CD/LP, NREC, MINKA, IOS
```

LP is an INTEGER variable with default value 6 set by BLOCK DATA that holds the stream number for warning messages. If LP ≤ 0 then no messages are printed.

NREC is an INTEGER variable which is set to the number of records written to file ND.

MINKA is an INTEGER variable which on return with IERR ≤ 1 is set to the minimum length of array A for a guaranteed successful entry.

IOS is an INTEGER variable which is used as the IOSTAT identifier in the statements that read and write to the direct-access file. On return it has the value 0 if no input/output error occurred. Otherwise its value is nonzero and processor dependent.

3 GENERAL INFORMATION

Use of common: uses common area MA36C/CD, see §2.5.

Other routines called directly: uses MA36D/DD, MA36E/ED (block data), MA36F/FD, MA36G/GD and MA36H/HD as private subprograms.

Input/output: diagnostic messages on unit LP (see §2.4 and §2.5) and direct-access file ND is used if necessary (see §2.3).

Restrictions: N ≥ M ≥ 1, LRECL ≥ M, ND ≥ 0.

4 METHOD

Symmetric $\mathbf{LDL}^T$ decomposition without interchanges is used. The elimination proceeds within the array A as long as possible, but if space runs out then a block of rows that are known not to be required later in the elimination are written to disk. The remaining space in A is employed in *wrap-round* fashion, with efficiency maintained by keeping each row contiguous.

5 EXAMPLE OF USE

Suppose that it is required to solve the equation

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}$$

where

$$\mathbf{A} = \begin{pmatrix} \mathbf{T} & -\mathbf{I} & & \\ -\mathbf{I} & \mathbf{T} & -\mathbf{I} & \\ & -\mathbf{I} & \mathbf{T} & \end{pmatrix}, \quad \mathbf{T} = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & -1 & 4 & -1 \\ & & -1 & 4 \end{pmatrix}$$

and

$$b_i = \begin{cases} 1 & i = 7 \\ 0 & \text{otherwise} \end{cases}$$

Suitable code is as follows:

```
      DOUBLE PRECISION A(35), B(12)
      INTEGER IP(12)
      COMMON /MA36CD/ LP,NREC,MINKA,IOS
      EXTERNAL ROW
      KA=35
      ND=51
      LRECL=10
      OPEN (UNIT=51,IOSTAT=IOS,STATUS='SCRATCH',ACCESS='DIRECT',
     *      FORM='UNFORMATTED',RECL=8*LRECL)
      N=12
      M=5
      DO 10 I=1,N
        B(I)=0.0D0
10    CONTINUE
      B(7)=1.0D0
      CALL MA36AD(N,A,KA,B,M,IERR,ROW,ND,LRECL,IP)
      WRITE (6,20) B
20    FORMAT (' ANSWER : '/12(' ',D9.3/))
      STOP
      END
C
C.....................................................
C
      SUBROUTINE ROW(R,M,I,L)
      DOUBLE PRECISION R(M)
      IF (I .LE. 8) L=M
      IF(I. GT. 8) L=2
      IF(I .EQ. 12) L=1
      DO 10 J=1,L
        R(J)=0.0D0
10    CONTINUE
      IF (I .LE. 8) R(M)=-1.0D0
      IF ((I/4)*4 .NE. I) R(2)=-1.0D0
      R(1)=4.0D0
      RETURN
      END
```

This produces the output:

```
 ANSWER:
 0.332D-01
 0.788D-01
 0.133D+00
 0.654D-01
 0.539D-01
 0.149D+00
 0.386D+00
 0.129D+00
 0.332D-01
 0.788D-01
 0.133D+00
 0.654D-01
```