1 SUMMARY

To **solve a sparse symmetric indefinite system of linear equations.** Given a sparse symmetric matrix $\mathbf{A} = \{a_{ij}\}_{n\times n}$ and an n-vector $\mathbf{b}$, this subroutine solves the system $\mathbf{Ax} = \mathbf{b}$.

The method used is a direct method using an $\mathbf{L}\,\mathbf{D}\,\mathbf{L}^T$ factorization, where $\mathbf{L}$ is unit lower triangular and $\mathbf{D}$ is block diagonal with blocks of order 1 and 2. Advantage is taken of the extra sparsity available with 2×2 pivots (blocks of $\mathbf{D}$) with one or both diagonal entries of value zero. The numerical values of the entries are taken in account during the first choice of pivots.

ATTRIBUTES — **Version:** 1.0.2. (5 January 2011) **Types:** Real (single, double). **Calls:** `_GEMM`, `_TPSV`, `_GEMV`, `_TRSV`, `_TPMV`, `I_AMAX`. **Origin:** J.K. Reid, Rutherford Appleton Laboratory. **Original date:** November 2001.

2 HOW TO USE THE PACKAGE

2.1 Argument lists and calling sequences

There are three subroutines that can be called by the user:

(a) `MA67I/ID` sets default values for the elements of the arrays that hold control parameters for the other subroutines.

(b) `MA67A/AD` factorizes a matrix **A** using Gaussian elimination with pivots chosen to preserve sparsity.

(c) `MA67C/CD` uses the factors generated by `MA67A/AD` to solve a system of equations $\mathbf{Ax} = \mathbf{b}$.

The user must call `MA67I/ID` prior to any other call of the package. This sets default values for control parameters. If any non-default values are required, they should be set immediately afterwards. A call to `MA67C/CD` must be preceded by a call to `MA67A/AD`. `MA67C/CD` can be used repeatedly to solve for different right-hand sides **b**.

2.1.1 To set default values of controlling parameters

The single precision version

```
CALL MA67I(CNTL,ICNTL)
```

The double precision version

```
CALL MA67ID(CNTL,ICNTL)
```

`CNTL` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length 5 that need not be set by the user. On return it contains default values. For details, see Section 2.2.

`ICNTL` is an `INTEGER` array of length 10 that need not be set by the user. On return it contains default values. For details, see Section 2.2.

2.1.2 To factorize a matrix

The single precision version

```
CALL MA67A(N,NE,A,IRN,JCN,LFACT,FACT,LIFACT,IFACT,CNTL,ICNTL,RINFO,INFO)
```

The double precision version

```
CALL MA67AD(N,NE,A,IRN,JCN,LFACT,FACT,LIFACT,IFACT,CNTL,ICNTL,RINFO,INFO)
```

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

NE is an INTEGER variable that must be set by the user to the number of entries in the matrix **A**. It is not altered by the subroutine. **Restriction:** NE ≥ 1.

A is a REAL (DOUBLE PRECISION in the D version) array of length NE that must be set by the user so that A(k) holds the value of the diagonal entry or pair of off-diagonal entries whose indices were held in IRN(k) and JCN(k), for k = 1, 2,..., NE. Multiple entries are summed and any that correspond to an IRN(k) or JCN(k) value that was out of range are ignored. It is not altered by the subroutine.

IRN and JCN are INTEGER arrays of length NE. The user must set them so that each diagonal entry a_{ii} is represented by IRN(k)=*i* and JCN(k)=*i* and each pair of off-diagonal entries a_{ij} and a_{ji} is represented by IRN(k)=*i* and JCN(k)=*j* or by IRN(k)=*j* and JCN(k)=*i*. Multiple entries are permitted. If IRN(k) or JCN(k) are less than 1 or greater than N, IRN(k) and JCN(k) are reset to zero and the entry is ignored. These arrays are not otherwise altered by MA67A/AD.

LFACT is an INTEGER variable that must be set by the user to the length of array FACT. It must be at least NE*2+5*N+4, but we recommend that it should be at least 20% greater than this. On return, INFO(6) (see Section 2.2) holds the minimum length needed. LFACT is not altered by the subroutine.

FACT is a REAL (DOUBLE PRECISION in the D version) array of length LFACT that need not be set by the user. On return, FACT will hold the entries of the factors of the matrix **A**. FACT must be preserved for subsequent calls to MA67C/CD.

LIFACT is an INTEGER variable. It must be set by the user to the length of array IFACT, which must be at least NE*2+12*N+10, but we recommend that it should be at least 20% greater than this. On return, INFO(7) (see Section 2.2) holds the minimum length needed. LIFACT is not altered by the subroutine.

IFACT is an INTEGER array of length LIFACT that need not be set by the user. On return, IFACT holds integer indexing information on the matrix factors. IFACT must be preserved for subsequent calls to MA47C/CD.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 4 that contains control parameters and must be set by the user. Default values for the elements are set by the call to MA47I/ID. Details of the control parameters are given in Section 2.2. This array is not altered by the subroutine.

ICNTL is an INTEGER array of length 10 that contains control parameters and must be set by the user. Default values for the elements are set by the call to MA67I/ID. Details of the control parameters are given in Section 2.2. This array is not altered by the subroutine.

RINFO is a REAL (DOUBLE PRECISION in the D version) array of length 4 that need not be set by the user. On exit from MA67A/AD, RINFO(1) will be set to the number of floating-point operations that would always be required for a factorization with the same pivot sequence and RINFO(2) will be set to the number of redundant floating-point operations required because of the use of the Level 3 BLAS routine _GEMM.

INFO is an INTEGER array of length 24 that need not be set by the user. On return from MA67A/AD, a value of zero for INFO(1) indicates that the subroutine has performed successfully. For nonzero values, see Section 2.3. For the meaning of the value of other elements of INFO set by MA67A/AD, see Section 2.2.

2.1.3 To solve equations, given the factorization

The single precision version

```
CALL MA67C(N,LFACT,FACT,LIFACT,IFACT,W,RHS,IW,ICNTL)
```

The double precision version

```
CALL MA67CD(N,LFACT,FACT,LIFACT,IFACT,W,RHS,IW,ICNTL)
```

N is an INTEGER variable that must be set by the user to the order *n* of the matrix **A**. It must be unchanged since the call to MA67A/AD and is not altered by the subroutine.

LFACT is an INTEGER variable that must be set by the user to the length of array FACT. It is not altered by the

subroutine.

FACT is a REAL (DOUBLE PRECISION in the D version) array of length LFACT that must be unchanged since the call to MA67A/AD. It is not altered by the subroutine.

LIFACT is an INTEGER variable that must be set by the user to the length of array IFACT. It is not altered by the subroutine.

IFACT is an INTEGER array of length LIFACT that must be unchanged since the call to MA67A/AD. It is not altered by the subroutine.

W is a REAL (DOUBLE PRECISION in the D version) array of length N that is used as workspace.

RHS is a REAL (DOUBLE PRECISION in the D version) array of length N that must be set by the user so that RHS(i) holds component i of the right-hand side of the equations being solved. On return, it will hold the corresponding component of the solution vector.

IW is an INTEGER array of length N that is used as workspace.

ICNTL is an INTEGER array of length 10 that contains control parameters and must be set by the user. Default values for the elements may be set by a call to MA67I/ID. Details of the control parameters are given in Section 2.2. This array is not altered by the subroutine.

2.2 Arrays for control and information

The elements of the arrays CNTL and ICNTL control the action of MA67A/AD and MA67C/CD. Default values for the elements are set by MA67I/ID. The elements of the arrays RINFO and INFO provide information on the action of MA67A/AD and MA67C/CD.

CNTL(1) has default value 0.1 and is used for threshold pivoting by MA67A/AD. Values greater than 0.5 are treated as 0.5 and less than zero as zero. Since this parameter is used on the assumption that the matrix is well-scaled, it is advisable that the user scale the matrix by calling MC30 before calling MA67A/AD (see Example 5.2).

CNTL(2) has default value zero. If it is set to a positive value, MA67A/AD will treat any pivot whose modulus is less than CNTL(2) as zero.

CNTL(3) has default value 0.9. A row whose proportion of entries is at least this is treated as full.

CNTL(4:5) are not used in the present version.

ICNTL(1) has default value 6 and holds the unit number to which the error messages are sent. A non-positive value suppresses all messages.

ICNTL(2) has default value 6 and holds the unit number to which warning messages and monitoring output are sent. A non-positive value suppresses all such output.

ICNTL(3) is used by the subroutines to control output. It has default value 2. Possible values are:

- 0 None.
- 1 Error messages only.
- 2 Error and warning messages only.
- 3 Error and warning messages, plus scalar parameters and a few entries of arrays on entry and exit from subroutines.
- 4 Error and warning messages, plus all parameter values on entry and exit from subroutines.

ICNTL(4) is used by the subroutine to control pivoting. It has default value 0. Possible values are:

- 0 Pivots to be chosen using the Markowitz criterion with a full search.
- ≥ 2 Pivots to be chosen using the Markowitz criterion with the search limited to ICNTL(4) block rows.

ICNTL(5) has default value 16. It is used by MA67A/AD as the block size for the use of Level 3 BLAS (see Section 4).

ICNTL(6) has default value 10. It limits the number of compresses of the real data structure or of the integer data

structure performed by MA67A/AD.

ICNTL(7) has default value 4. Direct addressing and Level 2 BLAS are used by the routine MA67C/CD on any block of the factorization having more than ICNTL(7) rows.

ICNTL(8:10) are not used in the present version.

RINFO(1:2) are used to record the number of floating-point operations performed (see Section 2.1.2).

RINFO(3:4) are not used in the present version.

INFO(1) has the value zero if the call was successful, a positive value in the case of a warning, and a negative value in the event of an error (see Section 2.3).

INFO(2) is not used in the present version.

INFO(3) gives, on return from MA67A/AD, the number of entries with indices that are out of range.

INFO(4) gives, on return from MA67A/AD, the number of duplicate entries.

INFO(5) gives, on return from MA67A/AD, the number of block pivot steps.

INFO(6) and INFO(7) give minimum lengths for FACT and IFACT for a call of MA67A/AD with the same matrix. Following an error return, it is not guaranteed that these lengths will lead to a successful solution, but they will allow the algorithm to progress further and provide better suggestions. Following a successful return, these lengths will have been calculated with the assumption that an unlimited number of compresses of the data structures is permitted.

INFO(8) is not used in the present version.

INFO(9) gives, on return from MA67A/AD, the maximum front size encountered.

INFO(10) and INFO(11) give, on return from MA67A/AD, the number of REAL (or DOUBLE PRECISION for the D version) and INTEGER words required to hold the matrix factors.

INFO(12) gives, on return from MA67A/AD, the number of compresses of the real data structure performed by MA67A/AD. If this is high (say more than 10), the performance of MA67A/AD may be improved by increasing the length of array FACT.

INFO(13) gives, on return from MA67A/AD, the number of compresses of the integer data structure performed by MA67A/AD. If this is high (say more than 10), the performance of MA67A/AD may be improved by increasing the length of array IFACT.

INFO(14), INFO(15), and INFO(16) give, on return from MA67A/AD, the number of simple tile, oxo, and full 2×2 pivots chosen (see Section 4).

INFO(17) gives, on return from MA67A/AD, the number of negative eigenvalues detected.

INFO(18) gives, on return from MA67A/AD, the number of zero eigenvalues detected.

INFO(19:24) are not used in the present version.

2.3 Error diagnostics

A successful return from MA67A/AD is indicated by a value of INFO(1) equal to zero. Possible nonzero values for INFO(1) are given below. There are no error returns from MA67C/CD.

A negative flag value is associated with an error message that will be output on unit ICNTL(1).

–1 N < 1.

–2 NE < 1.

–3 Failure due to insufficient space allocated to array FACT. INFO(6) is set to a value needed to progress beyond the current point of failure.

–4 Failure due to insufficient space allocated to array IFACT. INFO(7) is set to a value needed to progress beyond the current point of failure.

–5 Failure due to insufficient space allocated to arrays FACT and IFACT. INFO(6) and INFO(7) are set to values needed to progress beyond the current point of failure.

–6 Failure due to the need for more than ICNTL(6) compresses of the real data structure held in FACT.

–7 Failure due to the need for more than ICNTL(6) compresses of the integer data structure held in IFACT.

A positive flag value is associated with a warning message that will be output on unit ICNTL(2).

+1 Index (in IRN or JCN) out of range. The action taken by the subroutine is to set their value to zero and ignore them and continue. INFO(3) is set to the number of faulty entries. Details of the first ten are output on unit ICNTL(2).

+2 There are duplicate entries. These will be summed by MA67A/AD. This number is recorded in INFO(4).

+3 Both warnings +1 and +2 are operative.

+4 The matrix is rank deficient. The number of zero eigenvalues found is given by INFO(18).

+5 Both warnings +1 and +4 are operative.

+6 Both warnings +2 and +4 are operative.

+7 Warnings +1, +2, and +4 are operative.

2.4 Badly-scaled systems

If the user's input matrix has entries differing widely in magnitude, then an inaccurate solution may be obtained. In such cases, the user is advised to first use MC30A/AD to obtain scaling factors for the matrix and then explicitly scale it prior to calling MA67A/AD. Thereafter, both left and right-hand sides should be scaled as indicated in the code following the example in Section 5.2.

3 GENERAL INFORMATION

Use of common: None.

Other routines called directly: MA67F/FD, MA67G/GD, MA67H/HD, MA67J/JD, MA67K/KD, MA67L/LD, MA67M/MD, MA67N/ND, MA67O/OD, MA67P/PD, MA67Q/QD, MA67R/RD, MA67S/SD, MA67T/TD, MA67U/UD, MA67V/VD, MA67W/WD, MA67X/XD MA67Y/YD, MA67Z/ZD. The package uses the Basic Linear Algebra Subprograms SGEMM/DGEMM, STPSV/DTPSV, SGEMV/DGEMV, STRSV/DTRSV, STPMV/DTPMV, and ISAMAX/IDAMAX.

Input/output: Error messages on unit ICNTL(1). Warning messages and additional printing on unit ICNTL(2). Each has default value 6, and printing is suppressed if the value is non-positive.

Restrictions:

N $\geq$ 1.
NE $\geq$ 1.

4 METHOD

MA67A/AD starts by identifying columns of **A** with the same structure, and groups the corresponding variables into supervariables. It then stores a copy of the matrix by blocks corresponding to supervariables. Only one copy of each off-diagonal block is held.

Each pivot is chosen using the Markowitz criterion as either a block on the diagonal or a 2×2 block pivot of the form

$$\begin{pmatrix} 0 & \times \\ \times & \times \end{pmatrix} \qquad \text{or} \qquad \begin{pmatrix} 0 & \times \\ \times & 0 \end{pmatrix},$$

called a tile and an oxo pivot, respectively. We refer to the block row or rows of the pivot as the front and make a temporary copy of this matrix. For a 1×1 block pivot, this is a full rectangular matrix. For an oxo pivot, it has the form

$$\begin{pmatrix} 0 & A_3 & A_5 & A_6 & 0 \\ A_3^T & 0 & 0 & A_7 & A_8 \end{pmatrix}.$$

and for a tile pivot, it has the form

$$\begin{pmatrix} 0 & A_3 & A_6 & 0 \\ A_3^T & A_4 & A_7 & A_8 \end{pmatrix}.$$

For a 1×1 block pivot, as many simple 1×1 pivots and full 2×2 as the stability criterion (`CNTL(1)`) permits are chosen from the block pivot and the front is revised. Similarly, for a block oxo or tile pivot, as many simple oxo or tile pivots as the stability criterion permits are chosen and the front is revised. In all cases, the Schur complement is formed and added back into the main data structure, which may need to be enlarged to accommodate fill-in blocks. The chosen pivots rows are stored for `MA47C/CD`. Any rows of the pivot block that are not chosen as pivots are returned to the main data structure and are not considered again as potential pivots until modified by a later Schur complement. If any set of the supervariables involved in the front now have identical column structures, they are merged into a new bigger supervariable.

Unfortunately, there is no level-3 BLAS for the operation of forming the Schur complement. When the order of this matrix is large, we therefore break the Schur complement into blocks of size `ICNTL(5)` and use `SGEMM/DGEMM` for each block. The means that redundant operations are performed for the diagonal blocks. The number of such redundant operations recorded in `RINFO(2)`.

`MA67C/CD` uses the factors from `MA67A/AD` to solve systems of equations either by loading the appropriate parts of the vectors into an array of the current front size and using full matrix code or by indirect addressing at each stage. This is determined by control parameter `ICNTL(7)` which has default value set so that any block pivot with more than 4 rows uses direct addressing.

The pivotal strategy is explained by Duff, Gould, Reid, Scott, and Turner, *The factorization of sparse symmetric indefinite matrices* (IMA J. Numer. Anal. **11**, 181-204, 1991).

5 EXAMPLE OF USE

5.1 Solving sparse equations without scaling.

We illustrate the use of the package on the solution of the single set of equations given by:

$$\begin{pmatrix} 2 & 3 & & & \\ 3 & 0 & 4 & & 6 \\ & 4 & 1 & 5 & \\ & & 5 & 0 & \\ & 6 & & & 1 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 8 \\ 45 \\ 31 \\ 15 \\ 17 \end{pmatrix}$$

Note that this example does not illustrate all the facilities.

Program

```
C Simple example of use of MA67 package
      INTEGER LFACT, LIFACT, NEMAX, NMAX
      PARAMETER (LFACT=200, LIFACT=200, NEMAX=10, NMAX=5)
      INTEGER IRN(NEMAX),JCN(NEMAX),IFACT(LIFACT),
     *        IW(2*NMAX+2)
      INTEGER I,ICNTL(10),INFO(24),N,NE
      DOUBLE PRECISION A(NEMAX),FACT(LFACT),W(NMAX),RHS(NMAX),CNTL(5),
     *                 RINFO(4)

C Set default parameters
      CALL MA67ID(CNTL,ICNTL)

C Read matrix and right-hand side
      READ (5,*) N,NE
      IF (N.GT.NMAX .OR. NE.GT.NEMAX) THEN
        WRITE(6,'(A,2I10)') ' Immediate return N or NE too large = ',
     *                      N, NE
        STOP
      ENDIF
      READ (5,*) (IRN(I),JCN(I),A(I),I=1,NE)
      READ (5,*) (RHS(I),I=1,N)

C Factorize matrix
      CALL MA67AD(N,NE,A,IRN,JCN,LFACT,FACT,LIFACT,IFACT,
     *            CNTL,ICNTL,RINFO,INFO)

C Solve the equations
      CALL MA67CD(N,LFACT,FACT,LIFACT,IFACT,W,RHS,IW,ICNTL)

C     Print out solution vector.
      WRITE(6,'(/A/(1P,5D13.5))') ' The solution vector is:',
     *                        (RHS(I),I=1,N)
      END
```

Data

```
 5 7
 1 1 2.0
 1 2 3.0
 2 3 4.0
 2 5 6.0
 3 3 1.0
 3 4 5.0
 5 5 1.0
 8. 45. 31. 15. 17.
```

Output

```
 The solution vector is:
  1.00000D+00  2.00000D+00  3.00000D+00  4.00000D+00  5.00000D+00
```

5.2 Solving sparse equations using scaling.

In the example code shown below we scale the matrix and right-hand vector (see Section 2.4) prior to solution of the linear equations.

```
C Example of use of scaling with MA67
      INTEGER LFACT, LIFACT, NEMAX, NMAX
      PARAMETER (LFACT=200, LIFACT=200, NMAX=20, NEMAX=100)
      DOUBLE PRECISION CNTL(5),RINFO(4),A(NEMAX),FACT(LFACT),
     *                 RHS(NMAX),W(4*NMAX)
      INTEGER I,ICNTL(10),II,INFO(24),IRN(NEMAX),IFACT(LIFACT),J,
```

```
     *        JCN(NEMAX),N,NE,IW(2*NMAX+2),LP,IFAIL
      DOUBLE PRECISION S(NMAX)

C     Read in input matrix.
      READ(5, * ) N,NE
      IF (N.GT.NMAX .OR. NE.GT.NEMAX) THEN
         WRITE(6,'(A)') ' Error in input data for N and/or NE'
         STOP
      END IF
      READ(5, * ) (IRN(I), JCN(I), A(I), I=1,NE)

C     Scale input matrix
      LP = 6
      CALL MC30AD(N,NE,A,IRN,JCN,S,W,LP,IFAIL)
      IF (IFAIL.LT.0) THEN
        WRITE(6,'(A)') ' Failure in scaling routine'
        STOP
      ENDIF
      DO 10 I=1,N
        S(I)=EXP(S(I))
  10  CONTINUE
      DO 20 II=1,NE
        I=IRN(II)
        J=JCN(II)
        A(II)=A(II)*S(I)*S(J)
  20  CONTINUE

C     Set default controls
      CALL MA67ID(CNTL,ICNTL)

C Factorize matrix
      CALL MA67AD(N,NE,A,IRN,JCN,LFACT,FACT,LIFACT,IFACT,
     *            CNTL,ICNTL,RINFO,INFO)

C     Read in right hand side.
      READ(5, * ) (RHS(I), I=1,N)

C     Scale the right hand side vector by row weights
      DO 30 I=1,N
        RHS(I)=RHS(I)*S(I)
  30  CONTINUE

C Solve the equations
      CALL MA67CD(N,LFACT,FACT,LIFACT,IFACT,W,RHS,IW,ICNTL)

C     Scale the right-hand side vector by column weights
      DO 40 I=1,N
        RHS(I)=RHS(I)*S(I)
  40  CONTINUE

C     Print out solution vector.
      WRITE(6,'(/A/(1P,5D13.5))') ' The solution vector is:',
     *                        (RHS(I),I=1,N)
      END
```

Thus if, in this example we wish to solve:

$$\begin{pmatrix} 3.14E5 & 7.5E1 & \\ 7.5E1 & 3.2E{-}3 & 0.3 \\ & 0.3 & 4.1E2 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 3.1415E5 \\ 7.59064E1 \\ 1.2306E3 \end{pmatrix}$$

we have as input

```
      3     5
      1     1       3.14000D+05
      2     3       0.30
      3     3       4.10000D+02
      1     2       7.50000D+01
      2     2       3.20000D-03
   0.31415D+06  0.759064D2   0.12306D4
```

and output would be

```
  The solution vector is:
   1.00000D+00  2.00000D+00  3.00000D+00
```