

1 SUMMARY

To solve a symmetric, sparse and positive definite set of linear equations $Ax=b$ i.e.

$$\sum_{j=1}^n a_{ij}x_j = b_i \quad i=1,2,\dots,n$$

The solution is found by a preconditioned conjugate gradient technique, where the preconditioning is done by incomplete factorization.

- (a) MA61A and MA61AD perform the incomplete factorization based on an LDL^T decomposition. New entries which have small numerical values compared to the corresponding diagonal entries are dropped, and the diagonal entries are modified to ensure positive definiteness. This results in a preconditioning matrix C held in LDL^T form.
- (b) MA61B and MA61BD perform the iteration procedure using the preconditioned coefficient matrix (i.e. AC^{-1}) as the iteration matrix for the conjugate gradient algorithm.

ATTRIBUTES — **Version:** 1.1.0. (20 December 2005) **Types:** Real (single, double). **Remark:** MA61 is a threadsafe version of MA31. **Original date:** MA31: January 1979, MA61: June 2001. **Origin:** N. Munksgaard, Danish Natural Science Research Council, Grant. No. 511-10069.

2 HOW TO USE THE PACKAGE

2.1 Initialization

The MA61I/ID entry must be called prior to the first call to the MA61A/AD entry to initialize the control and private work arrays.

The single precision version

```
CALL MA61I(ICNTL,CNTL,KEEP)
```

The double precision version

```
CALL MA61ID(ICNTL,CNTL,KEEP)
```

ICNTL is an INTEGER array of length 5, see Section 2.5.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 3, see Section 2.5.

KEEP is the INTEGER array of length 12 described in Section 2.2.

2.2 The argument list and calling sequence of the entry to factorize a matrix incompletely

The single precision version:

```
CALL MA61A(N,NZ,A,INI,INJ,IAI,IAJ,IK,IW,W,C,ICNTL,CNTL,INFO,KEEP)
```

The double precision version:

```
CALL MA61AD(N,NZ,A,INI,INJ,IAI,IAJ,IK,IW,W,C,ICNTL,CNTL,INFO,KEEP)
```

N is an INTEGER variable which must be set by the user to n the order of the matrix **A**. This argument is not altered by the subroutine. **Restriction:** $1 \leq n$.

NZ is an INTEGER variable which must be set by the user to the number of nonzeros in the upper triangular part of matrix **A**. This argument is not altered by the subroutine. **Restriction:** $NZ \geq 1$

- A** is a REAL (DOUBLE PRECISION in the D version) array of length IAJ and $A(K)$, $K=1, NZ$, must be set by the user to contain the nonzero elements of the upper triangular part (including the diagonal) of the matrix **A**. On exit the first NZ locations contain the off-diagonal nonzeros of the upper triangular part of the matrix held in order by rows. The remaining part of the array **A** holds the factors of the incomplete factorized form of the input matrix. It must be passed on to the **B** entry unaltered (see §2.3).
- INI** is an INTEGER array of length IAI. On entry, $INI(K)$, $K=1, NZ$ must hold the row index of the nonzeros stored in $A(K)$. It is used as workspace by the subroutines and must be passed on to the **B** entry unaltered (see §2.3).
- INJ** is an INTEGER array of length IAJ. On entry, $INJ(K)$, $K=1, NZ$ must hold the column index of the nonzeros stored in $A(K)$. On exit $INJ(K)$, $K=1, NZ$ contains the column indices of the off-diagonal nonzeros of the original matrix contained in the array **A**. They are held in order by rows, with row 1 starting in location 1. $INJ(K)$, $K=NZ+1, \dots$ contains the column indices of the factors of the incomplete factorized matrix. The array must be passed on to the **B** entry unaltered (see §2.3).
- IAI** is an INTEGER variable which must be set by the user to the length of the array **INI**. This argument is not altered by the subroutine. (for choice of IAI see §4.1). **Restriction:** $IAI \geq NZ$.
- IAJ** is an INTEGER variable which must be set by the user to the length of the arrays **INJ** and **A**. Since these arrays hold the input matrix as well as its incomplete factorized form IAJ must be at least equal to $2 \cdot NZ$. This argument is not altered by the subroutine (for choice of IAJ see §4.1). **Restriction:** $IAJ \geq 2 \cdot NZ$.
- IK** is an INTEGER array of length $4n$ used as workspace by the subroutine and must be passed on to the **B** entry unaltered (see §2.3).
- IW** is an INTEGER array of length $4n$ which is used as workspace by the subroutine.
- W** is a REAL (DOUBLE PRECISION in the D version) array of length at least $3n$ It is used as workspace by the subroutine and must be passed on to the **B** entry unaltered (see §2.3).
- C** is a REAL (DOUBLE PRECISION in the D version) variable which must be set by the user. A new entry a_{ij} say created during the factorization is ignored if its numerical value is less than $|c| \sqrt{a_{ii} a_{jj}}$. If **C** has a negative input value then it is not altered by the subroutine. If **C** has a non-negative input value it may be changed during the decomposition with the aim of using the available space to obtain an accurate factorization. (The choice of value for **C** is discussed in §4.1).
- ICNTL** is an INTEGER array of length 5, see Section 2.5.
- CNTL** is a REAL (DOUBLE PRECISION in the D version) array of length 3, see Section 2.5.
- INFO** is an INTEGER array of length 10, see Section 2.5.
- KEEP** is an INTEGER array of length 12 provided by the user for MA61 to use as private workspace. It must be passed to MA61I/ID to be initialized and thereafter must not be altered by the user.

2.3 The argument list and calling sequence of the entry to solve $Ax = b$ using the preconditioning

The single precision version:

```
CALL MA61B(N, NZ, A, INI, INJ, IAJ, IK, B, W, W1, KMAX, EPS, ICNTL, INFO, KEEP)
```

The double precision version:

```
CALL MA61BD(N, NZ, A, INI, INJ, IAJ, IK, B, W, W1, KMAX, EPS, ICNTL, INFO, KEEP)
```

N, NZ, A, INI, INJ, IAJ, IK. These arguments (and **W**) correspond to those of the same name in MA61A and MA61AD, and should not be changed between calls to the **A** and **B** entries.

B is a REAL (DOUBLE PRECISION in the D version) array of length at least n on entry it should contain the right-hand side b_i , $i=1, 2, \dots, n$ of the equations and on exit it contains the computed solution x_j , $j=1, 2, \dots, n$.

W is a REAL (DOUBLE PRECISION in the D version) array of length $3n$ which must be passed on from the **A** entry

call with its first $2n$ locations unchanged.

W1 is a REAL (DOUBLE PRECISION in the D version) array of length at least $3n$. It is used for workspace in the subroutine.

KMAX is an INTEGER array of length at least 2 and KMAX(1) must be set by the user to the maximum number of iterations to be performed by the subroutine. On return, KMAX(2) will have been set by the subroutine to the number of iterations actually performed to reach the desired accuracy of the solution, while KMAX(1) remains unchanged.

EPS is a REAL (DOUBLE PRECISION in the D version) array of length at least 2 and EPS(1) must be set by the user to ϵ_1 the desired accuracy of the solution. The iterative procedure is stopped if

$$\|R\|_2 = \sqrt{\sum_{i=1}^n r_i^2} \leq \epsilon_1$$

where

$$r_i = \sum_{j=1}^n a_{ij}x_j - b_i \quad i=1,2,\dots,n.$$

On return EPS(2) is given the obtained value of $\|R\|_2$. (If on return EPS(2) is greater than EPS(1) the iterative procedure has been stopped because the limit on the number of iterations has been reached before the desired accuracy was obtained).

ICNTL is an INTEGER array of length 5, see Section 2.5.

INFO is an INTEGER array of length 10, see Section 2.5.

KEEP is the INTEGER private workspace array of length 12 used in the MA61A/AD entry which must be passed to the MA61B/BD entry unaltered, see Section 2.2.

2.4 Error diagnostics

A successful return from any of the entries is indicated by a value of INFO(1) equal to zero. Possible nonzero values for INFO(1) are given below and divided into two groups. Errors are indicated by negative values and warnings by positive values.

INFO(1) values indicating errors in the matrix **A**.

- 1 $n < 1$ violated
- 2 $NZ \leq 0$
- 3 $IAI < NZ$
- 4 $IAJ < 2 * NZ$
- 5 Non-zero held in position K of arrays A, INI and INJ does not belong to the upper triangular part of the matrix, i.e. one of the conditions $0 < INI(K) \leq INJ(K) \leq N$ is not satisfied.
- 6 Zero or negative elements on the diagonal.

INFO(1) values giving warnings about unexpected incidents during the execution

- 1 There is more than one nonzero value having the row and column index given by INI(K) and INJ(K). The calculations are continued with a value equal to the sum of the duplicate elements.
- 2 It has been necessary to modify the diagonal element d_{jj} to keep its value positive. d_{jj} is set to $\max_{k>j+1} \{a_{jk}^{(j)}\}$.
- 3 The solution has not reached the desired accuracy in the allowed number of iterations. The accuracy of the obtained solution is returned in the parameter EPS and is equal to the norm of the residual vector.

- 4 The available space for the factorized form has been used before 90 percent of the pivot steps have been taken. This is caused by a too small input value of $|c|$. The users response should be either to increase the value of $|c|$ or to allocate more storage for the arrays A, INJ and INI.

2.5 The control and information arrays

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

ICNTL(1) specifies the unit number to be used to output error messages when INFO(1) is returned with a negative value. The value zero indicates that such messages are to be suppressed. It has a default value of 6, which is set by MA61I/ID.

ICNTL(2) specifies the unit number to be used to output warning messages when INFO(1) is returned with a positive value. The value zero indicates that such messages are to be suppressed. It has a default value of 6, which is set by MA61I/ID.

ICNTL(3) to ICNTL(5) should not be altered and at present are not used by MA61.

The CNTL array argument which must be of length 3 can be used to pass optional floating point control values to the routine.

CNTL(1) is used by the routine to determine if the matrix should be treated as full. It is considered as full if its density is at least CNTL(1). It has a default value of 0.8, which is set by MA61I/ID.

CNTL(2) and CNTL(3) should not be altered and at present are not used by MA61.

The INFO array argument which must be of length 10 is used to return information to the user.

INFO(1) is set to an error code. On exit the value 0 indicates a successful run. For nonzero values see §2.4.

INFO(2) contains the number of locations in A and INJ that are in use for the incomplete factorized form of the matrix.

INFO(3) contains the number of locations in INI that are in use for the incomplete factorized form of the matrix.

INFO(4) is the number of compressions that have been performed in arrays INJ, A and INI during the factorization.

INFO(5) is the number of entries from the coefficient matrix that corresponds to diagonal and duplicate elements. If no duplicate elements have been encountered INFO(5) will have the value n .

INFO(6) is the number of the pivot step from which the active matrix has the density CNTL(1).

INFO(7) to INFO(10) at present are not used by MA61.

3 GENERAL INFORMATION

Use of common: none.

Workspace: provided by the user through the arguments IW, W and W1 of lengths $4n, 3n, 3n$.

Other routines called directly: The A entry calls MA61C/CD, MA61D/DD and MA61E/ED, and the B entry calls MA61F/FD, MA61G/GD and MA61H/HD.

Input/output: Error and warning messages only. Error messages on unit ICNTL(1), warning messages on unit ICNTL(2). Both have default value 6 and output is suppressed if they are set to zero.

Restrictions:

$1 \leq NZ,$
 $IAJ \geq 2 * NZ,$
 $IAI \geq NZ.$

4 METHOD

The preconditioned conjugate gradient method is used for the solution, and it is basically the algorithm by Hestenes and Stiefel (1952) in which the coefficient matrix $\mathbf{A}$ has been preconditioned by the inverse of an incomplete factorization of $\mathbf{A}$.

A more detailed description is given by Munksgaard (1980).

4.1 Choice of parameters C , IAI and IAJ .

The incomplete factorization drops element a_{ij} if it is a potential fill-in which is smaller than $|c|\sqrt{a_{ii}a_{jj}}$. If $\mathbf{A}$ is positive definite and c has the value -1 , all fill-ins will be dropped during the factorization. A sufficient size for IAI and IAJ will in that case be NZ and $2*NZ$. If the numerical value of C is smaller, less elements will be dropped provided there is additional space in the arrays to accommodate them. For $C=0$ a complete factorization is performed.

4.2 Choice of $CNTL(1)$.

The parameter $CNTL(1)$ has the default value 0.8. This means that the active part of the matrix is factorized as a full matrix from the pivot step in which its density exceeds the value 0.8, where the density is defined as the number of nonzeros over the number of coefficients in a full matrix of the same order.

When $CNTL(1)$ is less than one it means that a number of zeros are incorporated in the factorization, and to minimize the storage requirements $CNTL(1)$ must therefore be set to one. However the increase in storage is quite moderate for $CNTL(1) = 0.8$ and for this value a minimum execution time for the total factorization is obtained for a large number of test examples.

References

- Hestenes, MR. and Stiefel, E. (1952) Methods of conjugate gradients for solving linear systems. NBS J. Res 49, 409-436
- Munksgaard, N. (1980) Solving sparse symmetric sets of linear equations by preconditioned conjugate gradients. *ACM Trans. Math. Softw.* **6**, 206-219 .

$$\begin{bmatrix} 1 & -\frac{1}{4} & & & & & & & & & & & & & & & \\ -\frac{1}{4} & 1 & -\frac{1}{4} & & & & & & & & & & & & & & \\ & -\frac{1}{4} & 1 & -\frac{1}{4} & & & & & & & & & & & & & \\ & & -\frac{1}{4} & 1 & 0 & & & & & & & & & & & & \\ -\frac{1}{4} & & & 0 & 1 & -\frac{1}{4} & & & & & & & & & & & \\ & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & & & & & & & & \\ & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & & & & & & & \\ & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & 0 & & & & & & & & \\ & & & & -\frac{1}{4} & & & 0 & 1 & -\frac{1}{4} & & & & & & & \\ & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & & & & \\ & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & 0 & & & & & \\ & & & & & & & -\frac{1}{4} & & & 0 & 1 & -\frac{1}{4} & & & & \\ & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & & & \\ & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & 0 & & \\ & & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & -\frac{1}{4} & \\ & & & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 & \\ & & & & & & & & & & & & -\frac{1}{4} & & & -\frac{1}{4} & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \\ x_{16} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{2} \\ \frac{1}{4} \\ 0 \\ 0 \\ \frac{1}{4} \\ \frac{1}{4} \\ 0 \\ 0 \\ \frac{1}{4} \\ \frac{1}{2} \\ \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{2} \end{bmatrix}$$

Figure 1. The example equations

5 EXAMPLE OF USE

Suppose we are going to solve the Laplace equation $U_{xx} + U_{yy} = 0$ in a unit square using 5-point formula in a 6 by 6 discretization with $u=1.0$ on the boundaries. The 16 interior nodes will produce the set of equations shown in Figure 1.

The Fortran code shown in Figure 2. will generate the coefficient matrix, and factorize it incompletely. In the factorization all new entries less than 10^{-1} times their corresponding diagonal entries are ignored. The factorized form is used as a preconditioning matrix in a conjugate gradient iteration to find the solution.

Figure 2. The example program and output.

```

      DOUBLE PRECISION A(90),W(100),B(16),EPS(2),C
      INTEGER IW(16,4),KMAX(2)
      INTEGER INI(50),INJ(90),IK(16,4)
      DOUBLE PRECISION CNTL(3)
      INTEGER KEEP(12),ICNTL(5),INFO(10)
      INTEGER IAI,IAJ,N,M,NN,MM1,I,K,J,JJ,NP1,NZ
      DATA IAI,IAJ,N,M/50,90,4,4/,C,EPS(1),KMAX(1)/
      *-1.D-1,1.D-6,50/
      CALL MA61ID(ICNTL,CNTL,KEEP)
      NN=N*M
C Initialize the right-hand side.
      MM1=M-1
      DO 4 I=1,N
        K=(I-1)*M
        B(K+1)=0.25
        DO 3 J=2,MM1
          B(K+J)=0.0
        3 CONTINUE
        B(K+M)=0.25
      4 CONTINUE
      K=N*MM1
      DO 5 J=1,M
        B(J)=B(J)+0.25
        B(K+J)=B(K+J)+0.25
      5 CONTINUE
C Set up the coefficient matrix.
      K=1
      DO 10 I=1,NN
        A(K)=1.0
        INI(K)=I
        INJ(K)=I
        K=K+1
      10 CONTINUE
      DO 25 I=1,N
        DO 20 J=1,MM1
          JJ=(I-1)*N+J
          A(K)=-0.25
          INI(K)=JJ
          INJ(K)=JJ+1
          K=K+1
        20 CONTINUE
      25 CONTINUE
      NP1=N+1
      DO 30 I=NP1,NN
        A(K)=-0.25
        INI(K)=I-N
        INJ(K)=I
        K=K+1
      30 CONTINUE

```

```

30  CONTINUE
    NZ=K-1
C   Perform the preconditioning work
    CALL MA61AD(NN,NZ,A,INI,INJ,IAI,IAJ,IK,IW,W,C,
+           ICNTL,CNTL,INFO,KEEP)
C   Call the iterative procedure.
    CALL MA61BD(NN,NZ,A,INI,INJ,IAJ,IK,B,W,W(51),KMAX,EPS,
+           ICNTL,INFO,KEEP)
    WRITE(6,1001) INFO(1)
    WRITE(6,1002) KMAX(2),EPS(2)
    WRITE(6,1003)(B(I),I=1,NN)
1001 FORMAT(///' For this run INFO(1) has the value',I6)
1002 FORMAT(///' Number of iterations ',I7,
1       ' norm of residual ',E12.3)
1003 FORMAT(4E17.7)
    END

```

The output from this run:

For this run INFO(1) has the value 0

Number of iterations	4	norm of residual	0.353E-06
0.1000000E+01	0.9999999E+00	0.1000000E+01	0.9999999E+00
0.1000000E+01	0.9999999E+00	0.1000000E+01	0.9999999E+00
0.1000000E+01	0.9999999E+00	0.9999999E+00	0.9999999E+00
0.1000000E+01	0.1000000E+01	0.1000000E+01	0.1000000E+01