MA47

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 multifrontal sparse Gaussian elimination and is discussed by Duff and Reid (1995), *MA47, A Fortran code for direct solution of indefinite sparse symmetric linear systems,* Report RAL-95-001, Rutherford Appleton Laboratory, Oxfordshire.

ATTRIBUTES — Version: 1.0.0. **Types:** `MA47A`, `MA47AD`. **Calls:** `_GEMM`, `_TPSV`, `_GEMV`, `_TRSV`, `_TPMV`, `I_AMAX`. **Origin:** I.S. Duff and J.K. Reid, Rutherford Appleton Laboratory. **Original date:** May 1993.

2 HOW TO USE THE PACKAGE

Although there are both single and double precision versions of the routine available, the user is **strongly advised** to use the double precision version unless single precision on his or her machine actually means 8-byte arithmetic.

2.1 Argument lists and calling sequences

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

(a) `MA47I/ID` sets default values for the components of the arrays that hold control parameters for the other subroutines.

(b) `MA47A/AD` accepts the pattern of **A** and chooses pivots for Gaussian elimination to preserve sparsity. It also constructs information for actual factorization by `MA47B/BD`. The user may provide a pivot sequence, in which case the necessary information for `MA47B/BD` will be generated.

(c) `MA47B/BD` factorizes a matrix **A** using the information from a previous call to `MA47A/AD`. The actual pivot sequence used may differ from that of `MA47A/AD`.

(d) `MA47C/CD` uses the factors generated by `MA47B/BD` to solve a system of equations $\mathbf{Ax} = \mathbf{b}$.

Normally the user would call `MA47I/ID` prior to any other call of the package. If non-default values for any of the control parameters are required, they should be set immediately after the call to `MA47I/ID`. A call to `MA47C/CD` must be preceded by a call to `MA47B/BD` which in turn must be preceded by a call to `MA47A/AD`. Since the information passed from one subroutine to the next is not corrupted by the second, several calls to `MA47B/BD` for matrices with the same sparsity pattern but different values may follow a single call to `MA47A/AD`, and similarly `MA47C/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 MA47I(CNTL,ICNTL)
```

The double precision version

```
        CALL MA47ID(CNTL,ICNTL)
```

`CNTL` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length 2 that need not be set by the user. On return it contains default values. For further information see Section 2.2.

`ICNTL` is an `INTEGER` array of length 7 that need not be set by the user. On return it contains default values. For further information see Section 2.2.

2.1.2 To perform symbolic manipulations

The single precision version

```
CALL MA47A(N,NE,IRN,JCN,IW,LIW,KEEP,ICNTL,RINFO,INFO)
```

The double precision version

```
CALL MA47AD(N,NE,IRN,JCN,IW,LIW,KEEP,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:** $1 \le$ `N` $\le$ *HUGE*/3, where *HUGE* is the largest possible integer value, equivalent to that defined by Fortran 90 as `HUGE(N)`.

`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.

`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. Entries (on or off the diagonal) that are known to be zero should be excluded. 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. For some off-diagonal entries, the values of `JCN(k)` and `IRN(k)` are interchanged by the subroutine. These arrays are not otherwise altered by `MA47A/AD`. `JCN` must be preserved between a call to `MA47A/AD` and subsequent calls to `MA47B/BD`.

`IW` is an `INTEGER` array of length `LIW`. This is used as workspace by the subroutine. Its length must be at least `NE*2+5*N+4` (or `NE+5*N+4` if the pivot order is specified in `KEEP`), but we recommend that it should be at least 20% greater than this. `INFO(12)` (see Section 2.2) provides a means of checking whether the length is adequate.

`LIW` is an `INTEGER` variable. It must be set by the user to the length of array `IW` and is not altered by the subroutine.

`KEEP` is an `INTEGER` array of length `NE+5*N+2`. If the user wishes to provide the pivot sequence, the index of the variable in position `i` of the pivot order must be placed in `KEEP(i)`, `i = 1`, 2,..., `N` and `ICNTL(4)` must be set to `1`; the variables of any 2×2 pivot required must be adjacent in the sequence and their indices must be negated within `KEEP`. The given order is likely to be replaced by one that is equivalent apart from reordering of additions and subtractions. Otherwise, `KEEP` need not be set by the user. It must be preserved between a call to `MA47A/AD` and subsequent calls to `MA47B/BD`.

`ICNTL` is an `INTEGER` array of length 7 that contains control parameters and must be set by the user. Default values for the components may be set by a call to `MA47I/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 `MA47A/AD`, `RINFO(1)` will be set to the number of floating-point operations that would always be required by a subsequent factorization that exactly respects the pivot sequence chosen by `MA47A/AD` 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`. `RINFO(3)` and `RINFO(4)` are not accessed by `MA47A/AD`.

`INFO` is an `INTEGER` array of length 24 that need not be set by the user. On return from `MA47A/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 components of `INFO` set by `MA47A/AD`, see Section 2.2.

2.1.3 To factorize a matrix

The single precision version

```
CALL MA47B(N,NE,JCN,A,LA,IW,LIW,KEEP,CNTL,ICNTL,IW1,RINFO,INFO)
```

The double precision version

```
CALL MA47BD(N,NE,JCN,A,LA,IW,LIW,KEEP,CNTL,ICNTL,IW1,RINFO,INFO)
```

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 MA47A/AD and is not altered by the subroutine.

NE is an INTEGER variable that must be set by the user to the number of entries in the matrix **A**. It must be unchanged since the call to MA47A/AD and is not altered by the subroutine.

JCN is an INTEGER array of length NE that must be unchanged since the call to MA47A/AD and is not altered by the subroutine.

A is a REAL (DOUBLE PRECISION in the D version) array of length LA 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) on entry to MA47A/AD, 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. On return, A will hold the entries of the factors of the matrix **A**. A must be preserved between calls to this subroutine and subsequent calls to MA47C/CD.

LA is an INTEGER variable that must be set by the user to the length of array A. It must be at least as great as INFO(6) as output by MA47A/AD (see Section 2.2). It is advisable to allow a greater value because the use of numerical pivoting may increase storage requirements. It is not altered by the subroutine.

IW is an INTEGER array of length LIW that need not be set by the user. It is used as workspace by MA47B/BD and on return holds integer indexing information on the matrix factors. It must be preserved by the user between calls to this subroutine and subsequent calls to MA47C/CD.

LIW is an INTEGER variable that must be set by the user to the length of array IW. It must be at least as great as INFO(7) as output from MA47A/AD (see Section 2.2). A greater value is recommended because numerical pivoting may increase storage requirements. It is not altered by the subroutine.

KEEP is an INTEGER array of length NE+5*N+2 that must be unchanged since the call to MA47A/AD. It is not altered by the subroutine.

CNTL is a REAL (DOUBLE PRECISION in the D version) array of length 2 that contains control parameters and must be set by the user. Default values for the components may be set by a 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 7 that contains control parameters and must be set by the user. Default values for the components may be set by a call to MA47I/ID. Details of the control parameters are given in Section 2.2. This array is not altered by the subroutine.

IW1 is an INTEGER array of length 2*N+2. It is used as workspace 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 MA47B/BD, RINFO(3) 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(4) will be set to the number of redundant floating-point operations performed because of the use of the Level 3 BLAS routine GEMM. RINFO(1) and RINFO(2) are not accessed by MA47B/BD.

INFO is an INTEGER array of length 24 that need not be set by the user. On return from MA47A/AD, a value of zero for INFO(1) indicates that the subroutine has performed successfully. For nonzero values of INFO(1), see Section 2.3. For the meaning of the value of other components of INFO set by MA47B/BD see Section 2.2.

2.1.4 To solve equations, given the factorization

The single precision version

```
CALL MA47C(N,A,LA,IW,LIW,W,RHS,IW1,ICNTL)
```

The double precision version

```
CALL MA47CD(N,A,LA,IW,LIW,W,RHS,IW1,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 `MA47B/BD` and is not altered by the subroutine.

`A` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length `LA` that must be unchanged since the call to `MA47B/BD`. It is not altered by the subroutine.

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

`IW` is an `INTEGER` array of length `LIW` that must be unchanged since the call to `MA47B/BD`. It is not altered by the subroutine.

`LIW` is an `INTEGER` variable that must be set by the user to the length of array `IW`. 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 the i-th component of the right-hand side of the equations being solved. On return, it will hold the corresponding entry of the solution vector.

`IW1` is an `INTEGER` array of length `N` that is used as workspace.

`ICNTL` is an `INTEGER` array of length 7 that contains control parameters and must be set by the user. Default values for the components may be set by a call to `MA47I/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 `MA47A/AD`, `MA47B/BD`, and `MA47C/CD`. Default values for the elements are set by `MA47I/ID`. The elements of the arrays `RINFO` and `INFO` provide information on the action of `MA47A/AD`, `MA47B/BD`, and `MA47C/CD`.

`CNTL(1)` has default value 0.001 and is used for threshold pivoting by `MA47B/BD`. 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 `MA47B/BD` (see Example 5.2).

`CNTL(2)` has default value zero. If it is set to a positive value, `MA47B/BD` will treat any pivot whose modulus is less than `CNTL(2)` as zero.

`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 additional printing is sent. A non-positive value suppresses all such printing.

`ICNTL(3)` is used by the subroutines to control printing. It has default value 1. Possible values are:

0	No printing.
1	Error messages only.
2	Error and warning messages only.
3	Scalar parameters and a few entries of arrays on entry and exit from subroutines.
4	All parameter values printed on entry and exit from subroutines.

ICNTL(4) has default value 10. It must be set by the user to a value of 1 when calling MA47A/AD if a pivot sequence is being supplied by the user in array KEEP. If the value is 0, the pivot order is chosen automatically by the Markowitz strategy. If greater than 1, the pivot order is chosen automatically, but each search for a structured pivot limited to this number of rows.

ICNTL(5) has default value 5. It is used by MA47A/AD and MA47B/BD as the block size for the use of Level 3 BLAS (see Section 4). A value greater than N will mean that each frontal matrix is treated as a single block.

ICNTL(6) has default value 5. MA47A/AD may amalgamate tree nodes (see Section 4) in order to decrease the amount of indirect addressing, but at the cost of more arithmetic. Two nodes are merged at the risk of more arithmetic only if both involve less than ICNTL(6) eliminations.

ICNTL(7) has default value 4. Direct addressing and Level 2 BLAS are used by the routine MA47C/CD on any block of the factorization having more than ICNTL(7) rows.

RINFO(1:4) are used to record the number of floating-point operations performed (see Sections 2.1.2 and 2.1.3).

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) gives additional information on some returns:

MA47A/AD or MA47B/BD with INFO(1) = –3: a length for IW sufficient to continue after the point of failure.

MA47B/BD with INFO(1) = –4: a length for A sufficient to continue after the point of failure.

MA47A/AD with INFO(1) = –5 or –6: the index in KEEP of the faulty component.

INFO(3) gives, on return from MA47A/AD, the number of entries with indices that are out of range.

INFO(4) gives, on return from MA47A/AD, the number of duplicate entries.

INFO(5) gives, on return from MA47A/AD, the number of nodes in the assembly tree.

INFO(6) and INFO(7) give, on return from MA47A/AD and MA47B/BD, the lengths of A and IW required for a successful completion of MA47B/BD with the same pivot sequence. The actual lengths required may be greater because of numerical pivoting.

INFO(8) gives, on return from MA47A/AD, the number of zero eigenvalues detected in the structure of **A**.

INFO(9) gives, on return from MA47A/AD, the maximum front size encountered.

INFO(10) and INFO(11) give, on return from MA47A/AD, the number of REAL (or DOUBLE PRECISION for the D version) and INTEGER words required to hold the matrix factors provided the same pivot sequence is employed. Numerical pivoting may change this.

INFO(12) gives, on return from MA47A/AD, the number of compresses of the internal data structure performed by MA47A/AD. If this is high (say more than 10), the performance of MA47A/AD may be improved by increasing the length of array IW.

INFO(13) and INFO(14) give, on return from MA47A/AD, the number of tile and oxo pivots chosen (see Section 4).

INFO(15) is set by MA47B/D to the maximum front size encountered.

INFO(16) and INFO(17) give, on return from MA47B/BD, the amount of REAL (or DOUBLE PRECISION for the D version) and INTEGER words actually used to hold the factorization.

INFO(18) gives, on return from MA47B/BD, the number of compresses performed on the real data. If it is high (say more than 10), the speed of the factorization may be increased by allocating more space to the array A.

INFO(19) gives, on return from MA47B/BD, the number of compresses performed on the integer data. If it is high (say more than 10), the speed of the factorization may be increased by allocating more space to the array IW.

INFO(20), INFO(21), and INFO(22) give, on return from MA47B/BD, the number of tile, oxo, and full 2×2 pivots chosen (see Section 4).

INFO(23) gives, on return from MA47B/BD, the number of negative eigenvalues of **A**.

INFO(24) gives, on return from MA47B/BD, the number of zero eigenvalues detected in **A**.

2.3 Error diagnostics

A successful return from MA47A/AD or MA47B/BD 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 MA47C/CD.

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

–1 N < 1 or N > *HUGE*/3 where *HUGE* is the largest possible integer value (MA47A/AD and MA47B/BD).

–2 NE < 1 (MA47A/AD and MA47B/BD).

–3 Failure due to insufficient space allocated to array IW. INFO(2) is set to a value needed to progress beyond the current point of failure (MA47A/AD and MA47B/BD).

–4 Failure due to insufficient space allocated to array A. INFO(2) is set to a value needed to progress beyond the current point of failure (MA47B/BD only).

–5 The indices supplied in KEEP(i), i = 1, 2,..., N when ICNTL(4) = 1 do not constitute a permutation; if KEEP(j) was found to be faulty, j is returned in INFO(2) (MA47A/AD only).

–6 A negative index supplied in KEEP(i), i = 1, 2,..., N when ICNTL(4) = 1 is not immediately followed by another negative index; if KEEP(j) was found to be faulty, j is returned in INFO(2) (MA47A/AD only).

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. Action taken by 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 printed on unit ICNTL(2) (MA47A/AD only).

+2 There are duplicate entries. These will be summed by MA47B/BD. This number is recorded in INFO(4) (MA47A/AD only).

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

+4 The matrix is rank deficient. The number of zero eigenvalues found is given by INFO (8) (MA47A/AD) or INFO(24) (MA47B/BD).

+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 MA47A/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: MA47F/FD, MA47G/GD, MA47H/HD, MA47J/JD, MA47K/KD, MA47L/LD, MA47M/MD, MA47N/ND, MA47O/OD, MA47P/PD, MA47Q/QD, MA47R/RD, MA47S/SD, MA47T/TD, MA47U/UD, MA47V/VD, MA47W/WD, MA47X/XD MA47Y/YD, MA47Z/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, N $\leq$ *HUGE*/3.
NE $\geq$ 1.

4 METHOD

A version of sparse Gaussian elimination is used. It is implemented using a multifrontal method.

MA47A/AD chooses diagonal pivots of orders 1 and 2 using the Markowitz criterion or accepts a sequence of such pivots supplied by the user. Because of the facility for handling matrices with zeros on the diagonal, the 2×2 pivots can be of the form

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

called tile and oxo pivots respectively. MA47A/AD employs a generalized element model of the elimination, thus avoiding the need to store the filled-in pattern explicitly. The elimination is represented as an assembly and elimination tree with the order of elimination determined by a depth-first search of the tree. Opportunities are sought for grouping the variables into blocks where this would not affect the sparsity. This corresponds to merging nodes of the tree. Merges that do affect the sparsity are also performed under the control of ICNTL(6). For most vector and superscalar machines, the best balance is achieved with ICNTL(6) set to its default value of 8.

MA47B/BD factorizes the matrix by using the assembly and elimination ordering generated by MA47A/AD, but with additional numerical pivoting. The numerical pivoting can yield full 2×2 pivots in addition to the tile and oxo pivots above. An option exists for MA47B/BD to use the (full) matrix-matrix multiplication routine SGEMM/DGEMM from the Level 3 BLAS. Because we maintain symmetry in the factorization this can result in more operations than not using this kernel (given by RINFO(2) and RINFO(4)) but may still perform better on vector or parallel machines. The size of the blocks used by SGEMM/DGEMM is given by ICNTL(5).

MA47C/CD uses the factors from MA47B/BD 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, *Factorization of sparse symmetric indefinite matrices* (IMA J. Numer. Anal. **11**, 181-204, 1991). An explanation of the whole algorithm is given by Duff and Reid (1995), *MA47, A Fortran code for direct solution of indefinite sparse symmetric linear systems*, Report RAL-95-001, Rutherford Appleton Laboratory, Oxfordshire.

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 MA47 package
      INTEGER LA, LIW, NEMAX, NMAX
      PARAMETER (LA=50, LIW=50, NEMAX=10, NMAX=5)
      INTEGER IRN(NEMAX),JCN(NEMAX),IW(LIW),KEEP(NEMAX+5*NMAX+2),
     *        IW1(2*NMAX+2)
      INTEGER I,ICNTL(7),INFO(24),N,NE
      DOUBLE PRECISION A(LA),W(NMAX),RHS(NMAX),CNTL(2),RINFO(4)

C Set default parameters
      CALL MA47ID(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 Analyse sparsity pattern
      CALL MA47AD(N,NE,IRN,JCN,IW,LIW,KEEP,ICNTL,RINFO,INFO)

C Factorize matrix
      CALL MA47BD(N,NE,JCN,A,LA,IW,LIW,KEEP,CNTL,ICNTL,IW1,
     *            RINFO,INFO)

C Solve the equations
      CALL MA47CD(N,A,LA,IW,LIW,W,RHS,IW1,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 MA47
      INTEGER  LA, LIW, NEMAX, NMAX
      PARAMETER (LA=200, LIW=200, NMAX=20, NEMAX=100)
      DOUBLE PRECISION CNTL(2),RINFO(4),A(LA),RHS(NMAX),W(4*NMAX)
      INTEGER  I,ICNTL(7),II,INFO(24),IRN(NEMAX),IW(LIW),J,JCN(NEMAX),
     *         KEEP(NEMAX+5*NMAX+2),N,NE,IW1(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 340 I=1,N
        S(I)=EXP(S(I))
 340  CONTINUE
      DO 350 II=1,NE
        I=IRN(II)
        J=JCN(II)
        A(II)=A(II)*S(I)*S(J)
 350  CONTINUE

C     Set default controls
      CALL MA47ID(CNTL,ICNTL)

C Analyse sparsity pattern
      CALL MA47AD(N,NE,IRN,JCN,IW,LIW,KEEP,ICNTL,RINFO,INFO)
C
C Factorize matrix
      CALL MA47BD(N,NE,JCN,A,LA,IW,LIW,KEEP,CNTL,ICNTL,IW1,
     *            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 425 I=1,N
        RHS(I)=RHS(I)*S(I)
 425  CONTINUE

C Solve the equations
      CALL MA47CD(N,A,LA,IW,LIW,W,RHS,IW1,ICNTL)

C     Scale the right-hand side vector by column weights
      DO 475 I=1,N
        RHS(I)=RHS(I)*S(I)
 475  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  0.759064E2   0.12306E4
```

and output would be

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