1 SUMMARY

Solves the **linear least squares** problem by the so-called **normal equations method,** i.e. given an over-determined system of m linear algebraic equations in n unknowns

$$\sum_{j=1}^{n} a_{ij} x_{jl} = b_{il} \qquad i=1,2,...,n \qquad l=1,2,.,k$$

$$m \geq n$$

represented by $\mathbf{AX}=\mathbf{B}$, sets up and solves the system $\mathbf{A}^T\mathbf{AX}=\mathbf{A}^T\mathbf{B}$. The solution so obtained is such that the sums of squares of the equation residuals

$$S_l = \sum_{i=1}^{m} \left\{ \sum_{j=1}^{n} a_{ij} x_{jl} - b_{il} \right\}^2 \quad l=1,2,.,k$$

are minimized.

Cholesky decomposition is used to solve the system. Equations with more than one right-hand side can be solved and the user has options to obtain equation residuals, sum of squares values and the inverse $\mathbf{V}=\{\mathbf{A}^T\mathbf{A}\}^{-1}$ which is usually required for the variance-covariance matrix.

ATTRIBUTES — **Version:** 1.0.0. **Remark:** When there are many unknowns the method is likely to give poor results particularly when applied to fitting polynomials; try `MA14A` or for polynomials `VC01A`. **Types:** `MA09A`; `MA09AD`. **Calls:** `MA08`, and `MA10`. **Original date:** June 1964. **Origin:** M.J.Hopper, Harwell.

2 HOW TO USE THE PACKAGE

2.1 The argument list and calling sequence

The single precision version:

```
CALL MA09A(A,B,V,X,R,M,N,K,IA,IB,IV,IR,IFAIL,IOPT)
```

The double precision version:

```
CALL MA09AD(A,B,V,X,R,M,N,K,IA,IB,IV,IR,IFAIL,IOPT)
```

`A` is a two-dimensional `REAL` (`DOUBLE PRECISION` in the D version) array with dimensions at least m by n (the first dimension specified in `IA`), which must be set by the user to the elements of the matrix **A**, i.e. put a_{ij} i=1,2,...,m j=1,2,...,n into `A(I,J) I=1,M J=1,N`. This argument is not altered by the subroutine.

`B` is a two-dimensional `REAL` (`DOUBLE PRECISION` in the D version) array with dimensions at least m by k (the first dimension specified in `IB`), which must be set by the user to the elements of the right-hand side matrix **B**, i.e. put b_{il} i=1,2,...,m l=1,2,...,k into `B(I,L) I=1,M L=1,K`. This argument is not alter by the subroutine.

`V` is a two-dimensional `REAL` (`DOUBLE PRECISION` in the D version) array of dimensions at least n by n (first dimension specified in `IV`), which is set by the subroutine to the inverse matrix $\mathbf{V}=(\mathbf{A}^T\mathbf{A})^{-1}$ if requested or overwritten as workspace otherwise.

`X` is a two-dimensional `REAL` (`DOUBLE PRECISION` in the D version) array of dimensions at least n by k (first dimension must be `IV` the same as that of `V`), which is set by the subroutine to the elements of the solution matrix **X**, i.e. `X(J,L) J=1,N L=1,K` will be set to x_{jl} j=1,2,...,n l=1,2,...,k.

`R` is a two-dimensional `REAL` (`DOUBLE PRECISION` in the D version) array of dimensions at least m+1 by k (first dimension specified in `IR`), which, if requested, is set by the subroutine to the equation residuals, i.e. `R(I,L)` `I=1,M` `L=1,K` will be set to

$$r_{il} = \sum_{j=1}^{n} a_{ij} x_{jl} - b_{il} \qquad i=1,2,...,m \; l=1,2,...,k$$

and the sums of squares

$$s_l = \sum_{i=1}^{m} r_{il}^2 \qquad l=1,2,...,k$$

will be returned in `R(M+1,L)` `L=1,K.` If residuals are not requested `R` is not used.

`M` is an `INTEGER variable` which must be set by the user to m the number of equations. This argument is not altered by the subroutine. **Restriction:** $m \geq n$.

`N` is an `INTEGER` variable which must be set by the user to n the number of unknowns. This argument is not altered by the subroutine.

`K` is an `INTEGER` variable which must be set by the user to k the number of right-hand sides (number of columns in **B**. This argument is not altered by the subroutine.

`IA` is an `INTEGER` variable which must be set by the user to the first dimension of the array `A.` This argument is not altered by the subroutine.

`IB` is an `INTEGER` variable which must be set by the user to the first dimension of the array `B.` This argument is not altered by the subroutine.

`IV` is an `INTEGER` variable which must be set by the user to the first dimension of the arrays `V` and `X.` This argument is not altered by the subroutine.

`IR` is an `INTEGER` variable which must be set by the user to the first dimension of the array `R.` This argument is not altered by the subroutine.

`IFAIL` is an `INTEGER` variable set by the subroutine to indicate the success or failure of the calculation. The following values are possible

0 successful
1 the matrix $\mathbf{A}^T\mathbf{A}$ was found to be not positive definite, possibly because the rank of the matrix **A** was less than n. A diagnostic is printed and an immediate return to the caller is made.

`IOPT` is an `INTEGER` variable which must be set by the user to select one of the different options offered by the subroutine. The values for `IOPT` and the options are as follows:

1 calculate solution only.
2 solution and residuals (including sums of squares)
3 calculate only the inverse $(\mathbf{A}^T\mathbf{A})^{-1}$.
4 solutions, residuals (including sums of squares) and inverse.

This argument is not altered by the subroutine.

3 GENERAL INFORMATION

Use of Common: none.

Workspace: none.

Other subroutines: calls MA08A/AD, MA10A/AD.

Input/Output: possible diagnostic message.

Restrictions: $m \geq n$.

4 METHOD

The subroutine calls MA08A/AD to construct the normal equations $\mathbf{A}^T\mathbf{AX} = \mathbf{A}^T\mathbf{B}$ of the system $\mathbf{AX} = \mathbf{B}$ and then calls MA10A/AD to solve them using a Cholesky method.