1 SUMMARY

To **solve an over-determined system** of m **linear algebraic equations** in n unknowns in the **minimax** sense, i.e. given equations

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \qquad i=1,2,...,m \quad m > n$$

find the solution x_j j=1,2,...,n such that

$$\max_i \left\{ \left| \sum_{j=1}^{n} a_{ij} x_j - b_i \right| \right\}$$

is minimized.

The problem is posed as an n by m dual linear programming problem which is solved using a special adaptation of the Simplex algorithm.

The subroutine returns residual values and provides an option to print solution details.

The subroutine can be applied to the problem of approximation by linear combinations of general functions over a discrete point set.

ATTRIBUTES — **Version:** 1.0.0. **Types:** `MA11B`; `MA11BD`. **Calls:** `FD05`. **Original date:** September 1965. **Origin:** M.J.Hopper, Harwell.

2 HOW TO USE THE PACKAGE

2.1 The argument list and calling sequence

The single precision version:

```
CALL MA11B(M,N,A,B,IA,X,R,RMAX,IND,IP)
```

The double precision version:

```
CALL MA11BD(M,N,A,B,IA,X,R,RMAX,IND,IP)
```

`M` is an `INTEGER` variable which must be set by the user to m the number of equations. **Restriction:** $m \le 300$.

`N` is an `INTEGER` variable which must be set by the user to n the number of unknowns. **Restriction:** $n \le 100$, $n < m$.

`A` is a two-dimensional `REAL` (`DOUBLE PRECISION` in the D version) array of dimensions at least m by n (first dimension specified in `IA`) which must be set by the user to the equation coefficients a_{ij} i=1,2,...,m, j=1,2,...,n. This argument is not altered by the subroutine.

`B` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length at least m which must be set by the user to the right-hand sides b_i i=1,2,...,m. 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` (needed because the first dimension may be greater than m). This argument is not altered by the subroutine.

`X` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length at least n+2. On return, the first n elements will have been set to the solution x_j j=1,2,...,n.

`R` is a `REAL` (`DOUBLE PRECISION` in the D version) array of length at least m which is set by the subroutine to the final equation residuals

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

`RMAX` is a `REAL` (`DOUBLE PRECISION` in the D version) variable set by the subroutine to the minimax error, $\max_i\{|r_i|\}$.

`IND` is an `INTEGER` array of length at least n+2 set by the subroutine to identify the final minimax reference set. The absolute values of the first n+1 elements in `IND` specify the indices of the equations that make up the reference set. The signs of the elements have been set opposite to the signs of the final reference set residuals and the minimax solution should be that obtained by solving the n+1 equations

$$s_k h + \sum_{j=1}^{n} a_{ij} x_j = b_i \qquad i=|IND(k)| \quad k=1,2,...,n+1$$

where $s_k = -\text{sign}\{IND(k)\}$ and $h \geq 0$ is the minimax error. The subroutine does not solve these explicitly. Normally it arrives at the solution after many Simplex method iterations and the solution may be capable of improvement. This can be done as outlined above using the information returned in `IND.`

`IP` is an `INTEGER` variable which must be set by the user and provides the following print options

1 no printing

2 prints the solution, residuals, minimax error and number of iterations just before returning.

2.2 Error returns

If the rank of the equation matrix $\mathbf{A}=\{a_{ij}\}_{m\times n}$ is less than n, or if the solution is unbounded, the subroutine prints an error diagnostic and returns.

3 GENERAL INFORMATION

Use of Common: none.

Workspace: contains four local arrays which cause the restrictions $n \leq 100$ and $m \leq 300$, and requires 11,706 words of storage.

Other subroutines: calls `FD05.`

Input/Output: a diagnostic message may be printed on the line printer. (Stream 6).

4 METHOD

A linear programming method is used. Given the equations

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \quad i=1,2,...,m \qquad n<m,$$

and for any solution, x_j j=1,2,...,n, there are defined residuals

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

Defining $h = \max_i\{|r_i|\}$ we have the bounds

$$-h \leq \sum_{j=1}^{n} a_{ij} x_j - b_i \leq h \qquad i=1,2,...,m$$

which give the $2m$ inequality constraints

$$\left.\begin{aligned} h+\sum_{j=1}^{n} a_{ij}x_j &\geq b_i \\ h-\sum_{j=1}^{n} a_{ij}x_j &\geq -b_i \end{aligned}\right\} i=1,2,.,m \qquad (1)$$

and further, it is required to find a solution which minimizes h.

Linear programming methods are used for this problem, which is solved by forming the dual of (1) and applying a variant of the Simplex method. From this we obtain the solution x_j j=1,2,...,n, the minimax error h, the residual values r_i i=1,2,...,m and the indices of the n+1 inequalities which form the minimax reference set.

Advantage is taken of the similarity of the two halves of (1) and the extra work space is kept down to an n+2 by n+2 array plus three arrays of lengths m, m and n+2.