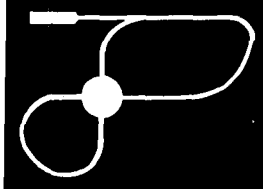
TRRL Supplementary Report 17 UC



TRANSPORT and ROAD RESEARCH LABORATORY
DEPARTMENT of the ENVIRONMENT

S R Loan

MPCODE: a versatile linear and quadratic mathematical programming system

by

A. H. Land, S. Powell, N. J. Paulley and M. R. Wigan

TRANSPORT and ROAD
RESEARCH LABORATORY

Department of the Environment

TRRL SUPPLEMENTARY REPORT 17 UC

MPCODE: A VERSATILE LINEAR AND QUADRATIC MATHEMATICAL PROGRAMMING SYSTEM

by

A. H. Land
(London School of Economics)

S. Powell
(Science Research Council)

N. J. Paulley and M. R. Wigan
(Transport and Road Research Laboratory)

Urban Transport Division
Transport Systems Department
Transport and Road Research Laboratory
Crowthorne, Berkshire
1974

CONTENTS

MPCODE: A VERSATILE LINEAR AND QUADRATIC MATHEMATICAL PROGRAMMING SYSTEM

ABSTRACT

An extensive Mathematical Programming System has been written by A Land and S Powell under SRC funding on CDC 6600 and ICL 1900 computers. TRRL have collaborated with L.S.E. and S.R.C. Atlas Computer Laboratory to develop and test a version of this system in parallel for ICL System 4 and IBM 360 Computers. A high degree of problem solving robustness and reliability has been achieved. Tolerance levels were determined that allow a substantial reduction in core storage required. Examples of linear, parametric, integer, discrete and constrained quadratic formulations are given and the document forms a Users Manual for this system, for both System 4 and IBM 360/370 computers.

1. INTRODUCTION

Scientific utility software for the ICL 4/70 system is not plentiful, due to the limited number of such installations. ICL systems for operational research work are limited to a Transportation Code[1] and the LP 400 Mathematical Programming system[2]. The specialised features of the Transportation Code make it unsuitable for a current TRRL research program on dynamic models of modal choice, but LP 400 has a number of attractive features.

Substantial effort has been required to make effective use of LP 400; and we have had significant help from the ICL subsidiary – Dataskil – in bringing LP 400 into use. From the time that the LP 400 had been handed over to ICL by the contractors who wrote it (Scicon), relatively few UK installations have made full use of the system. The report writing, input and problem formulation facilities are restricted, due partly to the restrictive environment provided by the host ICL system softwear, but the actual algorithms are efficient and LP 400 provides considerable problem solving power. We have carried out a systematic survey of tests on LP 400 under both J and Multijob operating systems, and have made extensive use of the MPCODE system to detect and diagnose erroneous solutions produced by LP 400. The linear programming part of LP 400 is now in regular production use for Research Programme Portfolio analysis within TRRL, and the system is demonstrably useful for linear, mixed integer, parametric and separable problem formulations. Unfortunately, the structure of LP 400 and its supporting documentation is such that a substantial knowledge of the mathematical field of convex programming is needed to exploit LP 400 properly. The level of generality at which LP 400 can be used requires a multiple stage process to set up first a suitably tailored subset of LP 400 for the class of formulation described, and then to apply this to the whole range of specific problem formulations of this type. This is a very considerable task for occasional users without a substantial background knowledge of operational research and systems analysis. It should be pointed out that this approach is common to almost every mathematical programming system, and does allow specific problem formulations to be switched from one such system to another with a minimum of effort.

For research or teaching purposes a less ponderous tool is very desirable, and an appropriate approach is to set up a FORTRAN, ALGOL or PL/1 program in a reasonably machine-independent manner and include a quadratic programming algorithm and other such powerful analytical methods that would not normally form a part of a large scale LP-oriented systems such as LP 400. One such system had been written and published[3] by A. H. Land and S. Powell with SRC support. The published version is in a form directly suitable for the CDC 6600 computer. We therefore transferred it to the TRRL ICL 4/70, and put it through the necessary series of benchmark tests to establish reliable problem solving performance. The CDC 6600 has a 60 bit word, and effectively

unlimited storage from the programmer's viewpoint. The ICL System 4 and IBM/360-370 computers both use 32 bit words addressed as 8 bit bytes, and in general (up to the release of the VS series of IBM operating systems) offer less core storage to the programmer. Parallel development and testing was carried out on the IBM 370 at SRC Chilton and on the ICL 4/70 at TRRL Crowthorne to produce a reliable system and establish tolerance values suitable for the IBM/ICL byte/word structures when the usage of double precision storage was cut to a minimum. The end result of this work was to produce a FORTRAN code which ran equally accurately on 4/70 and 370. The basic system has also been run on ICL 1900 and Univac 1108 machines.

The code written for the CDC 6600 and published in Land and Powell's book[3] differs from this 4/70, 370 version only in tolerance values and the use of single precision and double precision declarations for appropriate variables. For the single precision version of the code we are now satisfied after extensive testing and parallel running that the tolerance values quoted here ensure robust and accurate problem solving. The system has already been used in current TRRL research on dynamic models of modal choice and public transport competition, and in the analysis of turnover capacities of car parks and further applications now in hand.

2. SYSTEM SPECIFICATION

MPCODE is a suite of programs which will perform linear, parametric, integer, discrete and quadratic operations on mathematical programming formulations. In writing this guide it has been assumed that the user has some minimal familiarity with mathematical programming ideas, though not necessarily any deep knowledge. No attempt will be made here to deal with theory in depth; the user is recommended to consult one of the many excellent textbooks on the subject – for example, Reference 5. Mathematical formulations for these problems take the general form:-

maximise $z(\underline{x})$

subject to $A\underline{x} \leqslant \underline{b}$

$\underline{x} \geqslant \underline{o}$

In the linear programming problem $z(\underline{x}) \equiv \underline{c}'(\underline{x})$, with dual problem

minimise $\underline{y}'\ \underline{b}''$

subject to $\underline{y}'A \geqslant \underline{c}'$, $\underline{y} \geqslant \underline{o}$

In the integer (linear) programming problem, some or all of the variables, $\underline{x}$, are also limited to take integer values.

In the quadratic programming problem, $z(\underline{x}) = \underline{p}'\underline{x} + \tfrac{1}{2}\underline{x}'D\underline{x}$

Here A is an (m x n) matrix of coefficients of constraints, $\underline{b}$ is a (m x 1) vector of the right-hand-sides of the constraints and $\underline{c}$ or $\underline{p}$ are (n x 1) vectors of coefficients of the objective or function row. $\underline{y}$ is a (m x 1) vector and $\underline{x}$ is a (n x 1) vector. D is a (symmetric) matrix of coefficients of the quadratic form of the quadratic programming objective row. " ' " indicates transposition.

MPCODE will solve problems of this class up to a limiting size. The maximum size may be altered to take advantage of the local core resources (see Appendix 2). The system is central-processor (CPU) dominated with input and output being limited to card reader and line printer, except when the BB restart facility is being used. This does restrict the size of problem which may be attacked, but it also makes the program remarkably versatile and efficient in CPU time, and it may be favourably compared with commercial systems. MPCODE originally consisted of a number of distinct programs, each of which performed one of the operations mentioned above.

However, as these programs were essentially modular in form, having a large number of common subroutines, it was practical to combine these programs into one system to allow the operational method (linear programming, integer programming, etc.) to be chosen by a driver routine at run-time. This has been carried out at TRRL. This system naturally remains modular and appropriate use of overlays reduces the core storage required for the program – see Appendix 2. This modular construction is a source of much of the program's versatility.

The original program has been tested principally on the London University CDC 6600. The system employed at TRRL is the same except that the values of certain constants have been altered (after considerable testing) to take account of the difference in word lengths between the CDC 6600 and the ICL System 4 and IBM machines.. (60 bit words on the CDC 6600, 32 bit on System 4/IBM 360.) It is, therefore, now possible to keep to single precision variables thus enabling larger problems to be run. The constants that required adjustment were the tolerances. As Land and Powell state[3] "one of the critical decisions in designing LP systems is the tolerance to be used – 'when is a zero not a zero?'" Tolerances have been found for the 32 bit word machines which are robust under all known circumstances, and every attempt has been made to establish this for all the different algorithms available in MPCODE.

The program is based on the 'reduced inverse matrix'[3] version of the Simplex Method. The controlling routine for this method is a subroutine, DOANLP (see Fig. 2), which issues calls to a number of other subroutines which perform the algorithmic functions. The linear programming option (LINP) calls DOANLP, seeks – and if possible obtains – a feasible and subsequently an 'optimal solution', checks for accuracy then if necessary reinverts the reduced matrix and enters DOANLP again to reoptimise. The integer programming option (MIF) requires all coefficients and solution values to be integer: a version of Gomory's Method of Integer Forms is used, and the program also uses DOANLP[3] amongst other routines. For the discrete programming option (BB), where a subset (possibly total) of the variables are restricted to alter by specified step sizes, a Branch and Bound algorithm[3,5] is used once again with DOANLP as the basis. Parametric Linear Programming (PLP) carries out parametric analysis on the right-hand side ($\underline{b}$ vector) and objective function ($\underline{c}$ vector) elements, as an extension to the basic LP. The Quadratic Programming option (QP) employs a variant of Beale's quadratic programming algorithm[4], and requires a routine DOAQP analagous to DOANLP. For QP all the constraints must be linear in form, variables may not be constrained to be integer, but the objective function can be a quadratic expression.

The dual solution is obtained from the LP option, though interpretation of the meaning of the dual is, of course, up to the user (Reference 5). More detailed descriptions of the various options are presented in Section 4, but for full descriptions see Land and Powell[3] or a general reference book such as Vajda[5].

Major options are:-

Linear Programming (LINP)
Integer Programming (MIF)
Quadratic Programming (QP)
Discrete Programming (BB)
Parametric Linear Programming (PLP)

3. GENERAL OPERATING INSTRUCTIONS

Given that there is a problem which may be expressed in mathematical programming terms, the first decision to be made is which method of optimisation is to be used – i.e., is it to be LINP, MIF, etc. (see Section 4)? The user specifies this on data card, read in by the driver routine, by means of a keyword which causes a branch to the controlling subprogram for the chosen method. The number of problems to be served with separate data for each run (see below), and optional comments to document the printout are also read in at this stage.

Once the program has selected the chosen method it reads in the data specifying the problem. As matrices and vectors are packed by the program, only non-zero coefficients need be punched. All the data should be right-adjusted in the appropriate fields, and should be integer in form with the exception of the elements of the objective vector (C), the upper bound vector (BOUND), the right-hand-side vector (B) and matrix (A) which may be real and hence include a decimal point; and the variable TYPE of PLP which is character in format (see Section 4). Some scaling of the problem may be required due to the implicit restrictions on the size of values which may be read by the limited field lengths allocated on the input cards. The setting of the tolerances is such that the programs work best if the problem is scaled to bring all the coefficients in the A-matrix as close to ± 1 as can be achieved without too much trouble. Note also the more stringent scale requirements for BB and the integer requirements for MIF.

It is possible to run more than one problem within a single job. There are two ways of accomplishing this:

(i) Set the required number of problems (variable NRUN) as input to the driver routine at the beginning of the run; the program will then return to the driver routine as each separate problem is solved.

(ii) If the same method of optimisation is to be used on consecutive problems, the variable MORE, read in at run-time, may be set to indicate there are more problems requiring the same method which will follow on. Data for the second and subsequent problems are submitted directly after that for the first, as MORE inhibits a return to the driver routine.

Naturally both methods may be employed in the same job, but it is important to remember that NRUN controls the number of branches from the driver program and not necessarily the number of distinct options, which may be larger than NRUN if the MORE option is also used.

Data input formats and job control statements for the ICL System 4 series machines are specified in Appendices 1 and 3.

MPCODE traps errors and prints diagnostic messages (generally self-explanatory) for most of the data errors and abnormal conditions that are likely to occur; the amount of the data input copied out to the printer and the level of detail in the printout of the solutions are controlled by the user by the appropriate input value of variable MOREPR (see Appendix 1). These debugging aids are particularly important for mathematical programming where both problem infeasibility and unboundedness frequently occur. The user can control the maximum number of iterations and re-inversions allowed. If computer time is more valuable than the accuracy of a solution, partial (and therefore suboptimal) solutions can be obtained more quickly than an accurate result by restricting the number of iterations and reinversions.

The data input formats specified in Appendix 1: Blocks B to F are the same as those of Land and Powell[3]. This manual therefore serves as a users' guide to the system as described and listed in Reference 3.

4. SPECIFIC OPERATING INSTRUCTIONS

This section contains detailed guidance in the use of each of the options available in MPCODE. For each method a complete example is provided. First the formulation, then the data input are specified for each such example.

The following standard notation will be used throughout.

$\underline{x}$ or X is the vector of primal variables

$\underline{y}$ or Y is the vector of dual variables

$\underline{c}$ or C is the vector of coefficients of the function row

$\underline{b}$ or B or RHS is the vector of coefficients of the right-hand-side of the constraints.

A is the matrix of coefficients of the inequalities, so the expression $A\underline{x} \leqslant \underline{b}$ is a series of linear inequalities. The slack variables are the dummy variables added to each inequality constraint to make it an equality constraint (i.e. the slack in the row $\sum a_{ij}x_j \leqslant b_i$ is the variable x_{m+i} where $\sum a_{ij}x_j + x_{m+i} = b_i$).

It is conventional to designate the inequalities of general constraints in maximisation algorithms as 'less than-or-equal'. For particular rows these inequalities may of course be 'equal' or 'greater-than-or equal', in which case the corresponding dual variables will be free variables for equality constraints and non-positive for 'greater-than-or-equal' constraints

4.1 Linear Programming (LINP)

This program option deals with the ordinary type of LP problem where all variables and their duals take real values. Such problems have the general form

$$\begin{aligned} &\text{maximise} && \underline{c}\,'\,\underline{x} \\ &\text{subject to} && A\underline{x} \leqslant \underline{b} \\ & && \underline{x} \geqslant \underline{o} \end{aligned}$$

where $\underline{x}$ is not constrained to be integer but may have upper bounds (i.e., $o \leqslant x_j \leqslant \bar{x}_j$ for some j). The inequalities (*) may be '$\geqslant$' or '$=$' for particular rows of the series of inequalities.

As an illustration of LINP, the following problem is solved.

$$\begin{aligned} &\text{maximise} && x_1 + 3x_2 + 10x_3 \\ &\text{subject to} && 12x_1 + 5x_2 + 30x_3 \leqslant 120 \\ & && 2x_1 + 10x_2 + 30x_3 \leqslant 95 \\ & && x_1 \geqslant 0 \\ & && x_2 \geqslant 0 \\ & && 0 \leqslant x_3 \leqslant 2 \end{aligned}$$

so that $\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$, $\underline{c} = \begin{bmatrix} 1 \\ 3 \\ 10 \end{bmatrix}$, $\underline{b} = \begin{bmatrix} 120 \\ 95 \end{bmatrix}$

and A $= \begin{bmatrix} 12 & 5 & 30 \\ 2 & 10 & 30 \end{bmatrix}$

BOUND, the bound vector, is

$$\begin{bmatrix} -1 \\ -1 \\ +2 \end{bmatrix}$$

where a BOUND of -1 indicates a variable is unbounded

Data input and solution output are presented in Tables 1 and 6. Solution output is discussed in Section 5.1 For data input format see Appendix 1, Block B.

4.2 Integer Programming (MIF)

There are many types of problem where it is unrealistic for the variables to be non-integer. For example, problems where the variables refer to people or whole commodities. Integer Programming is designed for such LP problems, where *all* variables and coefficients are required to be integer. (For cases where only some of the variables must be integer see Discrete Programming, Section 4.4.) The method of Integer Forms adds 'cutting

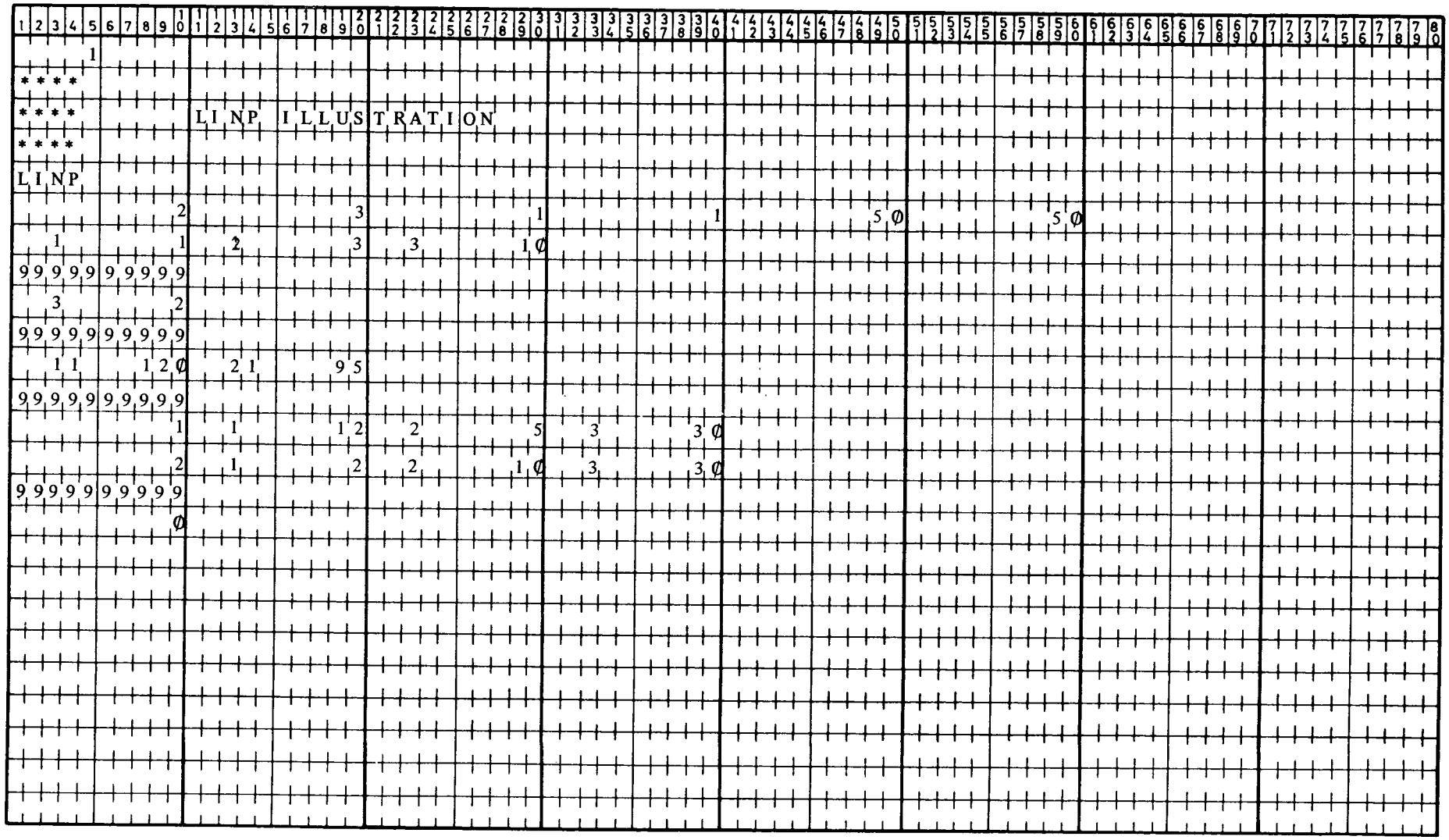


TABLE 1

Linear programming: example data input

```
                                        ****
                                        ****      LINP  ILLUSTRATION RUN
                                        ****

                                                        L.P. WITH ACCURACY CHECKS AND REINVERSIONS

MOREPR.=     1

IRMAX =     50

ITRMAX =     50


M (NO. OF CONSTRAINTS) =      2, N (NO. OF VARIABLES=-REAL, NOT SLACK) =      3,
NUMBER OF UPPER BOUNDED VARIABLES =      1.

INPUT CARDS FOR THE C ELEMENTS . . . .
(  1     1,000)(  2      3,000)(  3     10,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(999999999,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)

INPUT CARDS FOR THE UPPER BOUNDS ON SINGLE VARIABLES . . . .
(  3     2,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(999999999,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)

INPUT CARDS FOR THE B VECTOR, THE RIGHT-HAND SIDES OF THE CONSTRAINTS . . . .
(  11   120,00)(  21    95,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)
(9999999999,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)(  00     0,00)

INPUT CARDS FOR THE ROWS OF THE A MATRIX . . . .
(  0     1,000)(  1     12,000)(  2      5,000)(  3     30,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(  0     2,000)(  1      2,000)(  2     10,000)(  3     30,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(999999999,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)

OBJECTIVE            32,04544100

                J. . .  1          2          3

           BOUND VECTOR. . . .
                 =1,0000    =1,0000     2,0000

             X VECTOR. . . .
                 3,8636     2,7273     2,0000
 I   Y VECTOR                                          B        B=AX
                                                               (SLACK)
                (                                  )
 1     0,0364 (   12,0000     5,0000    30,0000  ) LE   120,00     0,0000
                (                                  )
 2     0,2818 (    2,0000    10,0000    30,0000  ) LE    95,00     0,0000
                (                                  )

             C VECTOR. . . .
                 1,0000     3,0000    10,0000

               Y'A=C. . . .
                 =0,0000    =0,0000    =0,4546


             COLUMN         1          2
             YBASIS         2          1

                YR           0,2818     0,0364

ROW XBS      XR      INVERSE MATRIX

 1   2       2,7273      0,1091     =0,0182
 2   1       3,8636     =0,0455      0,0909




   BIG  0,1000E 09, DRIVER          0,0,  INREV           0,     IR           0,  IRMAX          50,  ISBND           1
I$DONE           0, ISTATE           1,    ITR           4, ITRMAX          50,     M           2,  MARKI           0
 MARKK           0,   MAXA        1000,   MAXM          25,   MAXN          50,  MORE           0, MX$IZE          50
     N           3, NEGINV           0, NEGROW           0,   NEWX           0,  NEWY           2, NUMSLK           0
     R     3,86364,   SIZE           2,   SMALL  0,1000E=06, TOL(1)  0,1000E=02, TOL(2)  0,1000E=02, TOL(3)  0,1000E=02
TOL(4)  0,1000E=02, TOL(5)  0,1000E=05, TOL(6)  0,1000E=02, TOL(7)  0,1000E=02, TOL(8)  0,1000E=04,  XKPOS         1,0
YAMINC    =0,00100

ISEFF
  2  1

INBASE
  2  1 =1

    4 SIMPLEX ITERATIONS.

OPTIMUM
```

TABLE 6

planes' to the original problem to find the optimum integer solution. This algorithm is appropriate only for problems with very small (integer) coefficients in the A matrix. It may fail to reach an integer solution within the maximum iteration count or because the program can no longer find satisfactory additional cutting planes. If it fails, the branch and bound algorithm should be used, as this will almost certainly at least produce a feasible integer answer.

MIF requires the appropriate input data (i.e., A, B and BOUND) in integer form. Failure to specify integer coefficients in the data input produces an error message and program termination at the data input stage.

The integer programming problem used to illustrate the MIF option has the same form as that used for LINP with the additional constraint that integer solution values are required. For data input format see Appendix 1: Block C, and Tables 2 and 7 show data input and solution output to the example problem. For a discussion of the output produced see Section 5.

4.3 Quadratic Programming (QP)

The QP option is an appropriate algorithm when both linear (real) constraints and a quadratic objective function are required, for instance when the maximand contains products of two problem variables, such as price times quantity. It has also been used in econometric problems to obtain regression coefficients that are subject to linear constraints. The problems take the form:

maximise: $\underline{p}'\underline{x} + \frac{1}{2}\underline{x}'D\underline{x}$

subject to the usual constraints

$$A\underline{x} \leqslant \underline{b}$$
$$\underline{x} \geqslant \underline{o}$$

and $\underline{x}$ may have upper bounds $\underline{p}$ is the vector of coefficients of the linear part of the objective function, and D is the matrix of the quadratic form.

MPCODE uses the basic LP algorithm to solve this type of problem as at least a *local* optimum may be found if the problem

maximise: $\underline{q}'\underline{x}$

subject to: $A\underline{x} \leqslant \underline{b}$

$$\underline{x} \geqslant \underline{o}$$

is maximised at the point $\underline{x}_o$ where $\underline{q} = \underline{p} + \underline{x}'_o D$. This optimum will also be a *global* optimum if D is negative definite or negative semi-definite, or more generally if the objective function is concave within the feasible region. For instance, a quadratic function such as an hyperbola will yield a global optimum, so long as the saddle-point of the function lies outside the feasible region.

The algorithm is not affected by the condition of the D matrix; even a singular matrix D is acceptable.

As an illustration consider the problem below. It would, of course, be more efficient to divide the rows by 10^3 before solving. However, the large coefficients are retained to provide a test for the tolerance settings.

Maximise

$$-853x_1 + 994x_2 - 879x_3 + 991x_4 - 907x_5 + 989x_6 - 936x_7$$
$$+ 987x_8 - 961x_9 + 984x_{10} - 987x_{11} + 982x_{12}$$
$$+ 1x_2x_8 + 1x_5x_{11} + 1x_8x_{14} + 1x_{11}x_{17}$$

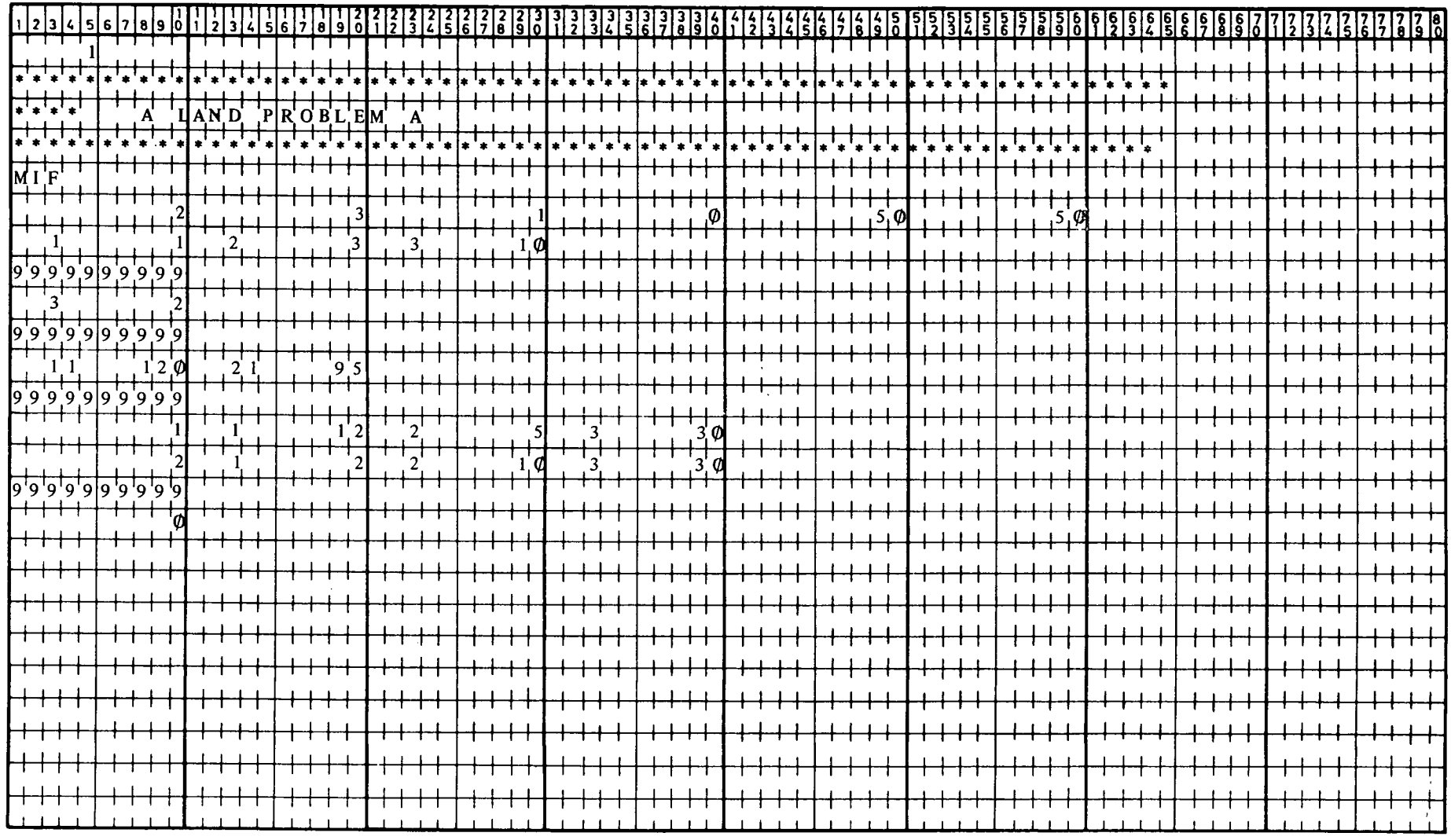


TABLE 2

Integer programming: example data input

```
**********************************************************************

****    A LAND PROBLEM A

**********************************************************************

                                METHOD OF INTEGER FORMS

IRMAX =    20

ITRMAX =   100

M (NO. OF CONSTRAINTS) =     2, N (NO. OF VARIABLES--REAL, NOT SLACK) =     3,
NUMBER OF UPPER BOUNDED VARIABLES =     1.

INPUT CARDS FOR THE C ELEMENTS . . . .
(  1      1,000)(  2      3,000)(  3     10,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(999999999,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)

INPUT CARDS FOR THE UPPER BOUNDS ON SINGLE VARIABLES . . . .
(  3      2,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(999999999,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)

INPUT CARDS FOR THE B VECTOR, THE RIGHT-HAND SIDES OF THE CONSTRAINTS . . . .
(  11    120,00)(  21     95,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)
(9999999999,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)(  00      0,00)

INPUT CARDS FOR THE ROWS OF THE A MATRIX . . . .
(  0      1,000)(  1     12,000)(  2      5,000)(  3     30,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(  0      2,000)(  1      2,000)(  2     10,000)(  3     30,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)
(999999999,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)(  0      0,000)

OBJECTIVE            31,00006100

              J. . .  1          2          3

         BOUND VECTOR. . . .
              =1,0000    =1,0000     2,0000

           X VECTOR. . . .
              2,0000     3,0000     2,0000
 I    Y VECTOR                                          B        B=AX
                                                                (SLACK)
              (                                    )
 1     0,0000 (   12,0000     5,0000    30,0000  ) LE   120,00   21,0002
              (                                    )
 2     0,0000 (    2,0000    10,0000    30,0000  ) LE    95,00    0,9999
              (                                    )
 3     0,0000 (    2,0000     8,0000    25,0000  ) LE    78,00    0,0000
              (                                    )
 4     0,2500 (    4,0000    12,0000    40,0000  ) LE   124,00    0,0000
              (                                    )

           C VECTOR. . . .
              1,0000     3,0000    10,0000

          Y'A=C. . . .
             =0,0000    =0,0001     0,0000

ISEFF
  0  0  1  2

INBASE
  2  1 =1

   27 SIMPLEX ITERATIONS.

INTEGER PROGRAM OPTIMUM.
```

TABLE 7

subject to

$$
\begin{aligned}
1000x_1 + 1000x_2 + 1000x_3 &= 7000 \\
1000x_4 + 1000x_5 + 1000x_6 &= 7000 \\
1000x_7 + 1000x_8 + 1000x_9 &= 7000 \\
1000x_{10} + 1000x_{11} + 1000x_{12} &= 7000 \\
1000x_{13} + 1000x_{14} + 1000x_{15} &= 7000 \quad \text{and } \underline{x} \geqslant \underline{o} \\
1000x_{16} + 1000x_{17} + 1000x_{18} &= 7000 \\
2000x_1 + 3000x_2 &= 14000 \\
2000x_4 + 3000x_5 &= 14000 \\
1000x_7 + 5000x_8 &= 14000 \\
2000x_{10} + 3000x_{11} &= 14000 \\
2000x_{13} + 3000x_{14} &= 14000 \\
2000x_{16} + 3000x_{17} &= 14000
\end{aligned}
$$

Data input format is specified in Appendix 1, Block D. Illustrations of data input and solution output are found in Tables 3 and 8; the output is discussed in Section 5.

4.4 Discrete Programming (BB)

BB is a Branch and Bound algorithm designed to solve problems which may be expressed in the usual LP form

$$
\begin{aligned}
&\text{Maximise} && \underline{c}' \underline{x} \\
&\text{Subject to} && A\underline{x} \leqslant \underline{b} \\
& && \underline{x} \geqslant \underline{o}
\end{aligned}
$$

and $\underline{x}$ may have upper bounds. There is, however, the additional requirement that some or all of the components of $\underline{x}$ take discrete values. Different step sizes may be allocated to different variables, but each discrete variable may have only one step size.

A common application of discrete programming is mixed integer programming where some or all of the variables are constrained to take integer values. This is equivalent to saying that some or all of the variables take discrete integer values with a step size of one.

As 'Branch and Bound' is a tree search algorithm[5] a feasible solution can be obtained by ending the run before the search is complete. Very frequently, the optimum solution is obtained (and printed) early in the search, but a very large amount of computing is required to prove that no better solution exists. This characteristic of the method is useful if computer time is limited; hence BB is provided with a restart facility. On the first run of a problem, the choice of a value for ITRBBM (the maximum number of iterations) is often little better than a guess. The computation will finish when the number of iterations, ITRBB, is greater than ITRBBM, and this may well occur before the Branch-and-Bound tree has been completely explored. The restart facility allows the computation to be started in a later run from the point at which the previous run ended.

If a solution is not found within the maximum number of iterations, a representation of the tree is saved on tape, disc, or card punch on logical unit 07. A description of the saved part of the tree is also printed out. In any subsequent computer run this data is read in at the beginning from tape, disc or card punch on logical unit 08, and a description of this data is printed. Apart from the printing on input and output there is no difference in the course of the computation between invoking the restart procedure and ignoring it.

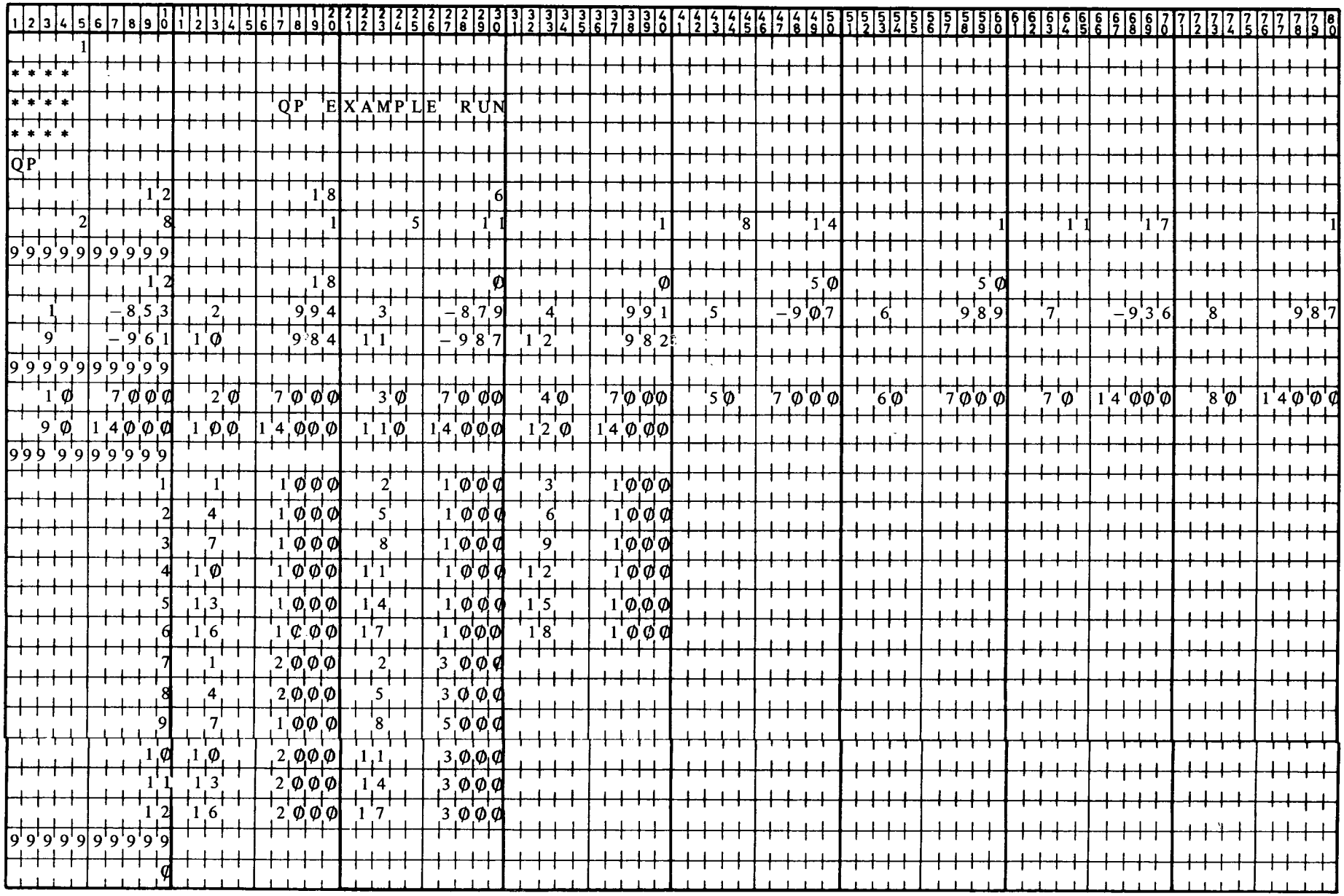


TABLE 3

Quadratic programming: example data input

```
                                        ****
                                        ****            QP  EXAMPLE RUN
                                        ****

                                                                        QUADRATIC PROGRAMMING

M =  12, N =  18, NUMBER OF VARIABLES WITH NON-LINEAR COEFFICIENTS =      6.

CARD INPUT FOR THE NON-LINEAR COEFFICIENTS OF THE FUNCTION. . . .
(     2)(     8)(    1.00)  (     5)(    11)(    1.00)  (     8)(    14)(    1.00)  (    11)(    17)(    1.00)
(99999)(99999)(    0.00)  (     0)(     0)(    0.00)  (     0)(     0)(    0.00)  (     0)(     0)(    0.00)

IRMAX =     50

ITRMAX =     50

M (NO. OF CONSTRAINTS) =     12, N (NO. OF VARIABLES--REAL, NOT SLACK) =     18,
NUMBER OF UPPER BOUNDED VARIABLES =      0.

INPUT CARDS FOR THE C ELEMENTS . . . .
(  1  -853.000)(  2    994.000)(  3  -879.000)(  4    991.000)(  5  -907.000)(  6    989.000)(  7  -936.000)(  8    987.000)
(  9  -961.000)( 10    984.000)( 11  -987.000)( 12    982.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(9999999999.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)

INPUT CARDS FOR THE B VECTOR, THE RIGHT-HAND SIDES OF THE CONSTRAINTS . . . .
(  10  7000.00)(  20  7000.00)(  30  7000.00)(  40  7000.00)(  50  7000.00)(  60  7000.00)(  70 14000.00)(  80 14000.00)
(  90 14000.00)( 100 14000.00)( 110 14000.00)( 120 14000.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)
(9999999999.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)

INPUT CARDS FOR THE ROWS OF THE A MATRIX . . . .
(  0     1.000)(  1  1000.000)(  2  1000.000)(  3  1000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     2.000)(  4  1000.000)(  5  1000.000)(  6  1000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     3.000)(  7  1000.000)(  8  1000.000)(  9  1000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     4.000)( 10  1000.000)( 11  1000.000)( 12  1000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     5.000)( 13  1000.000)( 14  1000.000)( 15  1000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     6.000)( 16  1000.000)( 17  1000.000)( 18  1000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     7.000)(  1  2000.000)(  2  3000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     8.000)(  4  2000.000)(  5  3000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     9.000)(  7  2000.000)(  8  3000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0    10.000)( 10  2000.000)( 11  3000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0    11.000)( 13  2000.000)( 14  3000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0    12.000)( 16  2000.000)( 17  3000.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(999999999.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)

P VECTOR . . . .

     1         2         3         4         5         6         7         8         9        10        11        12
  -853.00    994.00   -879.00    991.00   -907.00    989.00   -936.00    987.00   -961.00    984.00   -987.00    982.00

    13        14        15        16        17        18
   0.00      0.00      0.00      0.00      0.00      0.00

          THE D MATRIX OF NON-LINEAR COEFFICIENTS. . . .

          COLUMN  1         2         3         4         5         6
ROW    VARIABLE  2         8         5        11        14        17
  1        2     0.0000    1.0000    0.0000    0.0000    0.0000    0.0000
  2        8     1.0000    0.0000    0.0000    0.0000    1.0000    0.0000
  3        5     0.0000    0.0000    0.0000    1.0000    0.0000    0.0000
  4       11     0.0000    0.0000    1.0000    0.0000    0.0000    1.0000
  5       14     0.0000    1.0000    0.0000    0.0000    0.0000    0.0000
  6       17     0.0000    0.0000    0.0000    1.0000    0.0000    0.0000

IN THE FOLLOWING OUTPUT, THE C VECTOR CONTAINS THE PARTIAL DERIVATIVES OF THE QUADRATIC FUNCTION.  THERE MAY BE
MORE CONSTRAINTS PRESENT THAN THE M SPECIFIED ORIGINALLY, BUT THEY HAVE ZERO DUAL VALUES.
ALL VARIABLES HAVE BEEN ASSUMED TO HAVE AN UPPER BOUND OF 10E10 UNLESS OTHERWISE SPECIFIED.
```

TABLE 8

```
SINCE N EXCEEDS 7, AN ABBREVIATED FORM OF PRINTING FOLLOWS
THE SIGN(I) VECTOR INDICATES THE SIGN OF THE I-TH CONSTRAINT, 0 FOR EQ, 1 FOR LE, -1 FOR GE,

NON-ZERO ELEMENTS OF THE A MATRIX, FOLLOWED BY THEIR COLUMN LABELS....

       1         2         3         4         5         6         7         8         9        10        11        12
1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  1000,000
       1         2         3         4         5         6         7         8         9        10        11        12

      13        14        15        16        17        18        19        20        21        22        23        24
1000,000  1000,000  1000,000  1000,000  1000,000  1000,000  2000,000  3000,000  2000,000  3000,000  1000,000  5000,000
      13        14        15        16        17        18         1         2         4         5         7         8

      25        26        27        28        29        30        31
2000,000  3000,000  2000,000  3000,000  2000,000  3000,000     0,000
      10        11        13        14        16        17         1

THE FOLLOWING VECTORS SHOW THE STARTING POINTS OF THE SUCCESSIVE ROWS OF A IN THE ABOVE LIST OF THE NON-ZERO ELEMENTS,'..
   1   2   3   4   5   6   7   8   9  10  11  12  13
   1   4   7  10  13  16  19  21  23  25  27  29  31

OBJECTIVE          15166,19500000

                                    J       1         2         3         4         5         6         7         8
                            C VECTOR    -853,0     996,8    -879,0     991,0    -907,0     989,0    -936,0     996,3
                        BOUND VECTOR ********* ********* ********* ********* ********* ********* ********* *********
                            X VECTOR    0,0000    4,6667    2,3333    7,0000    0,0000    0,0000    0,0000    2,8000
                              Y'A-C     1224,53     0,00      0,00      0,00   1899,00    -0,00    366,47    -0,00

       9        10        11        12        13        14        15        16        17        18
  -961,0     984,0    -982,3     982,0       0,0       2,8       0,0       0,0       0,0       0,0
********* ********* ********* ********* ********* ********* ********* ********* ********* *********
   4,2000    7,0000    0,0000    0,0000    0,0000    4,6667    2,3333    0,0000    4,6667    2,3333
     0,00      0,00   1967,33    -0,00      1,87    -0,00      0,00      0,00      0,00      0,00

                                    I       1         2         3         4         5         6         7         8
                            B VECTOR    7000,0    7000,0    7000,0    7000,0    7000,0    7000,0   14000,0   14000,0
                                SIGN        0.        0.        0.        0.        0.        0.        0.        0.
                            Y VECTOR    -0,8790    0,9890   -0,9610    0,9820    0,0000    0,0000    0,6233    0,0010
                                B-AX    0,0000    0,0000    0,0000    0,0000    0,0000    0,0000    0,0000    0,0000

       9        10        11        12        13
 14000,0   14000,0   14000,0   14000,0     982,0
      0,        0.        0.        0.        1.
   0,3915    0,0010    0,0009    0,0000    0,0000
   0,0000    0,0000    0,0000    0,0000  982,0000

ISEFF
  1 11  8 12  9 10  7  2  3  4  5  6  0

INBASE
  0  1  7  2  0 11  0  3  8  4  0 12  0  5  9  0  6 10

   13 SIMPLEX ITERATIONS.

QUADRATIC PROGRAM OPTIMUM.
```

TABLE 8 (continued)

The restart facility is employed, and an attempt will be made to write to logical unit 07 if the problem has not been solved and

$$\text{ITRBBM} - \text{ITRBB} < \text{MNOW}$$

where: MNOW is the number of constraints

ITRBBM is the maximum number of BB iterations

ITRBB is the current number of BB iterations

Note: An attempt will be made to write to unit 07 if these conditions are satisfied.whether or not provision has been made for output to unit 07. If no provision has been made and the restart facility is invoked, the program will fail.

The BB restart facility included in MPCODE uses subroutines with unformatted input/output for the saved data items. This facility is, therefore, only appropriate in MPCODE for use with disc or tape units allocated to logic units 07 and 08. Users wishing to use a card punch on units 07 and 08 will need formatted read and write statements necessitating minor changes in code. For details of this variation of the code, the alterations required, and a full description of the relevant sub-routines see Reference 3, Chapter 5.

To demonstrate the use of the BB option consider the following problem

$$\begin{aligned} \text{Maximise} \quad & 2x_1 + 3x_2 + 10x_3 \\ \text{Subject to} \quad & 24x_1 + 5x_2 + 30x_3 \geqslant 1200 \\ & 4x_1 + 10x_2 + 30x_3 \geqslant 950 \end{aligned}$$

and $x_1 \geqslant 0$

$x_2 \geqslant 0$

$0 \leqslant x_3 \leqslant 20$

with step sizes of 5 for x_1, 10 for x_2 and x_3.

BB is particularly sensitive to the requirement that A coefficients should be kept within a range, since there is a risk of spurious infeasibility occurring and hence large parts of the tree not being examined. The A matrix coefficients for BB should be kept within ± 0.001 and ± 10.0 particularly for large problems. Note that this may necessitate setting a large step size for some variables.

Data input format for BB is described in Block E, Appendix 1. Illustrations of data input and solution output appear in Tables 4 and 9; the output is discussed in Para. 5.

4.5 Parametric Linear Programming (PLP)

This option allows the user to vary various elements of the problem after an initial optimal LP solution has been found and to discover how this optimal solution changes under these variations. Parametric programming is a form of post-optimal analysis which is a very important tool to the LP user. It allows him to examine the stability of a solution when certain parts of the data are varied without having to re-run the complete analysis: it is, therefore, possible to discover a great deal about the nature of the solution and the problem itself in a single run.

Parametric programming is particularly useful if solutions are required to a problem which has a range of values for RHS or objective function elements. There are many formulations which require this type of approach. The algorithm is designed to vary only single elements in the RHS and objective function, but a reformulation of the problem (see 3) enables the whole RHS or objective function to be varied.

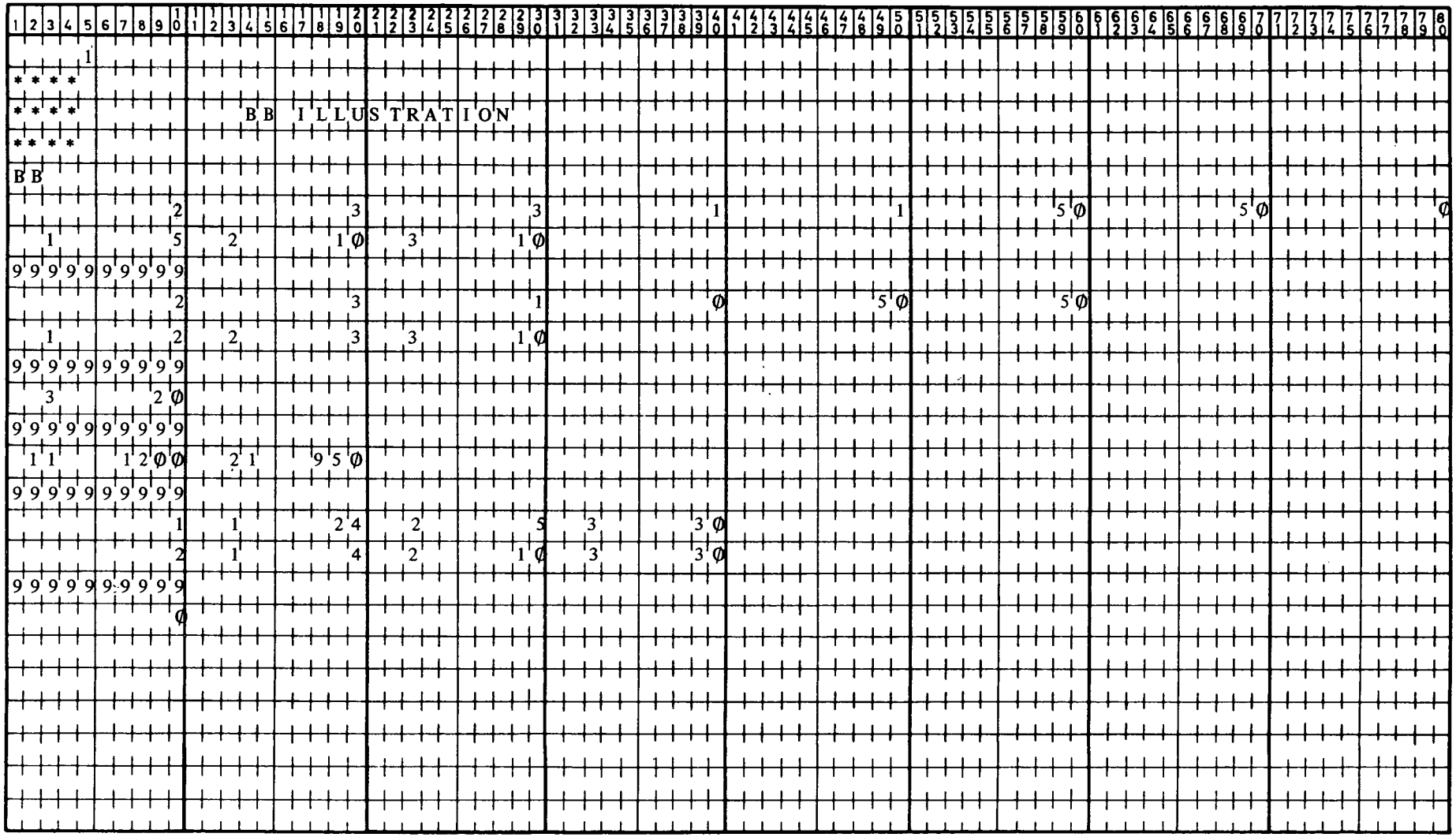


TABLE 4

Discrete programming (Branch and Bound): example data input

```
                                        ****
                                        ****              BRANCH AND BOUND ILLUSTRATION
                                        ****

                                                                  BRANCH AND BOUND

M (NO. OF CONSTRAINTS) =     2, N (NO. OF VARIABLES) =     3,
   3 OF THE VARIABLES ARE TO TAKE DISCRETE VALUES.

IRMAX =    50

ITRMAX =    25

M (NO. OF CONSTRAINTS) =     2, N (NO. OF VARIABLES--REAL, NOT SLACK) =     3,
NUMBER OF UPPER BOUNDED VARIABLES =     1.

INPUT CARDS FOR THE C ELEMENTS . . . .
(  1     2.000)(  2     3.000)(  3    10.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(999999999.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)

INPUT CARDS FOR THE UPPER BOUNDS ON SINGLE VARIABLES . . . .
(  3    20.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(999999999.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)

INPUT CARDS FOR THE B VECTOR, THE RIGHT-HAND SIDES OF THE CONSTRAINTS . . . .
(  11  1200.00)(  21   950.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)
(9999999999.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)(  00     0.00)

INPUT CARDS FOR THE ROWS OF THE A MATRIX . . . .
(  0     1.000)(  1    24.000)(  2     5.000)(  3    30.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(  0     2.000)(  1     4.000)(  2    10.000)(  3    30.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
(999999999.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)(  0     0.000)
```

TABLE 9

```
THE LP OPTIMUM, WITH FUNCTION VALUE          320.4543500, REACHED AT ITERATION     4, IS AS FOLLOWS...
    19.3181760      27.2727200      20.0000000

OBJECTIVE            320.45435000

                J. . .  1           2           3

           BOUND VECTOR. . . .
                 -1.0000     -1.0000     20.0000

             X VECTOR. . . .
                 19.3182     27.2727     20.0000
 I    Y VECTOR                                          B         B-AX
                ------------------------------------                (SLACK)
                (                                    )
 1      0.0364 (    24.0000      5.0000     30.0000 ) LE  1200.00      0.0000
                (                                    )
 2      0.2818 (     4.0000     10.0000     30.0000 ) LE   950.00      0.0000
                (                                    )
                ------------------------------------

             C VECTOR. . . .
                  2.0000      3.0000     10.0000

               Y'A-C. . . .
                  0.0000     -0.0000     +0.4545

ISEFF
  2  1

INBASE
  2  1 -1

    4 SIMPLEX ITERATIONS.

SOLUTION SATISFYING DISCRETE CONSTRAINTS, NO.     1, WITH FUNCTION VALUE,       309.99976 AT ITERATION     7                    *****
    10.0000000      30.0000000      20.0000000
THE VALUES OF THE SLACK VARIABLES ARE...
   209.9992100       9.9994946
THE FIXED VARIABLES ARE...
     2       3       1
FIXED AT VALUES...
   30.     20.     10.

INFEASIBLE TAIL AT ITR.     10.
THE FIXED VARIABLES ARE...
     2       3
FIXED AT VALUES...
   40.     20.

A TAIL WITH FUNCTION AT LEAST AS LOW AS THE BEST DISCRETE SOLUTION SO FAR, AT ITR.    12
 WITH FUNCTION   304.16650 AND FIXED VARIABLES...
     2       3
FIXED AT VALUES...
   50.     10.

A TAIL WITH FUNCTION AT LEAST AS LOW AS THE BEST DISCRETE SOLUTION SO FAR, AT ITR.    14
 WITH FUNCTION   305.00000 AND FIXED VARIABLES...
     2       3
FIXED AT VALUES...
   60.     10.

INFEASIBLE TAIL AT ITR.     16.
THE FIXED VARIABLES ARE...
     2       3
FIXED AT VALUES...
   70.     10.

A TAIL WITH FUNCTION AT LEAST AS LOW AS THE BEST DISCRETE SOLUTION SO FAR, AT ITR.    18
 WITH FUNCTION   306.66650 AND FIXED VARIABLES...
     2       3
FIXED AT VALUES...
   80.      0.

A TAIL WITH FUNCTION AT LEAST AS LOW AS THE BEST DISCRETE SOLUTION SO FAR, AT ITR.    19
 WITH FUNCTION   295.00000 AND FIXED VARIABLES...
     2
FIXED AT VALUES...
   90.

ALL BRANCHES OF THE TREE HAVE BEEN EXPLORED, AT ITERATION     19, AND THE OPTIMUM DISCRETE SOLUTION IS NO.
```

TABLE 9 (continued)

The first part of the PLP algorithm performs an ordinary LP optimisation (as in LINP) while the second part carries out the parametric variations on the optimal solution so obtained if indeed there is one to be found. There are three types of parametric variations available with PLP:

(i) Variation of an element or elements of the RHS, ($\underline{b}$), within specified limits, or variation of an element or elements of the objective function, ($\underline{c}$), between specified limits. Solutions (if any) are found for the problem.

Maximise $\underline{c}'\underline{x}$ such that $A\underline{x} \leqslant \underline{b}$ where

$$\underline{x} \geqslant \underline{o}$$

the jth component of $\underline{c}$ varies such that $r_j \leqslant c_j \leqslant l_j$

the ith component of $\underline{b}$ varies such that $s_i \leqslant b_i \leqslant t_i$

(ii) Perform sensitivity analyses on a RHS or an objective element. Thus a component of $\underline{b}$ or $\underline{c}$ is varied to find the range of values of the specified element of $\underline{b}$ or $\underline{c}$ within which the initial LP solution remains feasible and optimal.

(iii) Perform a range analysis on all RHS or objective elements, with less printing.

The type of parametrisation is controlled by TYPE. TYPE takes the value "B" for RHS parametrisation and "C" for parametrisation of the objective function. For complete data input formats see Block F of Appendix 1. Illustrations of data input and solution output appear in Tables 5 and 10; the output is discussed in Section 5.

We shall now extend the problem posed in 2(i) for LINP with the following additional data to illustrate the case of PLP:-

Vector	Element	Lower bound of variation range	Upper bound of variation range
RHS	b_1	50	200
RHS	b_2	95	150
Objective function	c_1	−1	1
Objective function	c_2	1	5
Objective function	c_3	10	20

5. DISCUSSION OF SOLUTION OUTPUT FORMAT OF THE SOLUTIONS OBTAINED

This Section gives some aid in interpreting the outputs to the examples posed in Section 4.

All printed output has the following general layout:

Page 1: Program Title

Page(s) 2: This page contains the run number of the particular problem. There will be NRUN such numbers. Following this any comments inserted at the beginning of the problem data are printed. The method employed by this particular problem is then stated.

Page(s) 3: This page contains details of data input and the problem solution. As the volume and type of this part of the output depends on the print control variable MOREPR, we will restrict our discussion to the examples presented in this guide. This discussion follows in Sections 5.1 to 55.

If MOREPR $\neq \emptyset$, page 3 is repeated, and pages 2-3 are repeated NRUN times.

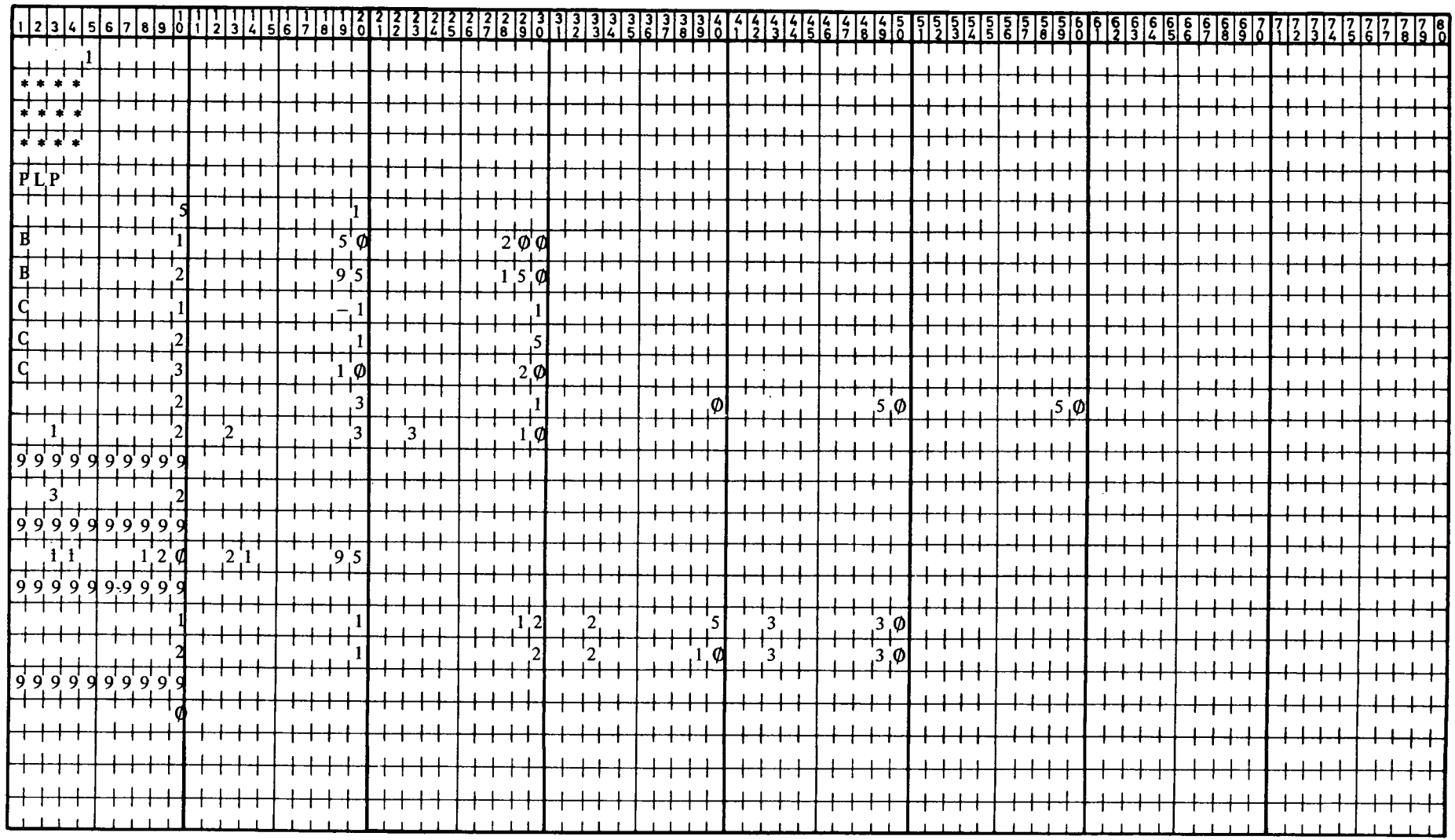


TABLE 5

Parametric linear programming: example data input

```
                                        ****
                                        ****                PLP  ILLUSTRATION
                                        ****
                                                                        PARAMETRIC L.P.

NUMBER OF PARAMETRIC REQUESTS      5 PRINT VARIABLE     0
B         1          50.000          200.000
B         2          95.000          150.000
C         1          -1.000            1.000
C         2           1.000            5.000
C         3          10.000           20.000

THE VALUE  1000000.000 WILL REPRESENT PLUS INFINITY AND THE VALUE -1000000.000 WILL REPRESENT MINUS INFINITY.

MOREPR =    25

IRMAX =   100

ITRMAX =    50

M (NO. OF CONSTRAINTS) =      2, N (NO. OF VARIABLES--REAL, NOT SLACK) =      3,
NUMBER OF UPPER BOUNDED VARIABLES =      1.

OBJECTIVE             32.04544100

                J. . .  1           2           3

           BOUND VECTOR. . . .
                 -1.0000     -1.0000      2.0000

             X VECTOR. . . .
                  3.8636      2.7273      2.0000
 I    Y VECTOR                                              B          B-AX
                ------------------------------------                   (SLACK)
                (                                  )
 1      0.0364 (    12.0000      5.0000     30.0000 ) LE    120.00      0.0000
                (                                  )
 2      0.2818 (     2.0000     10.0000     30.0000 ) LE     95.00      0.0000
                (                                  )
                ------------------------------------

             C VECTOR. . . .
                  1.0000      3.0000     10.0000

              Y*A-C. . . .
                 -0.0000     -0.0000     -0.4546

ISEFF
  2  1

INBASE
  2  1 -1

    4 SIMPLEX ITERATIONS.

OPTIMUM
```

TABLE 10

```
PARAMETRIC VARIATION ON THE OBJECTIVE FUNCTION ELEMENT OF VARIABLE     1


INITIAL BASIS OPTIMAL BETWEEN         0,600 AND         1,333

OBJECTIVE           32,045 VALUE OF OBJECTIVE FUNCTION ELEMENT       1,000
VALUE OF DUAL VARIABLES
      0,0364      0,2818
VALUE OF Y'A-C ELEMENTS
      0,0000     -0,0000     -0,4546

OBJECTIVE           32,045 VALUE OF OBJECTIVE FUNCTION ELEMENT       1,000
VALUE OF DUAL VARIABLES
      0,0364      0,2818
VALUE OF Y'A-C ELEMENTS
      0,0000     -0,0000     -0,4546


INITIAL BASIS OPTIMAL BETWEEN         0,600 AND         1,333


BASIS OPTIMAL BETWEEN -1000000,000 AND         0,600
VARIABLE     1 IS NON BASIC

OBJECTIVE           30,500 VALUE OF OBJECTIVE FUNCTION ELEMENT       0,600
VALUE OF DUAL VARIABLES
      0,0000      0,3000
VALUE OF Y'A-C ELEMENTS
      0,0000      0,0000     -1,0000

END OF PARAMETRIC VARIATION OF VARIABLE     1


PARAMETRIC VARIATION ON THE OBJECTIVE FUNCTION ELEMENT OF VARIABLE     2


INITIAL BASIS OPTIMAL BETWEEN         0,417 AND         3,167

OBJECTIVE           32,045 VALUE OF OBJECTIVE FUNCTION ELEMENT       3,000
VALUE OF DUAL VARIABLES
      0,0364      0,2818
VALUE OF Y'A-C ELEMENTS
      0,0000     -0,0000     -0,4546


BASIS OPTIMAL BETWEEN         3,167 AND         5,000

OBJECTIVE           32,500 VALUE OF OBJECTIVE FUNCTION ELEMENT       3,167
VALUE OF DUAL VARIABLES
      0,0333      0,3000
VALUE OF Y'A-C ELEMENTS
      0,0000      0,0000      0,0000

OBJECTIVE           47,500 VALUE OF OBJECTIVE FUNCTION ELEMENT       5,000
VALUE OF DUAL VARIABLES
      0,0000      0,5000
VALUE OF Y'A-C ELEMENTS
      0,0000      0,0000      5,0000


INITIAL BASIS OPTIMAL BETWEEN         0,417 AND         3,167

OBJECTIVE           26,591 VALUE OF OBJECTIVE FUNCTION ELEMENT       1,000
VALUE OF DUAL VARIABLES
      0,0727      0,0636
VALUE OF Y'A-C ELEMENTS
      0,0000     -0,0000     -5,9091

END OF PARAMETRIC VARIATION OF VARIABLE     2


PARAMETRIC VARIATION ON THE OBJECTIVE FUNCTION ELEMENT OF VARIABLE     3


INITIAL BASIS OPTIMAL BETWEEN         9,545 AND  1000000,000

OBJECTIVE           32,045 VALUE OF OBJECTIVE FUNCTION ELEMENT      10,000
VALUE OF DUAL VARIABLES
      0,0364      0,2818
VALUE OF Y'A-C ELEMENTS
     -0,0000     -0,0000     -0,4546

END OF PARAMETRIC VARIATION OF VARIABLE     3
```

TABLE 10 (continued)

```
PARAMETRIC VARIATION ON THE RIGHT HAND SIDE OF CONSTRAINT     1


INITIAL BASIS FEASIBLE BETWEEN        77,500 AND       270,000

OBJECTIVE           32,045 VALUE OF RIGHT HAND SIDE       120,000
VALUE OF REAL VARIABLES
      3,8636       2,7273       2,0000
VALUE OF SLACK VARIABLES
      0,0000       0,0000

OBJECTIVE           34,955 VALUE OF RIGHT HAND SIDE       200,000
VALUE OF REAL VARIABLES
     11,1364       1,2727       2,0000
VALUE OF SLACK VARIABLES
      0,0000       0,0000


INITIAL BASIS FEASIBLE BETWEEN        77,500 AND       270,000


BASIS FEASIBLE BETWEEN        47,500 AND        77,500

OBJECTIVE           30,500 VALUE OF RIGHT HAND SIDE        77,500
VALUE OF REAL VARIABLES
      0,0000       3,5000       2,0000
VALUE OF SLACK VARIABLES
      0,0000       0,0000

OBJECTIVE           28,667 VALUE OF RIGHT HAND SIDE        50,000
VALUE OF REAL VARIABLES
      0,0000       9,0000       0,1667
VALUE OF SLACK VARIABLES
      0,0000       0,0000

END OF PARAMETRIC VARIATION OF CONSTRAINT     1


PARAMETRIC VARIATION ON THE RIGHT HAND SIDE OF CONSTRAINT     2


INITIAL BASIS FEASIBLE BETWEEN        70,000 AND       180,000

OBJECTIVE           32,045 VALUE OF RIGHT HAND SIDE        95,000
VALUE OF REAL VARIABLES
      3,8636       2,7273       2,0000
VALUE OF SLACK VARIABLES
      0,0000       0,0000

OBJECTIVE           47,545 VALUE OF RIGHT HAND SIDE       150,000
VALUE OF REAL VARIABLES
      1,3636       8,7273       2,0000
VALUE OF SLACK VARIABLES
      0,0000       0,0000


INITIAL BASIS FEASIBLE BETWEEN        70,000 AND       180,000

OBJECTIVE           32,045 VALUE OF RIGHT HAND SIDE        95,000
VALUE OF REAL VARIABLES
      3,8636       2,7273       2,0000
VALUE OF SLACK VARIABLES
      0,0000       0,0000

END OF PARAMETRIC VARIATION OF CONSTRAINT     2
```

TABLE 10 (continued)

5.1 Linear Programming – Page 3 of the output produced by LINP

Problem formulation and data input are described in Section 4.1
The output is shown in Table 6.
Data Input formats are specified in Appendix 1, Block B.

As MOREPR = 1, there is a fair amount of solution and data printing (see Appendix 1, Block B). The value of MOREPR is printed together with the maximum allowable number of reinversions (IRMAX) and iterations (ITRMAX). The data input is printed in toto – this is self explanatory if studied in conjunction with Table 1. The actual solution to the problem then follows.

The figure against 'OBJECTIVE' is the current (i.e., optimal in this case) value of the objective function $\underline{c}'\underline{x}$. As the number of variables, N, is less than 8 in the example the problem is set out in pictorial form (an example where $N \geqslant 8$ is used in para. 5.3). A bound vector value of –1 indicates that this variable element has no individual upper bound (see Appendix 4). The box contains the A matrix, and the objective function coefficients are pointed directly underneath, while the RHS values appear to the right with the type of constraint – LE ($\leqslant$) in this case.

The primal values at the optimum (i.e., the current values) are printed in the row following 'X VECTOR', while the dual values are printed column-wise under 'Y VECTOR'. The 'B-AX' column contains the values of the - slack variables of each row.

As MOREPR = 1 in the example, the page following the main solution contains a print of the reduced inverse matrix along with other variables which may be of help in the event of a feasible or optimal solution not being found. A description of the role played by each of these variables may be found in Appendix 4.

5.2 Integer Programming – Page 3 of the output produced by MIF

Problem formulation and data input are described in Section 4.2. Data Input formats are specified in Appendix 1, Block C. Solution output is shown in Table 7.

The output is similar to that for LINP (5.1) except that all solution values are generally integer. As MOREPR = Ø in the example, the inverse print, etc., has been suppressed.

The user should be warned that due to the tolerance levels used here for MPCODE and the use of single precision variables in the program, some solution values of integer variables may not be exactly integer. For example. 2.ØØØ1 may appear instead of 2; this should cause no concern.

5.3 Quadratic Programming – Page(s) 3 of the output produced by QP

Problem formulation and data input are described in Section 4.3
Data input formats are specified in Appendix 1, Block D.
Solution output is shown in Table 8.

As for LINP (Section 5.1), data input is printed out before the program starts to optimise. The subsequent output from the QP example differs from that of the other examples as N, the number of variables, is greater than 7. On all occasions, and in all options, when $N \geqslant 8$ the solution cannot be output in picture form due to the limitation of the page width, and an abbreviated form of the solution is printed instead.

The non-zero elements of the A-matrix are printed in row-vector form. Each element is numbered (above each element) and these numbers are used to indicate the starting point of each successive row. This information follows the A-matrix print. Thus, in our example, the first row starts at element number 1, the second at element 4, and so on. Beneath each element is printed the variable associated with that element.

The objective function value follows.

The C, X and (Y'A – C) vectors are then printed row-by-row with each element under its respective variable number, J. For QP the C vector contains the partial derivatives of the quadratic function. The X vector holds the current values of the variables, as indeed it does for all MPCODE printouts.

Finally in the example (as MOREPR = Ø, the inverse print, etc., is suppressed) the B vector and sign, the Y vector and the (B – AX) vector are printed row-wise, each element under its respective row number, I. There may be more constraints present than were originally specified, but they will then have zero dual values. The sign vector, S, indicates the sign of the Ith constraint: Ø for "equal", 1 for "less-than-or-equal", –1 for "greater-than-or equal".

5.4 Discrete Programming – Page(s) 3 of the output produced by BB

Problem formulation and data input are described in Section 4.4.
Date input formats are described in Appendix 1, Block E.
Solution output is shown in Table 9.

The Branch-and-Bound algorithm uses as a starting point the continuous LP solution. The first part of the solution output on page(s) 3 is therefore the solution as obtained from LINP (Section 5.1). IPRBB = 1 in the example, so the solution is printed in full.

The program now proceeds with the tree search, and the rest of the output records details of each step in the search at the level of detail specified by the chosen value of IPRBB.

As IPRBB = 1 in the example, the fixed variables and their values are printed at each integer solution found. The first integer solution is, therefore, 310 (printed as 309.999 for the reasons stated in 5.2). This values is found with the variables x_1, x_2, x_3 fixed at 10, 30, 20 respectively. The variables x_1 and x_3 are then fixed at 40 and 20, and x_1 is varied. This tail of the tree is not feasible and is, therefore, not a candidate for a further revision of the x_1 value. All branches of the tree have now been explored, and a message is printed to this effect, together with the current best discrete solution, which in this case is the first. The solution is, therefore, $x_1 = 20$, $x_2 = 10$, $x_3 = 30$ with an objective function value of 310.

More diagnostic printing is available by setting IPRBB $\geqslant$ 2 (see Appendix 1): these settings are of particular assistance when MPCODE is being used to develop new algorithms and for full explanations of the specialised print-outs thus obtained, and a description of the BB algorithm, see Reference 3.

As the maximum number of iterations has not been exceeded, the restart facility was not invoked by the program.

5.5 Parametric Linear Programming – Page(s) 3 of the output produced by PLP

Problem formulation and data input are described in Section 4.5.
Data input formats are specified in Appendix 1, Block F.
Solution output is shown in Table 10.

PLP carries out a parametric analysis on the LP optimum. The first part of the solution output on page(s) 3 is, therefore, the data input and solution similar to that obtained from LINP, except that in the example MOREPR = Ø so the inverse print, etc., is suppressed.

The rest of the output contains details of the parametric analysis required.

In the example quoted, the first variation required is that of the first constraint with the RHS ranged between 50 and 200. The initial basis is found to be feasible between limits of 77.5 and 270 for the RHS. At the initial basis stage the values of the objective, the primal variables, the slack variables and (as IPRPLP = Ø) the values of the dual and function variables are printed. The value of the parameter, the RHS of constraint 1, is increased to 200 and the

values of the solution variables are printed. The value of the parameter is decreased to 77.5 and the lower limit, 47.5, of the parameter for that basis is calculated. The solution values when the RHS of constraint 1 is 77.5 are printed. Finally, the solution values at the lower limit, 50.0, of the specified range are printed.

The parametric variation of the right-hand-side of constraint 2 works the same way.

The parametric analysis of the objective function coeffecient of x_1 (c_1) is approached next. The initial basis is found to be optional between 0.6 and 1.333, and the objective value of 32.045 is found when $c_1 = 1$. The dual vector, (Y A – C) vector, real and slack variables are printed along with this information. As with the RHS variations the rest of the c_1 range ($-1 \leqslant c_1 \leqslant 1$) is investigated in 3 further similar steps and it is discovered that the 'central' value of c_1 is 1. (The initial LP solution in this case).

Parametric variations of the other two objective function elements are then done in the same manner, and as all variations have then been completed, the process ends.

6. AVAILABILITY OF MPCODE

The version of MPCODE described in this Report includes a driver routine and tolerance levels set to allow single precision declarations of all variables: this code has been run on IBM 360 and ICL Series 4 machines. The basic version of this code is completely specified in Land and Powell's book[3], and has been run on ICL 1900, CDC 6600 and Univac 1108 and also on an IBM 360 when double precision had been declared for the variables. This manual refers equally to both versions, although the driver routine is clearly only applicable to the IBM/ICL single precision release.

A standard 9 track, 800 bpi, IBM compatible tape has been prepared to support the system, and its contents are as follows in fixed length block format for records of 800 8-bit characters per block (10 card images per block) in odd parity with an interblock gap of 1.5 cm. The code is EBCDIC.

ICL/IBM standard volume label

ICL/IBM standard header label

Tape Mark

Descriptive cards specifying the tape format, contents and references to documentation.

Tape Mark

Card images of the code precisely as published in Land and Powell's book, Reference 3, (as for CDC 6600 and ICL 1900).

Tape Mark

Card images of the MPCODE code precisely as supported by this TRRL Report (as for ICL 4/70 and IBM 360/370).

Tape Mark

Data cards required to generate LINP example as in Reference 3.

Data cards required to generate MIF example as in Reference 3.

etc.

Tape Mark

Tape Mark

Arrangements can be made to copy this master tape onto tape supplied by prospective users.

7. ACKNOWLEDGEMENTS

The extensive testing required to establish the robustness of MPCODE on several machines could not have been completed without the facilities of the SRC IBM/370 and ICL/1900 installations at Chilton and the Univac 1108 at the National Engineering Laboratory. This Report was prepared with Urban Transport Division, (Head: A R Cawthorne) at TRRL.

8. REFERENCES

1. INTERNATIONAL COMPUTERS LTD. ICL Transportation Reference Manual. Technical publication 1074.

2. INTERNATIONAL COMPUTERS LTD. ICL Linear Programming 400 for System 4. Technical publication 4523.

3. LAND, A, and S POWELL. Fortran Codes for Mathematical Programming: Linear, Quadratic and Discrete. Wiley (London), 1973.

4. LAND, A, and G MORTON. Inverse-basis method for Beale's Quadratic Programming Algorithm. Man. Sci. 18 (1973).

5. VAJDA, S. Mathematical Programming. Addison Wesley (London), 1961.

9. APPENDIX 1

DATA INPUT FORMATS

These are the input cards required for each complete run. All data should be right-adjusted within their respective fields, and values should be integer except for the C, BOUND, B and A vectors which may take real values, and the variable "TYPE" in PLP which requires data in character form.

␣ indicates a *significant* blank space.

Block A

Data Input required to initialise MPCODE for each complete job.

	Field	Format	Internal Name	Specification
Card 1	Cols 1 - 5	(15)	NRUN	Number of runs with different data inputs and methods
Card(s) 2	Cols 1 - 4	(A4)	****	Comment card. Anything entered on this card is printed on execution. Any number of these may be used.
Card 3	Cols 1 - 4	(A4)		Keyword. The only valid entries are:
			L I N P	Linear Programming
			Q P ␣ ␣	Quadratic Programming
			M I F ␣	Integer Programming (Method of Integer Forms)
			B B ␣ ␣	Mixed Integer Programming (Branch and Bound)
			P L P ␣	Parametric Linear Programming

Block A together with any one of the Blocks B – F (see below) is repeated NRUN times. Blocks B – F: Data cards for optimisation by the method chosen in Block A.

Block B

Input cards for LINP (to be preceded by Block A).

	Field	Format	Internal Name	Specification
Card 1	Cols 1 - 10	(I10)	M	Number of constraints
	Cols 11 - 20	(I10)	N	Number of variables
	Cols 21 - 30	(I10)	ISBND	Number of bounded variables
			ISBND = –1 :	All variables have an upper bound of 1.

	Field	Format	Internal Name	Specification
Card 1 (Contd)	Cols 31 - 40	(I10)	MOREPR	= ∅: Print the input data, do not print the inverse matrix in IPRINT. = 1: Print the input data *and* inverse matrix. = 2: Print the inverse matrix but not the input data = 3: Do not print the inverse matrix or input data
	Cols 41 - 50	(I10)	ITRMAX	Maximum number of iterations = ∅ : will be set to 3* (M + N + (No. of bounded variables))
	Cols 51 - 60	(I10)	IRMAX	Maximum number of reinversions. 20 is a reasonable number to start with.
Card 2	Cols 1 - 3	(I3)	J	Number of a non-zero element of the objective function.
	Cols 5 - 10	(F6.0)	C(J)	Element of the objective function
	:			
	Cols 11 - 13	(I3)	J	
	:			
	Cols 71 - 73	(I3)	J	
	Cols 75 - 80	(F6.0)	C(J)	

Card 2 is repeated until all non-zero elements of the objective function have been specified, up to 8 per card.

Card 3	Cols 1 - 10	(I10)	9999999999	Terminates input of the objective function elements

If ISBND > 0, the next card is Card 4, otherwise it is Card 6.

Card 4	Cols 1 - 3	(I3)	J	Number of a variable with an associated upper-bound
	Cols 5 - 10	(F6.0)	BOUND(J)	Value of the upper bound on the Jth variable.
	Cols 11 - 13	(I3)	J	

	Field	Format	Internal Name	Specification
Card 4 (Contd)	Cols 15 - 20	(F6.0)	BOUND(J)	
	⋮		⋮	
	Cols 71 - 73	(I3)	J	
	Cols 75 - 80	(F6.0)	BOUND(J)	

Card 4 is repeated until all the upper bounds have been specified (up to 8 on each card). (Any not specified are assumed to have no upper bounds.)

Card 5	Cols 1 - 10		9999999999	Terminates input of the upper bounds.
Card 6	Cols 1 - 3	(I3)	I	Number of an element of the B (RHS) vector.
	Col 4	(I1)	S(I)	Type of constraint = $\emptyset$: an equality constraint = 1 : a $\leqslant$ constraint = 2 : a $\geqslant$ constraint
	Cols 5 - 10	(F6.0)	B(I)	The Ith element of the B vector.
	Cols 11 - 13	(I3)	I	
	Col 14	(I1)	S(I)	
	Cols 15 - 20	(F6.0)	B(I)	
	⋮		⋮	
	Cols 71 - 73	(I3)	I	
	Col 74	(I1)	S(I)	
	Cols 75 - 80	(F6.0)	B(I)	

Card 6 is repeated as usual until all the elements of the B vector have been specified. Any not specified are assumed to be less than a very large number (10^8 for System - 4).

Card 7	Cols 1 - 10		9999999999	Terminates input of the B elements.
Card 8	Cols 5 - 10	(I5)	I	Number of a row of the A-matrix
	Cols 11 - 13	(I3)	J	Number of a column of the A-matrix.
	Cols 15-20	(F6.0)	A(I,J)	A non zero element of the A-matrix.

	Field	Format	Internal Name	Specification
Card 8 (Contd)	Cols 21 - 23	(I3)	J	
	Cols 25 - 30	(F6.0)	A(I,J)	
	⋮			
	Cols 71 - 75	(I3)	J	
	Cols 75 - 80	(F6.0)	A(I,J)	

All the values on a card must belong to the same row, but they need not be in correct column-order. The rows must, however, be in correct row order. 7 values may be specified on each card, and Card 8 is repeated till all non-zero entries in the A-matrix have been specified; any values not specified here are assumed to be zero.

Card 9	Cols 1 - 10		9999999999	Terminates input of the A-matrix elements.
Card 10	Col 10	(I10)	MORE	= Ø: more problems are to be specified for optimisation by LINP*; Block A need, therefore, not be repeated unless there is another problem by a different method. = Ø: No more problems for this method (unless block A is repeated).

Block C

Data Input for MIF (to be preceded by Block A).

As in Block B except that if ITRMAX is read in as Ø, it will be set to 3*N*(M + N + (No. of upper bound variables)), and "LINP" (*) in the description of Card 10 should read instead "M I F —". Also the B(I) coeffecients on Card 6 and the A(I,J) coeffecients on Card 8 must have integer values.

Block D

Data Input for QP (to be preceded by Block A).

Card 1	Cols 1 - 10	(I10)	M	Number of constraints
	Cols 11 - 20	(I10)	N	Number of variables
	Cols 21 - 30	(I10)	NUMQ	Number of quadratic variables
Card 2	Cols 1 - 5	(I5)	ID	The variable associated with a row of the D matrix.
	Cols 6 - 10	(I5)	JD	The variable associated with the column of the D matrix.

	Field	Format	Internal Name	Specification
Card 2 (contd)	Cols 11 - 20	(F10.0)	D(ID,JD)	A non-zero value of the (ID,JD) element of D. By symmetry D(ID,JD) and D(JD,ID) have the same value and hence it is unnecessary to specify both coefficients.
	Cols 21 - 25	(I5)	ID	
	Cols 26 - 30	(I5)	JD	
	Cols 31 - 40	(F10.0)	D(ID,JD)	
	⋮			
	Cols 71 - 80	(F10.0)	D(ID,JD)	

Card 2 is repeated until all the non-zero elements of D have been specified.

Card 3	Cols 1 - 10		9999999999	Terminates input of the elements of D.

Cards 4 to 13 are as for Cards 1 to 10 of Block B, with suitable adjustments in the interpretation of Card 10 in Block B. The vector p of the quadratic function takes the place of c in the LINP input.

Block E

Input to BB (to be preceded by Block A).

Card 1	Cols 1 - 10	(I10)	M	Number of constraints.
	Cols 11 - 20	(I10)	N	*Total* number of variables
	Cols 21 - 30	(I10)	NUMD	Number of discrete variables $= -1$: all the variables have a stepsize of 1.
	Cols 31 - 40	(I10)	INTOBJ	= 1 : if the value of the objective function at the optimal integer solution will be an integer. $= \emptyset$: otherwise
	Cols 41 - 50	(I10)	IPRBB	$\geqslant \emptyset$: At an integer solution the values of the real and slack variables will be printed irrespective of the value of IPRBB $\geqslant$ 1 : IPRINT is called at the initial LP optimum. A message with the number of iterations is printed at all tails.

	Field	Format	Internal Name	Specification
Card 1 (Contd)		Cols 41 - 50 (Contd)		= 1 : The fixed variables and their values are also printed at an integer solution. $\geqslant$ 2 : The values and function estimates right and left of the tree are also printed at an integer solution. = 3 : The estimates of the rate of fall of the function to the left and right are printed at a tail.
	Cols 51 - 60	(I10)	ITRBBM	Maximum number of iterations (for BB). If $\emptyset$, it is set to N*ITRMAX
	Cols 61 - 70	(I10)	IRBBM	Maximum number of reinversions for BB.
	Cols 71 - 80	(I10)	IREST	= $\emptyset$: the current run is started from scratch. = 1 : the calculation is to be restarted.

If NUMD = 0 or -1 the next card is Card 4, otherwise Card 2.

Card 2	Cols 1 - 3	(I3)	J	Number of a discrete variable
	Cols 5 - 10	(I6)	JDISC(J)	Step size of the Jth discrete variable.
	Cols 11 - 13	(I3)	J	
	Cols 15 - 20	(I6)	JDISC(J)	
	⋮			
	Cols 71 - 73	(I3)	J	
	Cols 75 - 80	(I6)	JDISC(J)	
Card 3	Cols 1 - 10		9999999999	Terminates input of the step sizes.

Cards 4 - 13 are as Block B cards 1 - 10.

	Field	Format	Internal Name	Specification
Block F				
Data Input to PLP (to be preceded by Block A)				
Card 1	Cols 1 - 10	(I10)	NUMP	Number of parametric variations.
	Cols 11 - 20	(I10)	IPRPLP	= $\emptyset$: the dual and function row variables are printed when parametrising c_j, the original and slack variables are printed when parametrising b_i.
				= 1 : the dual, function row basic and slack variables are printed when parametrising either b_i or c_j.
Card 2 has 3 forms:				
i) to parametrise an objective or RHS element between specified limits.				
	Col 1	(A1)	TYPE	Character B, if an element of the RHS is to be varied, or C if an element of the objective function is to be varied.
	Cols 2 - 10	(I9)	I or J	Number of row, I, whose RHS element is to be varied, or column, J, whose objective function element is to be varied.
	Cols 11 - 20	(F10.0)	TLOW	Lower bound of the variation.
	Cols 21 - 30	(F10.0)	THIGH	Upper bound of the variation.
ii) to perform a sensitivity analysis on a RHS or an objective function element.				
	Col 1	(A1)	TYPE	B or C as above.
	Cols 2 - 10	(19)	I or J	As above.
	Cols 11 - 20	(F10.0)	B(I) or C(J)	The value of the element in the original LP
	Cols 21 - 30	(F10.0)	B(I) or C(J)	

Field	Format	Internal Name	Specification

(iii) to perform a range analysis on all objective function and/or RHS elements.

Col 1	(A1)	TYPE	B or C as above.
Cols 11 - 20	(I10)	I or J	= −1 all the RHS and objective function elements are to be analysed.

Card 2 is repeated NUMP times. The three types of Card 2 may be in any order.

Cards 3 to 13 are as Cards 1 to 10 of Block B.

10. APPENDIX 2

CORE AND TIME REQUIREMENTS FOR ICL SYSTEM - 4

The amount of core space available naturally limits the size of problem which MPCODE can handle. All tests on the TRRL System 4/70 machine have, therefore, been based on a program which will handle problems with up to 25 constraints and 50 variables. The MPCODE overlay at TRRL has been designed to satisfy minimum core requirements and disc space.

MPCODE with a 25 x 50 matrix and the above-mentioned overlay needs 88k bytes of store buffer space. The program occupies 52 tracks of disc space.

The following table gives indications of run time (in etu's) which may be expected (1 etu = 3½ secs).

Option	Rows	Variables	Bounds	Discrete Variables	Quadratic Variables	Parametric Requests	Iterations	Time (etu's)
LINP	2	3	1	–	–	–	4	1
BB	2	3	1	3	–	–	19	1
MIF	2	3	1	–	–	–	53	3
MIF	2	3	1	–	–	–	27	2
QP	16	18	–	–	6	–	13	1
PLP	2	3	1	–	–	5	4	1

11. APPENDIX 3

JOB CONTROL

11.1 Job control for the ICL System – 4 Multijob Operating System

Input is from Unit 05, unless the BB restart facility is being used, so the initial data must be set up in a line file with identifier of the form USRNAM: GRPNAM. DSET05 (Snnn Ø) where USRNAM is the user's username. GRPNAM is a groupname and nnn is an integer between 0 and 999. Printer output is to DSET06. If the BB restart facility is utilised, then output will also be to disc, tape or card punch on Unit 07, and input subsequently from the same device on Unit 08. The job description presented here assumes that DSET07 and DSET08 are disc files (the most efficient configuration).

```
//␣ LOGIN    ␣ USRNAM, -----
//␣ GROUP    ␣ GRPNAM
//␣ SCHEDULE ␣ MPCODE, t, nnn/00
//␣ CONFG    ␣ STORE = z, RSP = rsp
//␣ FILE     ␣ DSET07, RA, FILNM1, TK, VOLlvn          (a)
//␣ FILE     ␣ DSET08, RA, FILNM2, (zqqqØ), VOLlvn     (b)
//␣ EXEC     ␣ DSET06
//␣ LOGOUT
```

Notes: I. t is the time in etu's allowed for the job, and z is the store allocated. r is the rank, s the stream and p the priority of the job.

II. k is the number of tracks allocated to this file and lvn is the volume number of the disc where these files will be found (using the BB restart facility).

III. FILNM1, FILNM2 are the names of these files where the BB restart facility is being used. qqq is the run number of the (BB) job which created the previous output file.

IV. If the BB option is not being used, or if the user is sure the BB restart facility will not be employed by the program, parameters (a) and (b) may be omitted and notes II and III ignored.

V. A standard job description could use 180 units of store and 10 tracks of disc space.

12. APPENDIX 4

LIST OF VARIABLES WHICH MAY OCCUR IN OUTPUT AND INPUT AND A LIST OF TOLERANCES

(For more detailed explanations, see Reference 3)

B(I)	The right-hand side, $\underline{b}$, vector.
BOUND (J)	The vector of upper bounds for each variable. If no bound is present, BOUND(J) = $-1.\emptyset$
C(J)	The objective, $\underline{c}$, vector
D(ID,JD)	The matrix D, of the function $\underline{p}'\underline{x} + \frac{1}{2}\underline{x}' D\underline{x}$. Only rows and columns which have at least one non-zero entry need be stored.
DRIVER	Used when seeking a new variable to reduce the infeasibility of the NEGINV row; it shows whether the value of the variable in that row is to be driven up (DRIVER = $1.\emptyset$) or down (DRIVER = $-1.\emptyset$)
INBASE(J)	A vector of elements indicating whether the Jth variable is basic or not. If it is basic, INBASE(J) contains the element K showing which row of the inverse matrix and which element of XR(K) is associated with the Jth variable. If it is non basic at its lower bound of zero, INBASE(J) = $\emptyset$, and if it is non basic at its upper bound, INBASE(J) = -1.
INREV	An indicator as to whether to set up the slack variables from $\underline{b} - A\underline{x}$ (INREV = 1) or to alter the slacks by subtracting a multiple of the slack changes made at each base change (INREV = $\emptyset$).
INTOBJ	See Appendix 1, Block E.
IPRBB IPRBBM	See Appendix 1, Block E.
IPRPLP	See Appendix 1, Block F.
IR	The current number of reinversions performed.
IREST	See Appendix 1, Block E.
IRMAX	The maximum allowable number of LP reinversions
ISBND	The number of upper-bound variables. If ISBND = -1, all variables have an upper bound of 1
ISEFF(I)	A vector of elements indicating whether the Ith constraint is effective and explicitly represented in the inverse (in which case it shows the associated column of the inverse) or not (in which case ISEFF(I) = $\emptyset$).
ISTATE	This variable contains information about the conditions on entering and leaving LINP and QP. For full details see Reference 3.
ITR	The current number of LP iterations performed.
ITRBBM	See Appendix 1, Block E.
ITRMAX	The maximum allowable number of LP iterations.
JDISC(J)	Step size of the Jth discrete variable in BB
M	Number of rows of the basic problem.
MARKI	Marks the constraint represented explicitly by a slack variable in the basis.
MARKK	Marks the row of the inverse matrix containing the slack variable indicated by MARKI.
MAXA	Maximum number of elements allowed in the A matrix.
MAXN	Maximum number of variables which may be submitted to the program.

MAXM	Maximum number of rows which may be submitted to the program.
MORE	See Reference 3 and Appendix 1.
MOREPR	See Appendix 1, Block B.
MXSIZE	The maximum size of the inverse matrix.
N	The number of columns of the problem.
NEGINV	The row of the inverse associated with an infeasible variable.
NEGROW	The row of the constraints matrix with the greatest infeasibility.
NEWY	The row of the inverse or of the A matrix which limits the size of the current basis change.
NEWX	The new variable to be introduced into the basis.
NUMP	The number of parametric requests in PLP.
NUMQ	The number of quadratic variables in QP.
NUMSLK	The number of slack variables explicitly represented in the basis.
R	The limit of the entering variable.
S(I)	The vector of inequality types of the constraints.
SIZE	Current size of the inverse.
TYPE TLOW THIGH	See Appendix 1, Block F.
X(J)	The vector, $\underline{x}$, of the variables of the primal problem.
XBASIS(K) (Printed as XBS(K))	The labels of the rows of the inverse matrix containing the current basic variables. An explicit basic slack variable on the Ith row is shown by N + I in XBASIS(K).
XKPOS	Shows whether the new variable entering the basis is entering positively (XKPOS = 1.Ø) or negatively (XKPOS = −1.Ø).
XR(K)	The values of the variables listed in XBASIS(K)
Y(I)	Vector of the dual variables.
YAC(J)	(= y'A-c') The updated objective row of the LP calculation.
YAMINC	(= $y'a_k$-c_k) The element in the updated row of the entering variable (NEWX).
YBASIS(L)	The column labels of the inverse matrix, containing the numbers of the (currently) active constraints.
YR(L)	The values of the dual variables of the constraints in YBASIS(L).

Tolerances — with the values which have been set for the ICL4/70 single precision version.

Variable	Value	
BIG	10^{8}	This variable is large, and treated as a representation of 'infinity'.
SMALL	10^{-7}	A small value, sufficiently small so that a variable with a value less than it may be treated as zero without introducing any error.
TOL (1)	10^{-3}	Tests the feasibility of the basic primal variables at their upper and lower bounds.
TOL (2)	10^{-3}	Tests the feasibility of the slacks on the ineffective constraints.
TOL (3)	10^{-3}	Tests the optimality of the dual variables of the effective constraints.

Variable	Value	
TOL (4)	10^{-3}	Tests the optimality of the objective row values of the non-basic variables
TOL (5)	10^{-6}	Tests whether a pivot should be regarded as zero.
TOL (6)	10^{-3}	Tests with relative error of the primal variables of a solution on the Ith constraint.
TOL (7)	10^{-3}	Tests the relative error of the dual variables of a solution at the Jth variable.
TOL (8)	10^{-5}	Tests the size of a proposed pivot during a reinversion of the inverse (connected with TOL (5)).

There are two tolerances in MIF:-

TOLIF1	10^{-2}	Tests whether a variable has an integer value or not.
TOLIF2	10^{-1}	Used as a criterion to decide whether the coefficients of a new constraint are accurate or not.

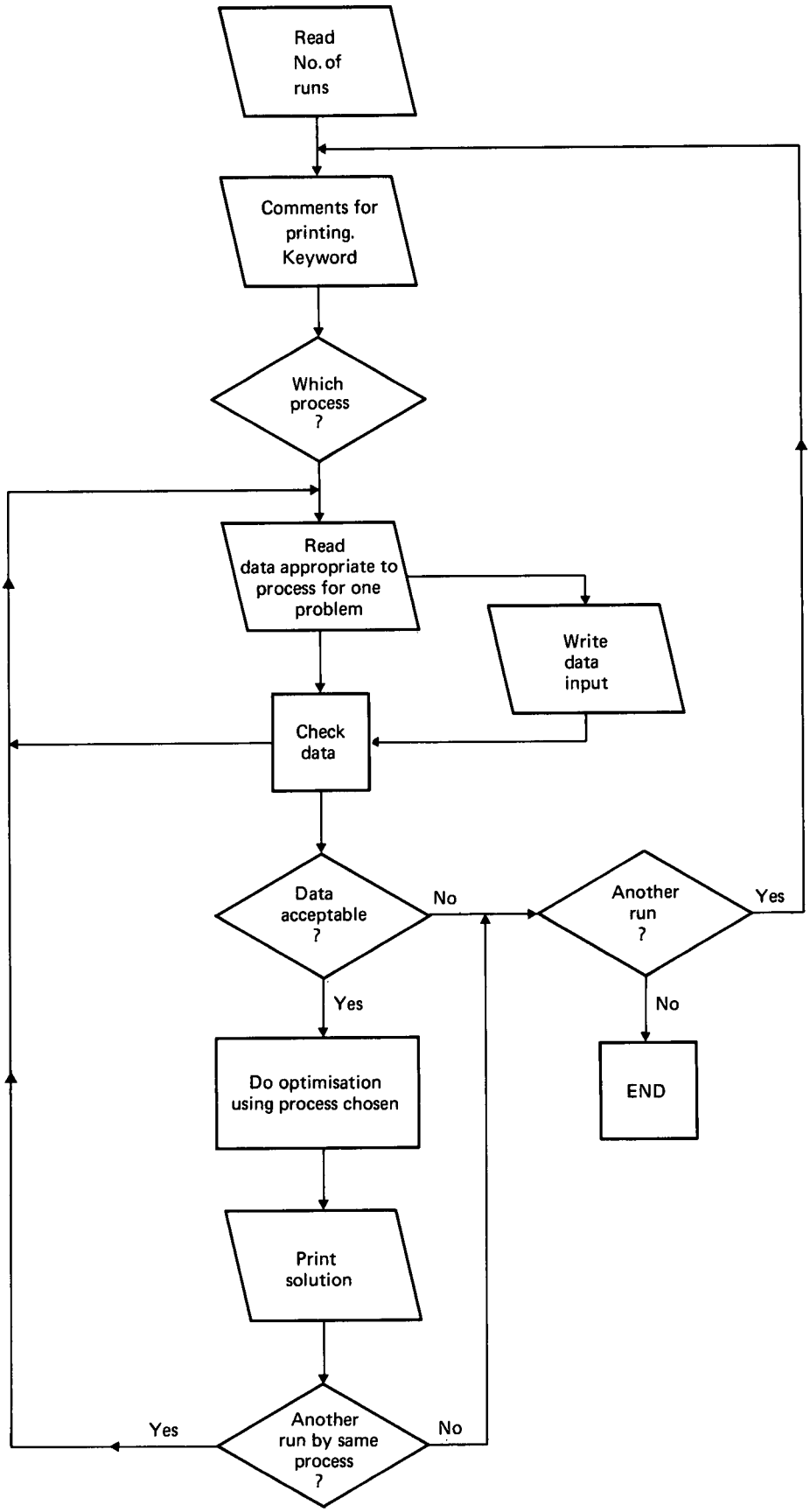


Fig.1. BASIC FLOWCHART FOR MPCODE

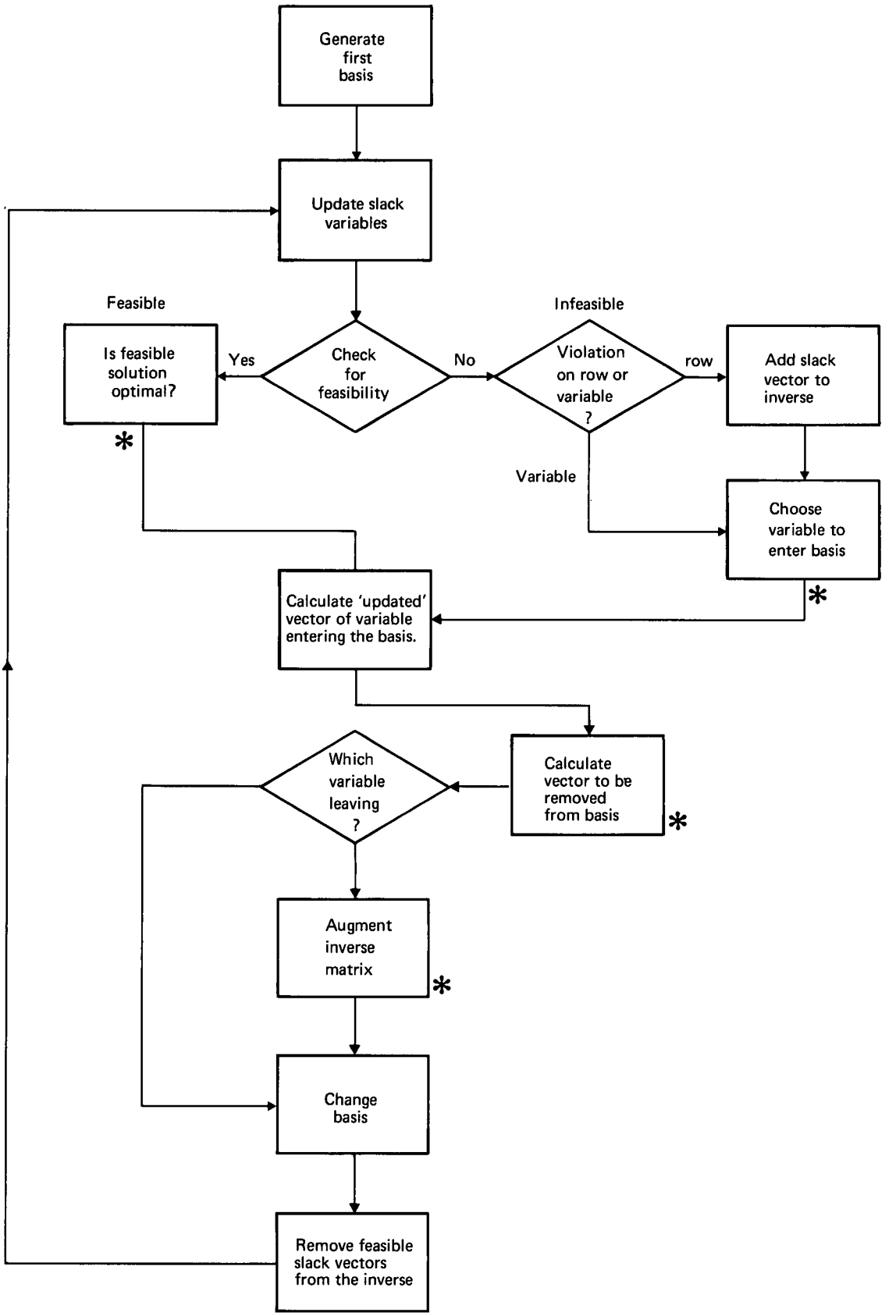


Fig.2. FLOWCHART FOR DOANLP DRIVING ROUTINE. PROCESS COULD FINISH AT ANY ✻

(2404) Dd635221 500 1/74 HP Ltd., So'ton G1915
PRINTED IN ENGLAND

ABSTRACT

MPCODE: A versatile linear and quadratic mathematical programming system: A H LAND S POWELL N J PAULLEY and M R WIGAN: Department of the Environment, TRRL Supplementary Report SR17UC, Crowthorne, 1974 (Transport & Road Research Laboratory). An extensive Mathematical Programming System has been written by A Land and S Powell under SRC funding on CDC 6600 and ICL 1900 computers. TRRL have collaborated with L.S.E. and S.R.C. Atlas Computer Laboratory to develop and test a version of this system in parallel for ICL System 4 and IBM 360 computers. A high degree of problem-solving robustness and reliability has been achieved. Tolerance levels were determined that allow a substantial reduction in core storage required. Examples of linear, parametric, interger, discrete and constrained quadratic formulations are given, and the document forms a Users Manual for this system for both System 4 and IBM 360/370 computers.

ABSTRACT

MPCODE: A versatile linear and quadratic mathematical programming system: A H LAND S POWELL N J PAULLEY and M R WIGAN: Department of the Environment, TRRL Supplementary Report SR17UC, Crowthorne, 1974 (Transport & Road Research Laboratory). An extensive Mathematical Programming System has been written by A Land and S Powell under SRC funding on CDC 6600 and ICL 1900 computers. TRRL have collaborated with L.S.E. and S.R.C. Atlas Computer Laboratory to develop and test a version of this system in parallel for ICL System 4 and IBM 360 computers. A high degree of problem-solving robustness and reliability has been achieved. Tolerance levels were determined that allow a substantial reduction in core storage required. Examples of linear, parametric, interger, discrete and constrained quadratic formulations are given, and the document forms a Users Manual for this system for both System 4 and IBM 360/370 computers.