Digital Computer Laboratory
Massachusetts Institute of Technology
Cambridge 39, Massachusetts

To: D. N. Arden

From: R. L. Bishop

Date: October 29, 1956

SUBJECT: SIMPLEX METHOD SUBROUTINE

There is available a program which will solve linear programming problems by the simplex method. This program has been used with good success on problem 216.

Description of Problem

This program is used to find the maximum of a linear function in several variables,

$$(1) \qquad Z = \sum_{j=1}^{m} c_j x_j$$

subject to the restrictions

$$(2) \qquad \sum_{j=1}^{m} a_{ij} x_j \leqq b_i, \; b_i \geqq 0, \; i = 1, ---, n,$$

$$\text{and } x_j \geqq 0, \; j = 1, ---, m.$$

This differs from the more general formulation of the problem in that no equalities are allowed under (2). It is also required that the initial variable values be zeros.

Use of the Program

The program starts with the matrix of coefficients, (a_{ij}), stored by columns on the auxiliary drum beginning with an arbitrary drum address, which is the preset parameter ph 1. The value of the function, initially zero, is stored in CM at flad Z1, which must be assigned. Following it, starting at 2Z1, are the numbers $-c_j$, i.e., the negatives of the coefficients in the linear form (1). The restriction constants b_i are also in CM, starting with flad m1, which must be assigned. In addition, storage for two columns of the matrix must be provided at flads p1 and m2, both of which must be assigned. All of these numbers are in (24,6) form.

Besides these, the program requires a block of WW integers to keep track of the order of the variables and restrictions. There will be n+m of these single register numbers, referred to as tags, and it is arbitrary what numbers are used, although it is very convenient to use relative addresses of the variables in subsequent programs. The tags for the restrictions start at flad d1, which must be assigned, and hence, the tags for the variables start at dl+n.

Finally, the parameters pnl = n = number of variables, pml = m = number of b_i's and phl = drum address of matrix must be assigned. Alternately, these may be inserted in three registers of the program, m at 20r, phl at 21r, and n at 22r.

The entry to the program is a ta instruction at or. The exit is an sp to the original control unless an infinite maximum is indicated, in which case a jump to 0 is made from 4t8.

Form of Results

The maximal value of z will be at zl.

The values of the variables, x_j, and the amounts by which the restrictions (2) are inequalities, i.e., the so-called slack variables, $p_i = b_i - \sum a_{ij} x_j$, are determined by the quantities left at ml and by the order of the tags. Recall that the tags correspond in one-to-one fashion with the variables x_j and slack variables p_i. To find the desired values the following correspondence is used:

(number at ml + 2i) ⟷ (variable or slack variable corresponding to tag at dl+i)

i = 0, ---, n-1

(zero) ⟷ (variable or slack variable corresponding to tag at dl+n+j)

j = 0, ---, m-1

If addresses (relative or otherwise) are used as tags, a sorting technique which picks up the n numbers at ml and places them at the addresses which are the corresponding tags and then fills in the remaining addresses with zeros is convenient.

Other results which are sometimes used are the numbers which Charnes denotes by $z_j - c_j$. These will be found starting at 2zl in an order corresponding to the last m tags.

The final matrix is the partially inverted original one, and the order of rows and columns also corresponds to the final order of the tags.

For convenience, the number of passes is counted in both FF6 and register tl3. In addition, the value of z at any moment of the maximizing process is stored as a (24,6) number in FF4 and FF5. Thus these results may be obtained as the process continues and can be used to determine a stopping point previous to the attaining of the actual maximum. In order that the program could be stopped at a non-crucial point of the cycle, an sil instruction has been included.

Operating Time

The operating time is determined by two factors, one of which can be measured precisely, i.e., the time per pass, and the other of which can only be estimated, i.e., the number of passes. Since the time consumed in a pass is mostly due to performing an algorithm on each element of an nxm matrix, the time should be proportional to mn. Upon actually plotting pass times vs. mn a suitable equation was found:

$$\text{time per pass} = .000075mn + .03 \text{ minutes.}$$

The classical estimate of the number of passes is 2m. In twelve different trials with matrices ranging from 31 x 10 to 77 x 36 the number of passes was found to vary from 1.2m to 2.4m, with an apparent tendency to give the fewer number of passes with smaller matrices. In addition, twenty-five trials were made in which the initial conditions were somewhat closer to the maximum, with the number of passes varying from 1.1m to 1.6m.

Notes

If it is desired to eliminate the restriction $x_j \geqq 0$ for one or more variables the classical procedure has been to make a variable substitution which doubles the number of variables. In problem 216 it was found convenient to group the variables according to order of magnitude and then make only one substitution for each group. For instance, if say, $x_1, \ldots x_5$ were to be non-restricted and (x_1, x_2), (x_3, x_4, x_5) were the grouping according to order of magnitude, then two new variables y_1 and y_2 would be introduced, and the variables used in the program would be

$$x_1 + y_1,\ x_2 + y_1,\ x_3 + y_2,\ x_4 + y_2,\ x_5 + y_2,\ y_1,\ y_2$$

These seven variables would be restricted, i.e., $x_1 + y_1 \geqq 0$, etc., but the restrictions $x_1 \geqq -y_1$, etc., will allow $x_1, \ldots, x_5$ to become as negative as necessary.

Degeneracy has been said to be the greatest fault of the simplex method, to the possibility of "cycling". However, this possibility is quite rare, so no precaution against it was made in the program. It was found, however, that the closely connected question of linear independence of variables could give trouble, a false infinite maximum caused by round-off error. It was found that in the case where several variables were linearly dependent, as in the example above, that due to round-off error they did not appear to be so, and thus could occur in the basis simultaneously. To see how this could cause a false infinite maximum consider the example above. Suppose that $x_1 + y_1$, $x_2 + y_1$, and y_1 were all in the basis, although they are linearly dependent. Now the coefficients of these variables in the linear form z are c_1, c_2, and $-c_1-c_2$ respectively. If, however, $-c_1-c_2$ had acquired an excess, e, of round-off error, we could increase y_1 without bound causing z to increase also; i.e., if $y_1 = M$, a large number, then the computed z would look like

$$z^1 = c_1(x_1+M) + c_2(x_2+M) + (-c_2-c_1+e)M + \text{other terms}$$

$$= c_1 x_1 + c_2 x_2 + Me + \text{other terms},$$

in spite of the fact that x_1 and x_2 are such that restrictions (2) are satisfied. Since this false maximum actually occurred, a precaution should be taken against it. It is sufficient to decrease the coefficient of y_1 in the form (1) by an amount larger than any expected round-off error but not enough to seriously effect the value of z.

References

1. Charnes, Cooper, and Henderson, "An Introduction to Linear Programming".

2. Journal of Applied Mathematics, October, 1955.

Richard L. Bishop

RLB:gn