AN ALGORITHM FOR THE SOLUTION OF LINEAR PROGRAMING PROBLEMS

DONALD LEROY SPARKS

AN ALGORITHM FOR THE SOLUTION OF

LINEAR PROGRAMMING PROBLEMS

by

Donald Leroy Sparks
Captain, United States Army
B.S., Oklahoma State University, 1963

[redacted]

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 1968

ABSTRACT

Linear programming techniques are becoming of greater importance because the use of computerization has increased the fields for applications for linear programs. The primal-dual algorithm, in which the constraints are added one at a time, is investigated as a possible faster solution method. A computer program was developed to compare this method with the standard primal-dual algorithm using the full set of constraints at one time. Several random problems were solved using these two methods, and the results indicated a significant improvement in the solution time by the use of adding the constraints one at a time.

TABLE OF CONTENTS

I. INTRODUCTION

New linear programming algorithms have been developed to reduce the computational time in solving linear programs. The purpose of this thesis is to investigate the merits of one such new algorithm. This method consists of introducing the constraint equations one at a time. After each constraint is added, the "smaller", or submatrix, problem is solved using the primal-dual algorithm. This continues until all constraints have been added and a solution is obtained.

The rationale for this approach is that small matrices are used in the initial stages of solving the linear program; the size of the matrices increases only when additional constraints are introduced. If the number of iterations used in this method is not significantly different from the number of iterations used with the full matrices, the manipulation of the smaller matrices in the initial stages will reduce the solution time.

II. NOTATION

m number of constraint equations.

n number of legitimate variables.

A mxn matric of coefficients of the constraint equations with elements a_{ij}, $i = 1,\ldots,m$, $j = 1,\ldots,n$.

P mxm matrix of the basis vectors.

P^{-1} inverse of the basis.

P_j mx1 column vector which is the j^{th} column of A.

$$\text{i.e., } P_j = \begin{bmatrix} a_{ij} \\ \cdot \\ \cdot \\ \cdot \\ a_{mj} \end{bmatrix}, \; j = 1,\ldots,n \; .$$

P_{ai} mx1 column vector associated with the i^{th} artificial variable, $i = 1,\ldots,m$.

x_j mx1 column vector with elements x_{ij}, where $X_j = P^{-1}P_j$.

C nx1 column vector with elements c_j which are the costs of the legitimate variables.

B mx1 column vector with elements b_i which are the right-hand sides of the constraint equations.

s_j dual slack variables.

$\hat{s}_j$ dual slack variables after a dual iteration.

$\bar{d}$ mx1 column vector whose elements are the coefficients of the basis variables of the added constraint equation.

Notation used with tableau:

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	
P_{a0}	b_0-4	1	1/2	1/3	0	5/6	1	1	-1/6	$\theta = b_0 - 4$
P_3	4	0	1/2	2/3	1	1/6	0	0	1/6	
z_j-c_j	$4-b_0$	$\boxed{-1}$	-1/2	-1/3	0	-5/6	-1	-------		$\hat{\theta} = -4/-1 = 4$
s_j	$4b_0$	4	2	4	0	4	4	-------		
$\hat{s}_j$	16	0	0	8/3	0	2/3	0	-------		
		↑								

↑ the vector to be introduced into the basis. e.g., in this tableau P_0 will be introduced.

1 pivot element for a primal iteration, i.e., the θ criterion is $\theta = \min_i (x_{iB}/x_{ij})$ such that $x_{ij} > 0$.

$\boxed{-1}$ pivot element for a dual iteration, i.e., the $\hat{\theta}$-criterion is $\hat{\theta} = \min_j (-s_j/z_j-c_j)$ such that $z_j-c_j < 0$.

III. FORMULATION OF THE PROBLEM AND SOLUTION PROCEDURE

The general linear programming problem is to maximize

$$z = \sum_{j=1}^{n} c_j x_j$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j = b_i \geq 0, \; i = 1,\ldots,m \; , \qquad (1)$$

and

$$x_j \geq 0, \; j = 1,\ldots,n \; .$$

The *modified primal* uses an additional constraint

$$x_0 + \sum_{j=1}^{n} x_j = b_0 ,$$

where the cost of x_0 is zero and b_0 is arbitrarily large, so that for $x_0 > 0$ the constraint adds no additional restriction on (1).

The modified primal is written to maximize

$$z = \sum_{j=1}^{n} c_j x_j$$

subject to

$$\begin{aligned} x_0 + \sum_{j=1}^{n} x_j &= b_0 \; , \\ \sum_{j=1}^{n} a_{ij} x_j &= b_i , \; i = 1,\ldots,m \; , \end{aligned} \qquad (2)$$

and

$$x_j \geq 0, \; j = i,\ldots,n \; .$$

From (2) we can write the *modified dual* with slack variables, s_j, $j = 0,1,\ldots,n$, added. That is to minimize

$$w_0 b_0 + \sum_{i=1}^{m} w_i b_i$$

subject to

$$w_0 \qquad\qquad - s_0 = 0,$$

$$w_0 + \sum_{i=1}^{m} w_i a_{ij} - s_j = c_j, \; j = 1,\ldots,n, \qquad (3)$$

and

$$w_i \text{ unrestricted for } i = 0,1,\ldots,m.$$

The starting feasible solution to the primal dual algorithm is $w_i = 0$, $i = 1,\ldots,m$, and $w_0 = \max_j (c_j, 0)$.

For an optimal solution the complementary slackness condition must hold. That is

$$s_0 x_0 + \sum_{j=1}^{n} s_j x_j = 0 \ . \qquad (4)$$

Adding artificial variables, x_{ai}, $i = 0,1,\ldots,m$, to (2) with the cost of the artificial variables set to -1 and the cost of the legitimate variables set to zero, the <u>extended primal</u> can be written as

maximize

$$- x_{a0} - \sum_{i=1}^{m} x_{ai}$$

subject to

$$x_0 + \sum_{j=1}^{n} x_j + x_{a0} = b_0 \ ,$$

$$\sum_{j=1}^{n} a_{ij} x_j + x_{ai} = b_i, \; i = 1,\ldots,m,$$

and

$$x_j \geq 0, \; j = 1,\ldots,n, \text{ and } x_{ai} \geq 0, \; i = 0,\ldots,m \ .$$

Solving the extended primal is similar to using a Phase I Revised Simplex method.[1] However, in the primal-dual algorithm, when Phase I ends the linear program is solved because complementary slackness is maintained throughout the solution procedure.

If a new constraint is added to the tableau, complementary slackness is maintained without changing the dual slack variables. This can be shown as follows:

> Assume we have a feasible solution to the problem with k constraint equations. This means that $s_j = 0$ for all j such that $P_j \in P$ and $x_j = 0$ for all j such that $P_j \notin P$. These conditions imply that complementary slackness is maintained, and that the modified dual also has a feasible solution.
>
> Now we add the $k + 1^{st}$ constraint which introduces a new dual variable, w_{k+1}, but no new dual slack variables, s_j, $j = 0,1,\ldots,n$. We need a feasible solution to the modified dual for the enlarged system. Observe that we have a feasible solution if we set $w_{k+1} = 0$ since then the s_j, $j = 0,1,\ldots,n$, remain unchanged. In particular, $s_j = 0$ for all j such that $P_j \in P$, that is, for the legitimate variables. Also, $x_j = 0$ for all j such that $P_j \notin P$, which implies that we have maintained complementary slackness.

It is worth noting that the $z_j - c_j$, $j = 0,1,\ldots,n$, must be recalculated since the new constraint which is added to the extended primal starts with its artificial variable, $x_{a,k+1}$, in the basis with its cost set at -1.

The new basis is $\bar{P} = \begin{matrix} P & \bar{0} \\ \bar{d}^T & 1 \end{matrix}$, where $P_{a,k+1} = \begin{matrix} \bar{0} \\ 1 \end{matrix}$

is the artificial vector associated with $x_{a,k+1}$. Now we can solve the new k + 1 system using the primal-dual algorithm since complementary slackness has been maintained.

An optimal solution exists if and only if the following criteria are satisfied:

1. $z_j - c_j \geq 0$ for $j = 0,1,\ldots,n$,
2. $z_B - c_B = 0$, and
3. $x_0 > 0$.

The solution procedure is as follows:

The first tableau is set up using the first two constraints of the <u>extended primal</u> and the starting solution to the modified dual, which implies that at least one $s_j = 0$.

<u>Step 1</u>. Is there a j, say j_0, such that $s_{j0} = 0$ and $z_{j0} - c_{j0} < 0$?

a. Yes. Go to 2.

b. No. Go to 3.

<u>Step 2.</u> Introduce P_{j0} into the basis using the minimum θ-criterion and a primal iteration. Since the extended primal is bounded a pivot will always exist. Note that the s_j remain unchanged for all j. Go to 1.

<u>Step 3</u>. Is $z_j - c_j < 0$ for some j?

a. Yes. Go to 4.

b. No. Go to 5.

Step 4. Use the minimum $\hat{\theta}$-criterion. Is $\hat{\theta}$ bounded?

a. Yes. Perform a dual iteration to compute a new set of s_j's, say $\hat{s}_j$'s. Go to 1.

b. No. The linear program has no feasible solution. Stop.

Step 5. Is $z_B - c_B < 0$?

a. Yes. The linear program has no feasible solution. Stop.

b. No. Go to 6.

Step 6. Have all of the constraints been added?

a. Yes. Go to 9.

b. No. Go to 7.

Step 7. Introduce the next restraint, say the $k + 1^{st}$. Place the artificial vector $P_{a,k+1}$ in the basis. Compute $x_{B,k+1}$. Is $x_{B,k+1} \geq 0$?

a. Yes. Go to 8.

b. No. Multiply all coefficients of the new constraint, except for the artificial variable, by -1. This assures that $x_{B,k+1} \geq 0$ and the artificial variable is non-negative. Go to 8.

Step 8. For the system with k + 1 restraints, compute the new values of $z_j - c_j$ for $j = 0,1,\ldots,n$. Go to 1.

Step 9. Is $x_0 = 0$?

a. Yes. The linear program is unbounded. Stop.

b. No. An optimal solution has been found. Stop.

IV. SAMPLE PROBLEM

Consider the following example:

maximize

$$z = 2x_1 + 4x_3$$

subject to

$$\begin{aligned} 3x_1 + 4x_2 + 6x_3 + x_4 \quad &= 24 \ , \\ 4x_1 + 3x_2 + 12x_3 \quad + x_5 &= 24 \ , \\ x_1 + x_2 + 4x_3 \quad &= 8 \ , \end{aligned}$$

and

$$x_j \geq 0, \ j = 1,\ldots,5.$$

Then the extended primal is

maximize

$$- x_{a0} - x_{a1} - x_{a2} - x_{a3}$$

subject to

$$\begin{aligned} x_0 + x_1 + x_2 + x_3 + x_4 + x_5 + x_{a0} \quad &= b_0, \\ 3x_1 + 4x_2 + 6x_3 + x_4 \quad + x_{a1} \quad &= 24, \\ 4x_1 + 3x_2 + 12x_3 \quad + x_5 \quad + x_{a2} \quad &= 24, \\ x_1 + x_2 + 4x_3 \quad + x_{a3} &= 8, \end{aligned}$$

$x_j \geq 0$ for $j = 0,\ldots,5$, and $x_{ai} \geq 0$ for $i = 0,\ldots,3$.

The dual slack variables are $s_0 = \max_j(c_j,0) = 4$ with $j = 3$.

Then $s_B = x_0 b_0 = 4b_0$, and $s_j = s_0 - c_j$ for $j = 1,\ldots,5$, so that $s_1 = 2$, $s_2 = 4$, $s_3 = 0$, $s_4 = 4$, and $s_5 = 4$.

The starting tableau, using the first original constraint, is

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	
P_{a0}	b_0	1	1	1	1	1	1	1	0	
P_{a1}	24	0	3	4	6	1	0	0	1	$\theta = 24/6 = 4$
z_j-c_j	$-b_0-24$	-1	-4	-5	$\boxed{-7}$	-2	-1	---	---	
s_j	$4b_0$	4	2	4	0	4	4	---	---	
					↑					

From step 1, we see that $s_3 = 0$ and $z_3-c_3 < 0$. Using the minimum θ-criterion (as discussed in section III) in step 2, we introduce P_3 into the basis and remove P_{a1} from the basis.

Since there is no j_0 for which $s_{j0} = 0$ and $z_{j0}-c_{j0} < 0$, but $z_j-c_j < 0$ for several values of j, we arrive at step 4. Using the minimum $\hat{\theta}$-criterion a new set of s_j's, called $\hat{s}_j$'s are calculated.

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	
P_{a0}	b_0-4	1	1/2	1/3	0	5/6	1	1	-1/6	$\theta = b_0-4$
P_3	4	0	1/2	2/3	1	1/6	0	0	1/6	
z_j-c_j	$-b_0+4$	$\boxed{-1}$	-1/2	-1/3	0	-5/6	-1	---	---	$\hat{\theta} = -4/-1 = 4$
s_j	$4b_0$	4	2	4	0	4	4	---	---	
$\hat{s}_j$	16	0	0	8/3	0	2/3	0	---	---	
		↑								

Now $\hat{s}_0 = 0$ and $z_0-c_0 < 0$ so, from step 2, we introduce P_0 into the basis and remove P_{a0} from the basis.

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}
P_0	b_0-4	1	1/2	1/3	0	5/6	1	1	-1/6
P_3	4	0	1/2	2/3	1	1/6	0	0	1/6
z_j-c_j	0	0	0	0	0	0	0	---	---
s_j	16	0	0	8/3	0	2/3	0	---	---

From the above tableau we trace through steps 1b, 3b, 5b, 6b, and arrive at step 7. Note that we have obtained an optimal solution to the subproblem with one constraint. In step 7 we introduce the second constraint. Since P_0 and P_3 are basis vectors, $\bar{d}^T = (a_{20}, a_{23}) = (0,12)$; the new basis consists of P_0, P_3 and P_{a2}. With this basis we find that $x_{B2} = -24 < 0$ so that step 7b must be used. The second restraint is replaced by

$$- 4_{x1} - 3_{x2} - 12_{x3} - x_5 + x_{a2} = -24 ,$$

which is used throughout the remainder of the solution procedure. Note that now $\bar{d}^T = (0,-12)$ and $x_{B2} = 24 > 0$. New values of $x_j - c_j$ for $j = 0,1,\ldots,n$ are computed (step 8), and the new tableau is

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	P_{a2}	
P_0	b_0-4	1	1/2	1/3	0	5/6	1	1	-1/6	0	
P_3	4	0	1/2	2/3	1	1/6	0	0	1/6	0	$\theta = \frac{4}{1/2} = 8$
P_{a2}	24	0	2	5	0	2	-1	0	2	1	
z_j-c_j	-24	0	$\boxed{-2}$	-5	0	-2	1	---	---	---	
s_j	16	0	0	8/3	0	2/3	0	---	---	---	
			↑								

Since $s_1 = 0$ and $z_1-c_1 < 0$ (step 1) we go to step 2. Using a primal iteration, we introduce P_1 into the basis and eliminate P_3 from the basis.

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	P_{a2}	
P_0	b_0-8	1	0	-1/3	-1	2/3	1	1	-1/3	0	
P_1	8	0	1	4/3	2	1/3	0	0	1/3	0	
P_{a2}	8	0	0	7/3	-4	4/3	-1	0	4/3	1	$\theta = \frac{8}{4/3} = 6$
z_j-c_j	-8	0	0	-7/3	4	$\boxed{-4/3}$	1	---	---	---	$\hat{\theta} = \frac{-2/3}{-4/3} = 1/2$
s_j	16	0	0	8/3	0	2/3	0	---	---	---	
$\hat{s}_j$	12	0	0	3/2	2	0	1/2	---	---	---	
						↑					

Now $z_j-c_j \geq 0$ for all j for which $s_j = 0$. From steps 1b, 3a, and 4a, a new set of s_j's are computed. Then $\hat{s}_4$ becomes zero and $z_4-c_4 < 0$ so, from step 2, P_4 enters the basis and P_{a2} is removed from the basis.

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	P_{a2}
P_0	b_0-12	1	0	-3/2	1	0	3/2	1	-1	-1/2
P_1	6	0	1	3/4	3	0	1/4	0	0	-1/4
P_4	6	0	0	7/4	-3	1	-3/4	0	1	3/4
z_j-c_j	0	0	0	0	0	0	0	---	---	---
s_j	12	0	0	3/2	2	0	1/2	---	---	---

Optimality has now been obtained with the second restraint added. Following steps 1b, 3b, 5b, 6b, and 7, we introduce the third and final restraint. Since the basis vectors were P_0, P_1, and P_4, $\bar{d}^T = (a_{30}, a_{31}, a_{34}) = (0,1,0)$. The new basis vectors are P_0, P_1, P_4, P_{a3}. We find that $x_{B3} = 2 > 0$ so we go to step 8 and recompute $z_j - c_j$ for $j = 0,1,\ldots,n$.

The next sequence of steps is 1b, 3a, and 4a, which leads to a new set of s_j's.

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	P_{a2}	P_{a3}	
P_0	b_0-12	1	0	-3/2	1	0	3/2	1	-1	-1/2	0	
P_1	6	0	1	3/4	3	0	1/4	0	0	-1/4	0	
P_4	6	0	0	7/4	-3	1	-3/4	0	1	3/4	0	
P_{a3}	2	0	0	1/4	1	0	-1/4	0	0	1/4	1	$\theta = 2$
z_j-c_j	-2	0	0	-1/4	$\boxed{-1}$	0	1/4	---	---	---	---	$\theta = -2/-1 = 2$
s_j	12	0	0	3/2	2	0	1/2	---	---	---	---	
$\hat{s}_j$	8	0	0	1	0	0	1	---	---	---	---	
					↑							

Steps 1a and 2 bring P_3 into the basis with the elimination of P_{a3} from the basis. The new tableau is:

P	B	P_0	P_1	P_2	P_3	P_4	P_5	P_{a0}	P_{a1}	P_{a2}	P_{a3}
P_0	b_0-14	1	0	-7/4	0	0	7/4	1	-1	-3/4	-1
P_1	0	0	1	0	0	0	1	0	0	- 1	-3
P_4	12	0	0	5/2	0	1	-3/2	0	1	3/2	3
P_3	2	0	0	1/4	1	0	-1/4	0	0	1/4	1
z_j-c_j	0	0	0	0	0	0	0	---	---	---	---
s_j	8	0	0	1	0	0	1	---	---	---	---

This tableau is the final tableau since the sequence of steps 1b, 3b, 5b, 6a, and 9b inform us that we have found an optimal solution to the original linear program. Note that the complementary slackness condition has been maintained, that is, $s_0 x_0 + \sum_{j=1}^{n} s_j x_j = 0$. Therefore, the optimal solution is

$$x_0 = b_0 - 14 > 0,$$

$$x_1 = x_2 = x_5 = 0,$$

$$x_3 = 2, \text{ and}$$

$$x_4 = 12,$$

with the optimal cost $z = s_B = 8$.

V. PROGRAMMING TECHNIQUE

The linear programming technique described in this thesis was programmed in FORTRAN IV for use on the IBM 360/67 computer. One subroutine is used for the primal simplex iteration and another is used for the dual iteration. The final subroutine is used for the addition of constraints. The main (driving) routine is used to solve both the full linear program and the linear program using addition of constraints as described in this thesis. Using the main routine to solve both problems eliminates any time differences due to differences in programming techniques. Both of the solution procedures were timed*, and the number of iterations of each were counted. Read and print times were not included in the timing.

*The timing routine was developed by Lt. E.A. Singer, a student at the Naval Postgraduate School.

VI. EFFICIENCY OF THE ALGORITHM

To eliminate the considerable time and effort required to input data by hand, a subroutine was designed which generates random problems of a large size. This routine uses a random number generator to produce the elements of the A, B, and C matrices. The following criteria were used in order to insure the existence of a bounded feasible solution:

maximize

$$z = \sum_{j=1}^{n} c_j x_j$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j - x_{si} = b_i, \; i = 1,\ldots,m,$$

and

$$c_j \leq 0, \; x_j \geq 0, \; x_{si} \geq 0, \; a_{ij} \geq 0, \; b_i \geq 0$$

$$\text{for } i = 1,\ldots,m, \; j = 1,\ldots,n,$$

where x_{si} is the slack variable for the i^{th} constraint.

For this investigation the subroutine generated problems having 70 variables (including 20 slack variables), 20 constraint equations, and 50 cost coefficients using the following uniform distributions:

a_{ij}, uniform (0,1);

b_i, uniform (0,5);

c_j, uniform (-1,0).

A total of 40 problems were solved using the random problem generator described above. The execution times for these problems are given in Table I at the end of this section.

The comparison of solution times shows that all 40 problems ran faster using the method described in this thesis, than with the standard primal-dual algorithm. The time differences range from 15.17 seconds to 48.18 seconds with an average time difference of 27.53 seconds.

TABLE I

Prob. No. i	Full Array Time (x_i) (sec.)	Full Array Iter.	Addition of Constraints Time (y_i) (sec.)	Addition of Constraints Iter	$x_i - y_i$
1	87.82	23	56.50	21	31.32
2	87.84	23	56.91	23	30.93
3	80.22	21	56.53	21	23.69
4	87.93	23	56.70	22	31.23
5	84.06	22	58.61	23	25.45
6	91.63	24	56.62	22	35.01
7	95.45	25	58.65	23	36.80
8	80.31	21	56.52	21	23.79
9	99.60	26	65.67	32	33.93
10	80.16	21	56.51	21	23.65
11	80.18	21	56.50	21	23.68
12	91.65	24	65.12	29	26.53
13	80.19	21	57.99	22	22.20
14	91.74	24	64.35	30	27.39
15	80.20	21	56.48	21	23.72
16	84.02	22	57.10	22	26.92
17	95.52	25	59.36	24	36.16
18	83.95	22	58.71	22	25.24
19	84.14	22	65.47	25	18.67
20	99.25	26	70.03	28	29.22
21	107.08	28	58.90	25	48.18
22	87.85	23	56.51	21	31.34
23	84.01	22	61.39	24	22.62
24	80.16	21	56.52	21	23.64
25	80.17	21	56.49	21	23.68
26	80.16	21	56.50	21	23.66
27	80.19	21	56.52	21	23.67
28	84.21	22	58.84	23	25.37
29	103.27	27	57.85	24	45.42
30	87.83	23	59.47	23	28.36
31	84.07	22	59.33	22	24.74
32	88.19	23	56.92	23	31.27
33	80.28	21	56.56	21	23.72
34	87.96	23	56.56	21	31.40
35	84.11	22	57.09	23	27.02
36	84.05	22	56.56	21	27.49
37	80.20	21	56.51	21	23.69
38	80.18	21	56.54	21	23.64
39	84.03	22	68.86	26	15.17
40	80.29	21	56.54	21	23.75
Total	3454.01		2352.77		1101.24
Average	86.35		58.82		27.53

VII. SUMMARY AND CONCLUSIONS

A modification of the primal-dual algorithm has been presented. This modification differs from the standard primal-dual algorithm in that the constraint equations are introduced one at a time, and each subproblem is solved before the next constraint is added.

This algorithm was programmed in FORTRAN IV for the IBM 360/67. The program was designed so that any given linear program is solved first by the standard primal-dual algorithm, and then is resolved using the modified primal-dual procedure. Further, the same subroutines are used for both methods in order to eliminate timing bias due to coding differences. In fact the modified procedure contains steps which are not included in the timing of the standard routine.

In all cases the modified procedure was faster than the standard procedure. Of 40 test problems the standard method averaged 86.35 seconds per problem, whereas the modified method averaged 58.82 seconds per problem. One should not judge the actual running time of the test problems since no attempt was made to improve the efficiency of the computer program on an absolute basis; only the relative speeds of the two methods is of importance.

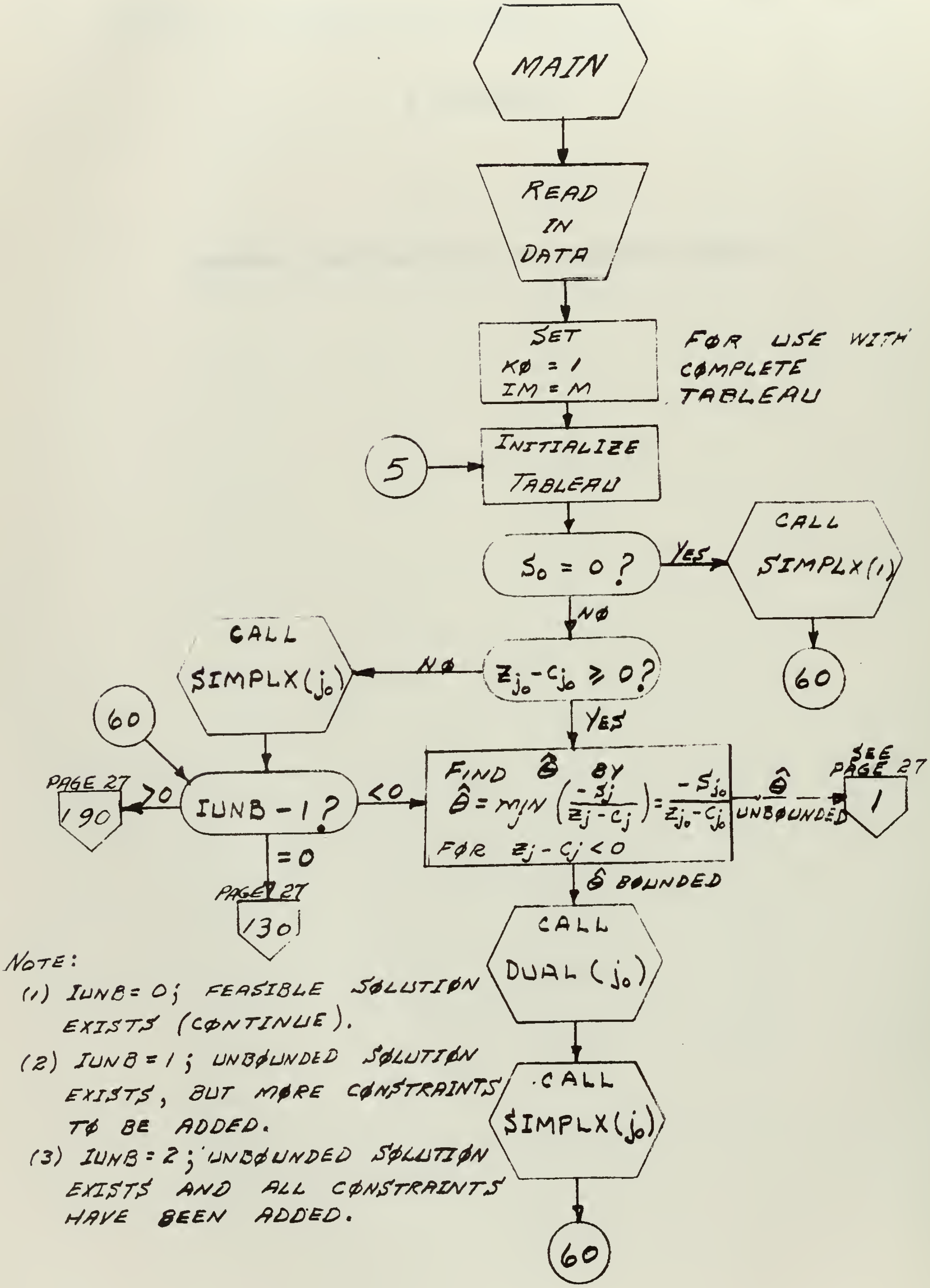
BIBLIOGRAPHY

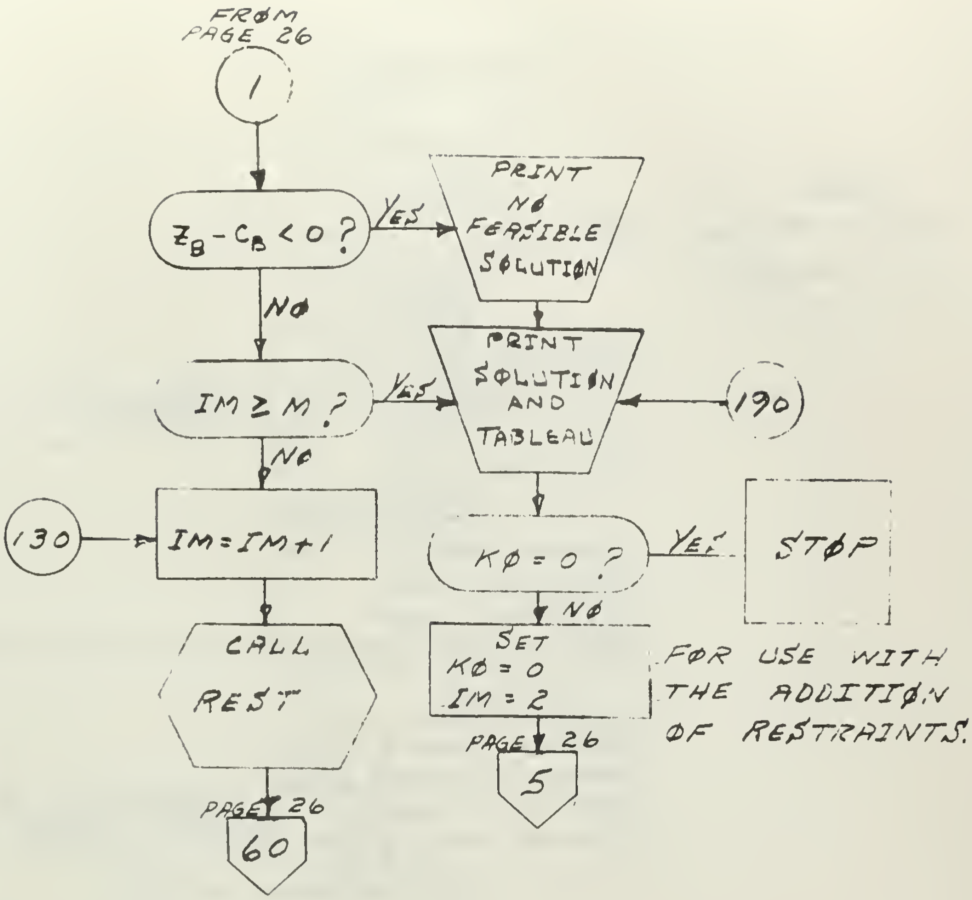
1. Hadley, G., Linear Programming. Reading, Mass: Addison-Wesley Publishing Co., Inc., 1962.

APPENDIX A

FLOW DIAGRAMS OF THE COMPUTER PROGRAM

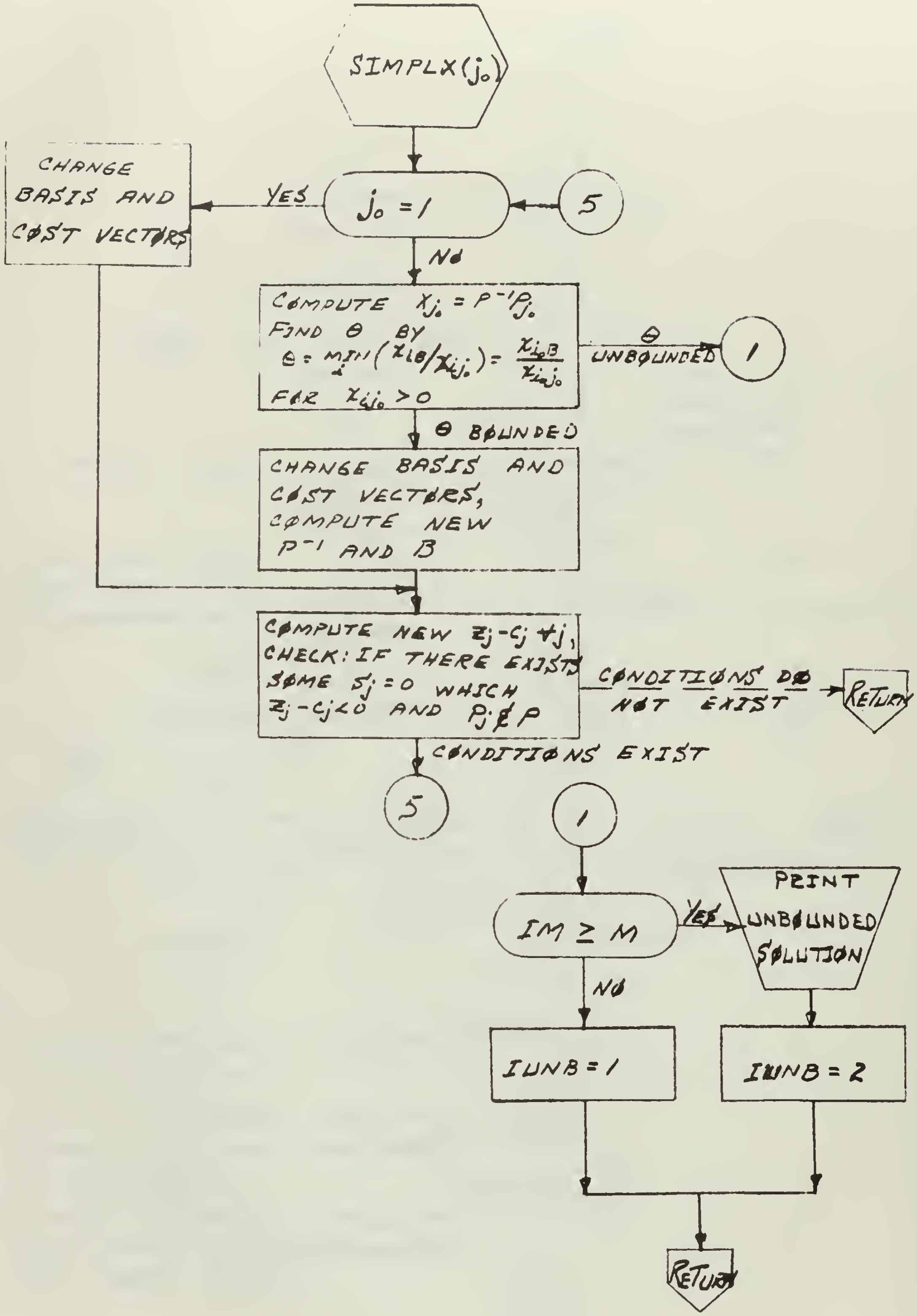
1. MAIN PROGRAM



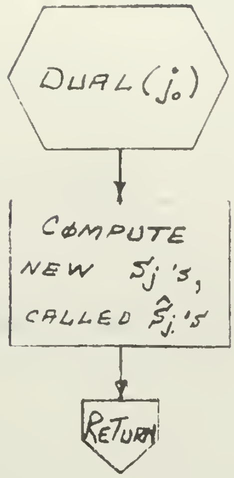


FRØM PAGE 26
1
$Z_B - C_B < 0$?
YES
PRINT NØ FEASIBLE SØLUTIØN
NØ
IM ≥ M ?
YES
PRINT SØLUTIØN AND TABLEAU
190
NØ
130
IM=IM+1
KØ = 0 ?
YES
STØP
NØ
CALL REST
SET KØ = 0 IM = 2
FØR USE WITH THE ADDITIØN ØF RESTRAINTS.
PAGE 26
5
PAGE 26
60

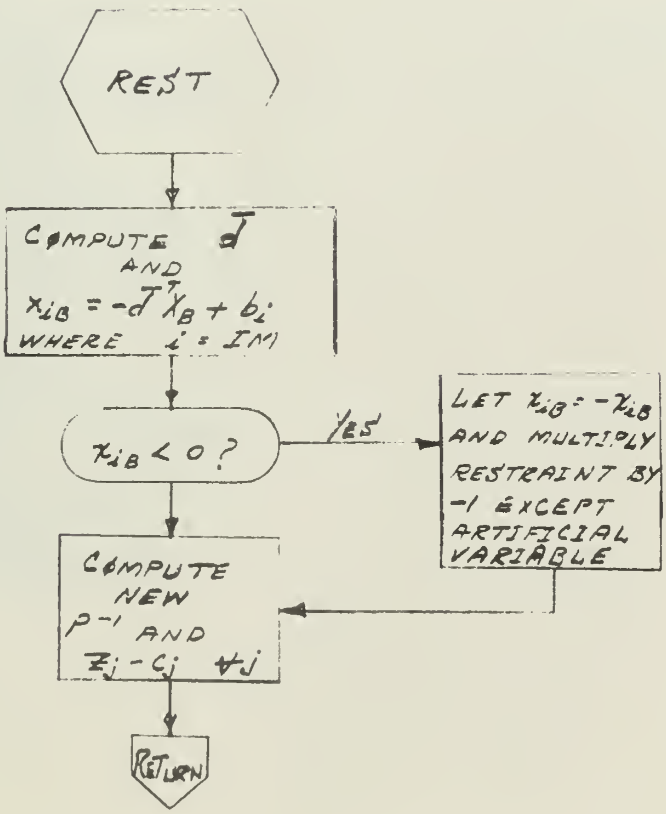
2. SIMPLEX ITERATION SUBROUTINE



3. DUAL ITERATION SUBROUTINE



4. ADDITION OF NEW RESTRAINT SUBROUTINE



APPENDIX B

FORTRAN LISTING OF THE COMPUTER PROGRAM

```
      DIMENSION P(53,151),B(52),C(151),BT(53,2),IBAS(53),ONE(53),TEM(2),
     1PINV(53,53),ZCB(2),ZC(151),WSB(2),WS(151),T(2),X(53),DBAR(53)
      COMMON/INT/IUNB,IM,M,N,IBAS,ITER
      COMMON/FLOAT/P,B,C,BT,ONE,TEM,PINV,ZCB,ZC,WSB,WS,T,X,DBAR,EP
C
C NOTE - WS(J) IN THIS PROGRAM AND S(J) IN THESIS ARE EQUIVALENT
C
      EP = 10.0E-6
      XE = URN(0)
      IPROB = 0
    6 IF(IPROB.GE.50) GO TO 500
      IPROB = IPROB +1
      CALL PSEUDO
      M = M + 1
      N = N + 1
C
C KO=1 IS FOR USE OF COMPLETE TABLEAU
C
      KO = 1
      IM = M
C
C INITIALIZE TABLEAU
C
    5 DO 1 I=2,M
      BT(I,1) = 0.0
      BT(I,2) = B(I-1)
    1 P(I,1) = 0.0
      ITER = 1
      DO 2 J=1,N
    2 P(1,J) = 1.0
      BT(1,1) = 1.0
      BT(1,2) = 0.0
      DO 3 I=1,M
      IBAS(I) = N+I
    3 ONE(I) = -1.0
      DO 10 I=1,M
      DO 10 J=1,M
      IF(I.EQ.J) GO TO 9
      PINV(I,J) = 0.0
      GO TO 10
    9 PINV(I,J) = 1.0
   10 CONTINUE
      CALL TIMEIT(0,CL)
      ZCB(1) = -1.0
      ZCB(2) = 0.0
      DO 11 I=2,IM
   11 ZCB(2) = ZCB(2) -BT(I,2)
      ZC(1) = -1.0
      DO 20 J =2,N
      ZC(J) = -1.0
      DO 20 I = 2,IM
   20 ZC(J) = ZC(J) -P(I,J)
      IND = 0
      WS(1) = 0.0
      DO 30 J=2,N
      IF(C(J).LE.WS(1))GO TO 30
      WS(1) = C(J)
      IND = J
   30 CONTINUE
      WSB(1) = WS(1)
      WSB(2) = 0.0
      DO 40 J=2,N
   40 WS(J) = WS(1) -C(J)
C
C DETERMINE VECTOR TO INTRODUCE
C
      IF(IND.EQ.0.0)GO TO 50
      IF(ZC(IND).GE.0.0)GO TO 70
      CALL SIMPLX(IND)
      GO TO 60
   50 CALL SIMPLX(1)
   60 IF(IUNB-1)70,130,190
   70 TEMP = 999999.
```

```
      C
      C FIND THETA FOR DUAL = MIN(-S(J)/ZC(J))OVER J FOR ZC(J)<0
      C
            IND = 0
            DO 80 J=1,N
            IF(ZC(J).GE.0.0)GO TO 80
            DO 75 I=1,IM
            IF(IBAS(I).EQ.J) GO TO 80
         75 CONTINUE
            XE= -WS(J)/ZC(J)
            IF(TEMP.LE.XE)GO TO 80
            TEMP = XE
            IND = J
         80 CONTINUE
            IF(IND.EQ.0) GO TO 100
            CALL DUAL(IND)
            CALL SIMPLX(IND)
            GO TO 60
      C
      C CHECK FOR INFEASIABILITY
      C
        100 IF(ZCB(1))200,110,120
        110 IF(ZCB(2).LT.0.0) GO TO 200
        120 IF(IM.GE.M) GO TO 190
      C
      C ADD NEXT RESTRAINT
      C
        130 IM = IM + 1
            CALL REST
            IUNB = 0
            GO TO 60
        190 CALL TIMEIT(-1,CL)
            WRITE(6,5000)IPROB,CL,ITER
       5000FORMAT(10X,'PROBLEM',I4,'  TIME IS ',-6PF15.6,'  SECONDS WITH',
          1I6,'  ITERATIONS',//)
            IF(KO.EQ.0)GO TO 6
      C
      C KO=0 IS FOR USE OF ADDITION OF RESTRAINTS
      C
            IM=2
            KO=0
            GO TO 5
        500 STOP
        200 WRITE(6,4010)
            GO TO 190
       4010 FORMAT(10X,'SOLUTION INFEASIBLE'//)
            END
```

```
          C
          C
          C
0001            SUBROUTINE SIMPLX(J)                                                0001090
0002           0DIMENSION P(53,151),B(52),C(151),BT(53,2),IBAS(53),ONE(53),TEM(2),  0001100
               1PINV(53,53),ZCB(2),ZC(151),WSB(2),WS(151),T(2),X(53),DBAR(53)       0001110
0003            COMMON/INT/IUNB,IM,M,N,IBAS,ITER                                    0001120
0004            COMMON/FLOAT/P,B,C,BT,ONE,TEM,PINV,ZCB,ZC,WSB,WS,T,X,DBAR,EP        0001125
0005            IUNB = 0                                                            0001130
          C
          C FOR J=1 THE ONLY CHANGE IS THE BASIS AND ZC(J)'S
          C
0006          5 IF(J.EQ.1) GO TO 130                                                0001140
          C
          C COMPUTE X(J) AND FIND THETA FOR PRIMAL = MIN(X(I,B)/X(I,J)) OVER J
          C    FOR X(I,J)>0
          C
0007            T(1) = 999999.                                                      0001150
0008            T(2) = T(1)                                                         0001160
0009            DO 20 I=1,IM                                                        0001170
0010            L = IM-I+1                                                          0001180
0011            X(L) = 0.0                                                          0001190
0012            DO 10 K=1,IM                                                        0001200
0013            Y = PINV(L,K)*P(K,J)                                                0001205
0014            X(L) = X(L) + Y                                                     0001210
0015         10 CALL ROUND(X(L),Y,EP)                                               0001211
0016            IF(X(L).LE.0.)GO TO 20                                              0001215
0017            TEM(1) = BT(L,1)/X(L)                                               0001220
0018            TEM(2) = BT(L,2)/X(L)                                               0001230
0019            IF(T(1)-TEM(1))20,11,12                                             0001240
0020         11 IF(T(2).LE.TEM(2))GO TO 20                                          0001250
0021         12 T(1) = TEM(1)                                                       0001260
0022            T(2) = TEM(2)                                                       0001270
0023            ID = L                                                              0001280
0024         20 CONTINUE                                                            0001290
          C
          C COMPUTE NEW TABLEAU
          C
0025            IF(T(1).EQ.999999.)GO TO 50                                         0001300
0026            IBAS(ID) = J                                                        0001310
0027            ONE(ID) = 0.0                                                       0001320
0028            DO 30 I=1,IM                                                        0001330
0029            IF(I.EQ.ID) GO TO 30                                                0001333
0030            Y = T(1)*X(I)                                                       0001333
0031            BT(I,1) = BT(I,1) - Y                                               0001334
0032            CALL ROUND(BT(I,1),Y,EP)                                            0001335
0033            Y = T(2)*X(I)                                                       0001336
0034            BT(I,2) = BT(I,2) - Y                                               0001336
0035            CALL ROUND(BT(I,2),Y,EP)                                            0001337
0036            XE = X(I)/X(ID)                                                     0001360
0037            DO 30 K=1,IM                                                        0001370
0038            Y = PINV(ID,K)*XE                                                   0001371
0039            PINV(I,K) = PINV(I,K) - Y                                           0001372
0040            CALL ROUND(PINV(I,K),Y,EP)                                          0001375
0041         30 CONTINUE                                                            0001395
0042            BT(ID,1) = BT(ID,1)/X(ID)                                           0001387
0043            BT(ID,2) = BT(ID,2)/X(ID)                                           0001388
0044            DO 31 I=1,IM                                                        0001386
0045         31 PINV(ID,I) = PINV(ID,I)/X(ID)                                       0001389
0046         70 ZCB(1) = 0.0                                                        0001390
0047            ZCB(2) = 0.0                                                        0001410
0048            DO 80 I=1,IM                                                        0001420
0049            ZCB(1) =ZCB(1) +ONE(I)*BT(I,1)                                      0001430
0050         80 ZCB(2) =ZCB(2) +ONE(I)*BT(I,2)                                      0001440
0051            ZC(1) = ONE(1)                                                      0001450
0052            DO 90 L=2,N                                                         0001460
0053            ZC(L) = 0.                                                          0001470
0054            DO 90 I=1,IM                                                        0001490
0055            DO 90 K=1,IM                                                        0001500
0056            Y = ONE(I)*PINV(I,K)*P(K,L)                                         0001505
0057            ZC(L) = ZC(L) + Y                                                   0001510
0058         90 CALL ROUND(ZC(L),Y,EP)                                              0001515
          C
```

```
      C CHECK FOR AN ADDITION ITERATION WITH PRIMAL
      C
            TEMP = 0.0
            DO 120 L=1,N
            IF(WS(L).NE.0.0) GO TO 120
            DO 110 I=1,IM
            IF(IBAS(I).EQ.L)GO TO 120
        110 CONTINUE
            IF(ZC(L).GE.TEMP) GO TO 120
            J=L
            TEMP = ZC(L)
        120 CONTINUE
            ITER = ITER + 1
            IF(TEMP.NE.0.0) GO TO 5
        200 RETURN
        130 IBAS(1) = 1
            ONE(1) = 0.0
            ID = 1
            GO TO 70
      C
      C SOLUTION UNBOUNDED
      C
         50 IF(IM.GE.M) GO TO 60
            IUNB = 1
            GO TO 200
         60 WRITE(6,4020)
            IUNB = 2
       4020 FORMAT(10X,'SOLUTION UNBOUNDED'//)
            GO TO 200
            END
```

```
             C
             C
             C
0001               SUBROUTINE DUAL(J)                                                0001750
0002              0DIMENSION P(53,151),B(52),C(151),BT(53,2),IBAS(53),ONE(53),TEM(2), 0001760
                  1PINV(53,53),ZCB(2),ZC(151),WSB(2),WS(151),T(2),X(53),DBAR(53)     0001770
0003               COMMON/INT/IUNB,IM,M,N,IBAS,ITER                                  0001780
0004               COMMON/FLOAT/P,B,C,BT,ONE,TEM,PINV,ZCB,ZC,WSB,WS,T,X,DBAR,EP      0001785
             C
             C COMPUTE NEW WS(J) WHICH IS EQUIVALENT TO S(J)
             C
0005               XE = WS(J)/ZC(J)                                                  0001790
0006               DO 10 K=1,N                                                       0001800
0007               Y = ZC(K)*XE                                                      0001801
0008               WS(K) = WS(K) - Y                                                 0001802
0009            10 CALL ROUND(WS(K),Y,EP)                                            0001803
0010               Y = ZCB(1)*XE                                                     0001804
0011               WSB(1) = WSB(1) - Y                                               0001805
0012               CALL ROUND(WSB(1),Y,EP)                                           0001806
0013               Y = ZCB(2)*XE                                                     0001807
0014               WSB(2) = WSB(2) - Y                                               0001808
0015               CALL ROUND(WSB(2),Y,EP)                                           0001810
0016               RETURN                                                            0001840
0017               END                                                               0001850
```

```
            C
            C
            C
 0001             SUBROUTINE REST                                                 0001860
 0002           ODIMENSION P(53,151),B(52),C(151),BT(53,2),IBAS(53),ONE(53),TEM(2), 0001870
                1PINV(53,53),ZCB(2),ZC(151),WSB(2),WS(151),T(2),X(53),DBAR(53)    0001880
 0003             COMMON/INT/IUNB,IM,M,N,IBAS,ITER                                0001890
 0004             COMMON/FLOAT/P,B,C,BT,ONE,TEM,PINV,ZCB,ZC,WSB,WS,T,X,DBAR,EP    0001895
            C
            C COMPUTE D-BAR AND B VECTOR
            C
 0005             K = IM-1                                                        0001900
 0006             BT(IM,1) = 0.0                                                  0001910
 0007             BT(IM,2) = B(K)                                                 0001920
 0008             DO 10 I=1,K                                                     0001930
 0009             IF(IBAS(I).GT.N) GO TO 5                                        0001935
 0010             L = IBAS(I)                                                     0001940
 0011             DBAR(I) = -P(IM,L)                                              0001950
 0012             BT(IM,1) = BT(IM,1) + DBAR(I)*BT(I,1)                           0001960
 0013             BT(IM,2) = BT(IM,2) + DBAR(I)*BT(I,2)                           0001970
 0014             GO TO 10                                                        0001971
 0015           5 DBAR(I) = 0.0                                                   0001972
 0016          10 CONTINUE                                                        0001973
 0017             IF(BT(IM,1))50,20,30                                            0001980
 0018          20 IF(BT(IM,2).LT.0.0) GO TO 50                                    0001990
            C
            C COMPUTE NEW INVERSE AND COST VECTOR
            C
 0019          30 DO 40 J=1,K                                                     0002000
 0020             DO 40 I=1,K                                                     0002010
 0021          40 PINV(IM,J) = PINV(IM,J) + DBAR(I)*PINV(I,J)                     0002020
 0022             ZC(1) = ONE(1)                                                  0002021
 0023             DO 45 J=2,N                                                     0002022
 0024             ZC(J) =0.0                                                      0002023
 0025             DO 45 I=1,IM                                                    0002024
 0026             DO 45 IK=1,IM                                                   0002025
 0027             Y = ONE(I)*PINV(I,IK)*P(IK,J)                                   0002026
 0028             ZC(J) = ZC(J) + Y                                               0002027
 0029          45 CALL ROUND(ZC(J),Y,EP)                                          0002028
 0030             ZCB(1) = ZCB(1) - BT(IM,1)                                      0002029
 0031             ZCB(2) = ZCB(2) - BT(IM,2)                                      0002030
 0032         100 RETURN                                                          0002031
            C
            C USED TO INSURE FEASIABILITY BY MAKING THE B COMPONENT NON-NEGATIVE
            C
 0033          50 BT(IM,1) = -BT(IM,1)                                            0002040
 0034             BT(IM,2) = -BT(IM,2)                                            0002050
 0035             DO 60 I=1,K                                                     0002060
 0036          60 DBAR(I) = -DBAR(I)                                              0002070
 0037             PINV(IM,IM) = -1.0                                              0002130
 0038             GO TO 30                                                        0002140
 0039             END                                                             0002150
```

```
          C
          C
          C
0001            SUBROUTINE ROUND(A,B,EP)                                         0002160
          C  A=STARTING VALUE, B=ADDED VALUE, EP= LOWEST ROUND-OFF DESIRED        0002170
0002            IF(A.EQ.0.0.OR.B.EQ.0.0) GO TO 200                               0002180
0003            IF(ABS(A/B).GT.EP) GO TO 200                                     0002190
0004            A = 0.0                                                          0002200
0005        200 RETURN                                                           0002210
0006            END                                                              0002220
```

```
               C
               C
               C
 0001                SUBROUTINE TIMEIT(N,TIME)                                         0002230
               C
               C  N=0 STARTS CLOCK, N=-1 STOPS CLOCK
               C
 0002                IT=N+2                                                            0002270
 0003                GO TO (20,10),IT                                                  0002280
 0004             10 CALL TIMON(M)                                                     0002290
 0005                TIMEM=M                                                           0002300
 0006                RETURN                                                            0002310
 0007             20 CALL TIMOFF(M)                                                    0002320
 0008                TIME=M                                                            0002330
 0009                TIME=(TIMEM-TIME)*26.0                                            0002340
 0010                RETURN                                                            0002350
 0011                END                                                               0002360
```

```
C
C
C
      SUBROUTINE PSEUDO                                                 0002370
     0DIMENSION P(53,151),B(52),C(151),BT(53,2),IBAS(53),ONE(53),TEM(2), 0002380
     1PINV(53,53),ZCB(2),ZC(151),WSB(2),WS(151),T(2),X(53),DBAR(53)     0002290
      COMMON/INT/IUNB,IM,M,N,IBAS,ITER                                  0002400
      COMMON/FLOAT/P,B,C,BT,ONE,TEM,PINV,ZCB,ZC,WSB,WS,T,X,DBAR,EP      0002410
C
C M = NUMBER OF CONSTRAINTS, N = NUMBER OF VARIABLES INCLUDING M SLACK
C     VARIABLES
C MAXIMUM M = 50, MAXIMUM N = 150
C
C CHANGE STATEMENTS 1 AND 2 TO DESIRED PROBLEM SIZE
C
    1 M = 20
    2 N = 70
      IM = M + 1
      IN = N + 1
      K1 = N - M + 1
      K2 = K1 + 1
      C(J) = -URN(1)                                                    0002430
      DO 20 J=2,K
      IF(C(J).LE.-.001) GO TO 10
      C(J) = 0.0                                                        0002450
   10 DO 20 I=2,IM
      P(I,J) = URN(1)                                                   0002470
      IF(P(I,J).GT.0.001) GO TO 20                                      0002480
      P(I,J) = 0.0                                                      0002490
   20 CONTINUE                                                          0002500
      DO 30 I=1,M
      B(I) = URN(1)*5.0                                                 0002520
      IF(B(I).GT.0.01) GO TO 30                                         0002530
      B(I) = 0.0                                                        0002540
   30 CONTINUE                                                          0002550
      DO 40 J=K2,IN
      C(J) =0.0
      DO 40 I = 2,IM
   40 P(I,J) = 0.0                                                      0002580
      DO 50 I=2,IM
      K = N- M + I
   50 P(I,K) = -1.0
      RETURN                                                            0002640
      END                                                               0002650
```

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library Naval Postgraduate School Monterey, California 93940	2
3.	Chief of Naval Operations (OP-96) Department of the Navy Washington, D. C. 20350	1
4.	Department of the Army Civil Schools Branch, OPO, OPD Washington, D. C. 20315	1
5.	Department of Operations Analysis Naval Postgraduate School Monterey, California	1
6.	Professor Rex H. Shudde (Thesis Advisor) Operations Analysis Department Naval Postgraduate School Monterey, California 93940	1
7.	Lt. E. E. Flesher, Jr. COMASWFORPAC FPO San Francisco 96610	1
8.	Capt. Donald L. Sparks 5131 E. 30th Place Tulsa, Oklahoma 74114	1

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

AN ALGORITHM FOR THE SOLUTION OF LINEAR PROGRAMMING PROBLEMS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Thesis

5. AUTHOR(S) (First name, middle initial, last name)

SPARKS, Donald Leroy, Captain, USA

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1968	41	1

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT [redacted]

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California

13. ABSTRACT

Linear programming techniques are becoming of greater importance because the use of computerization has increased the fields for applications for linear programs. The primal-dual algorithm, in which the constraints are added one at a time, is investigated as a possible faster solution method. A computer program was developed to compare this method with the standard primal-dual algorithm using the full set of constraints at one time. Several random problems were solved using these two methods, and the results indicated a significant improvement in the solution time by the use of adding the constraints one at a time.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Unclassified
Security Classification

A-31408

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
LINEAR PROGRAMMING (COMPUTERIZED)						
PRIMAL-DUAL ALGORITHM						

[redacted]