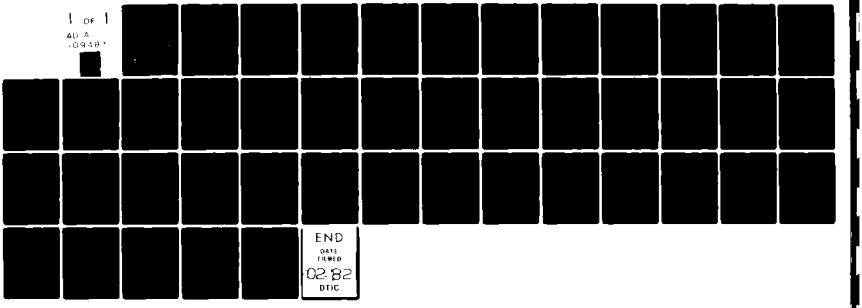
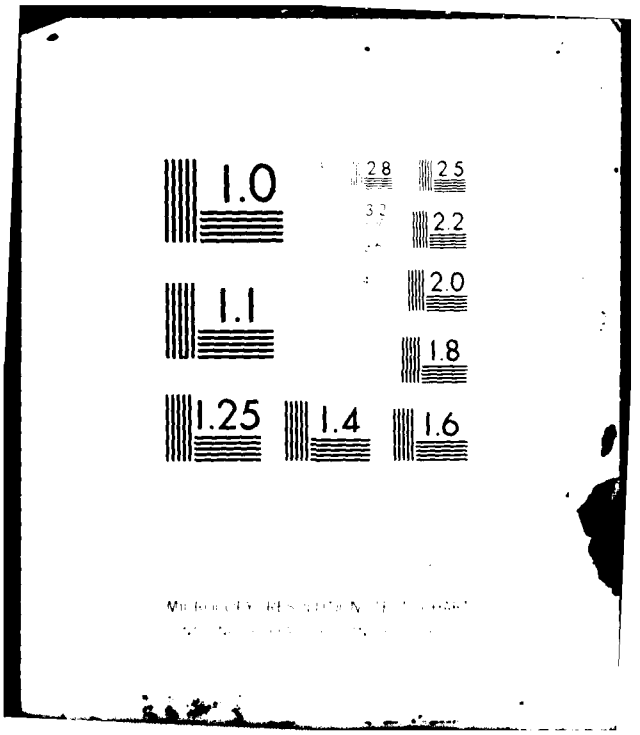
AD-A109 487 NAVAL SURFACE WEAPONS CENTER SILVER SPRING MD F/G 12/1
LINOPT: A FORTRAN ROUTINE FOR SOLVING LINEAR PROGRAMMING PROBLE--ETC(U)
OCT 81 J W WINGATE
UNCLASSIFIED NSWC/TR-80-413 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5

(12) LEVEL

NSWC TR 80-413

AD A109487

LINOPT, A FORTRAN ROUTINE FOR SOLVING LINEAR PROGRAMMING PROBLEMS

BY JOHN W. WINGATE

RESEARCH AND TECHNOLOGY DEPARTMENT

9 OCTOBER 1981

NAVAL SURFACE WEAPONS CENTER

Dahlgren, Virginia 22448 • Silver Spring, Maryland 20910

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: NSWC TR 80-413	2. GOVT ACCESSION NO.: AD-A109 487
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): LINOPT, A FORTRAN ROUTINE FOR SOLVING LINEAR PROGRAMMING PROBLEMS	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): John W. Wingate	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: NAVAL SURFACE WEAPONS CENTER (R44), WHITE OAK, SILVER SPRING, MD 20910	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 65856N, R0128-SB, R0128-SB, 2R44EA501
11. CONTROLLING OFFICE NAME AND ADDRESS	12. REPORT DATE: 9 October 1981
	13. NUMBER OF PAGES: 46
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Linear Programming; Dual Simplex method; Minimum ℓ_1-norm problems

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report documents a FORTRAN routine LINOPT for solving linear programming problems. Upper and lower bounds on all variables are permitted, and the dual problem includes as a special case linearly-constrained minimum ℓ_1-norm problems. Basic theory, the algorithm used, input-output procedures and examples of use are included.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 55 IS OBSOLETE

S/N 0102-014-6601

UNCLASSIFIED

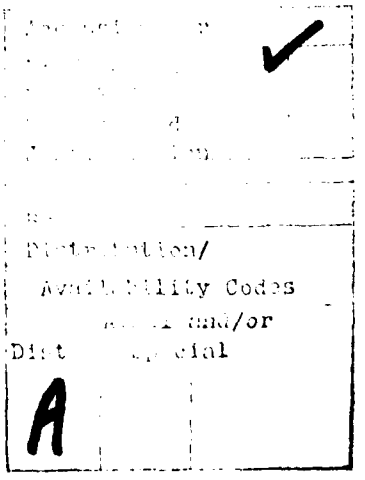
SECURITY CLASSIFICATION OF THIS PAGE When Data Entered

FOREWORD

This report documents a FORTRAN routine LINOPT for solving linear programming problems. Upper and lower bounds on all variables are permitted, and the dual problem includes as a special case linearly-constrained minimum l_1-norm problems. Basic theory, the algorithm used, input-output procedures and examples of use are included.

Ira M Blatstein

IRA M. BLATSTEIN
By direction



CONTENTS

ILLUSTRATIONS

TABLES

CHAPTER 1

INTRODUCTION

REPORT ORGANIZATION

This report documents a FORTRAN subroutine (with associated subroutines) called LINOPT for solving linear programming (LP) problems. It is divided into several chapters. Following the INTRODUCTION is a chapter (PROBLEM FORMAT AND PROGRAM USE) explaining the types of problems which can be solved by LINOPT, some manipulations on them and correspondences with program notation. A chapter (SUPPORT FUNCTIONS AND DUALITY IN LINEAR PROGRAMMING) discusses some of the duality concepts behind the formulation and solution of LP problems. The next chapter, DUAL SIMPLEX METHOD gives some information about the algorithm used in the program. EXAMPLES and a LISTING follow.

PROBLEM FORMULATION

A rather abstract formulation of an LP problem in the following: Let X be a real vector space paired with M under the bilinear form (inner product) $(\mu,x) \longrightarrow \mu\cdot x$, $\mu\epsilon M$, $x\epsilon X$, and let Y_i, $i = 1,.....m$ be similarly paired with Λ_i, $i = 1,..., m$. Given $\mu\epsilon M$, closed convex sets C_i in Y_i, and linear transformations A_i, : $X \rightarrow Y_i$, $i = 1,...,m$,

$$\begin{aligned} &\text{maximize} && \mu\cdot x \\ &\text{subject to} && A_i x \in C_i,\ i=1,...,m. \end{aligned}$$

This problem in convex programming has the following as its dual problem:

$$\text{minimize} \sum_{i=1}^{m} \sigma_{C_i}(\lambda_i)$$

$$\text{subject to} \sum_{i=1}^{m} A_i^{*}\lambda_i = \mu$$

where σ_{C_i} is the support function of the set C_i — see Chapter 3 for more on support functions. For explanatory purposes it is sufficient to take each A_i equal to the identity, so that $Y_i = X$, $\Lambda_i = M$, $i = 1,...,m$. Chapter 3 treats this simplified version.

The program LINOPT is set up to handle constraints of the form $\mu_i\cdot x \in C_i$ where $\mu_i \in M$ and C_i is a nonempty closed interval. If some C_i is a bounded interval of nonzero length, the dual problem has a nonlinear objective; it is, however, convex and piecewise linear. The details are given in Chapter 2, where it is shown that the general form of the dual objective is the sum of two terms, one linear, and one a weighted ℓ_1 norm.

The algorithm used in the program is a form of the revised dual simplex algorithm modified to handle upper and lower bound constraints. The inverse matrix used is a row-basis inverse. Accordingly, the algorithm is more efficient on problems with many constraints. (On problems with fewer dependent variables than independent variables, a column-basis inverse would be smaller.) Use of a row basis has definite advantages in modifying a problem and then reoptimizing.

No new theory is involved in this program. The dual simplex algorithm can be found in standard linear programming texts.[1,2] Insisting that all variables in the primal problem have both upper and lower bounds makes it trivial to find a dual-feasible point to start the algorithm, and because of the resulting asymmetry between primal and dual problems, allows us to handle directly (via the dual problem) certain piecewise linear convex minimization problems.

The results from convex analysis used in Chapter 3 can be found in greater generalization and detail in Rockafellar's book.[3]

---

[1]Hadley, G., Linear Programming, Addison-Wesley, Reading, 1962.

[2]Simmonard, M., Linear Programming, Prentice-Hall, Englewood Cliffs, 1966.

[3]Rockafellar, R. T., Convex Analysis, Princeton University Press, Princeton, 1970.

CHAPTER 2

PROBLEM FORMAT AND PROGRAM USE

INTRODUCTION

LINOPT is programmed to solve a problem maximizing (or minimizing) a linear function subject to upper and lower bounds on linear constraint functions. (The bounds are equal for an equality constraint.) The dual to this problem has a piecewise linear objective function and linear equality constraints. Missing bounds in the primal problem correspond to sign constraints on the dual variables. Such missing bounds can be handled by introducing a penalty function for the sign constraint violations in the dual problem. An even simpler and more direct interpretation is that the missing bounds can be replaced by bounds so large in magnitude that they are effectively infinite.

PRIMAL PROBLEM

Maximize x_{k_0} subject to $\underline{b}_k \le x_k \le \bar{b}_k$, $k=1,\ldots, m+n$, where $k_0 \in \{1,\ldots, m+n\}$ and

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

The objective variable x_{k_0} can be expressed by:

$$x_{k_0} = \sum_{j=1}^{n} c_j\, x_j,$$

where

$$c_j = \begin{cases} \delta_{k_0 j} & \text{if } k_0 \le n, \\ a_{k_0-n,\,j} & \text{if } k_0 > n. \end{cases}$$

The same data also define the dual problem.

DUAL PROBLEM

$$\text{Minimize} \sum_{k=1}^{m+n} \max\{u_k \underline{b}_k,\ u_k \bar{b}_k\} \text{ subject to}$$

$$u_j + \sum_{i=1}^{m} u_{n+i}\, a_{ij} = c_j,\ j=1,\ldots,n.$$

The dual objective has another form which is more likely to be recognized in an application:

$$\sum_{k=1}^{m+n} \max\{u_k \underline{b}_k,\ u_k \bar{b}_k\} = \sum_{k=1}^{m+n} \left(\frac{\bar{b}_k + \underline{b}_k}{2}\right) u_k + \sum_{k=1}^{m+n} \left(\frac{\bar{b}_k - \underline{b}_k}{2}\right) |u_k|$$

Thus the dual objective contains a linear term and a weighted ℓ_1-norm term.

The dual variable u_k can be thought of as a Lagrange multiplier for the constraint $x_k \in [\underline{b}_k, b_k]$.

MISSING BOUNDS; SIGN CONSTRAINTS

The following table (TABLE 1) shows how to prepare primal problems with missing constraints or dual problems with sign constraints. M is a very large positive number. (The default value supplied by the program is 10^{100}.) The first line of the table gives the standard two-sided constraint assumed by the program. The other lines give the modifications for unilateral and no constraints. In the modified problem u_k is always unconstrained in sign. The dual objective picks up the original linear term when the sign constraint is satisfied, and a penalty term when the sign constraint is violated.

When the program gives an optimal solution in which $x_k = \pm M$ for some k, the original problem has an unbounded solution. If it has a finite solution, the program will yield it, and it will not depend on M (unless M has been set so small that it interferes with the "legitimate" constraints). The calculations are arranged so that roundoff errors due to the disparity in magnitude between M and the original data do not propagate from iteration to iteration, and appear within an iteration only if some $x_k = \pm M$.

The objective variable x_{k_0} is also formally a constrained variable, although generally the constraint will be $-M \leq x_{k_0} \leq M$, i.e. essentially no constraint at all. Tighter bounds may at times be useful. The constraint $\underline{b}_{k_0} \leq x_k \leq M$ can be used to answer the question: Is $\max x_{k_0} \geq \underline{b}_{k_0}$? If the answer to this question is negative, the constraints are inconsistent. As soon as the inconsistency is detected, the program returns to the calling program without going on to calculate the solution completely.

TABLE 1 FILLING IN MISSING BOUNDS

Original constraints on x_k	Original sign constraints on u_k	Original dual objective term	Modified constraints	Modified dual objective term
$\underline{b}_k \leq x_k \leq \bar{b}_k$	u_k unconstrained in sign	$\max\{u_k\underline{b}_k,\ u_k\bar{b}_k\}$	$\underline{b}_k \leq x_k \leq \bar{b}_k$	$\max\{u_k\underline{b}_k, u_k\bar{b}_k\}$
$x_k \leq \bar{b}_k$	$u_k \geq 0$	$u_k\bar{b}_k$	$-M \leq x_k \leq \bar{b}_k$	$\max\{u_k(-M), u_k\bar{b}_k\}$
$x_k \geq \underline{b}_k$	$u_k \leq 0$	$u_k\underline{b}_k$	$\underline{b}_k \leq x_k \leq M$	$\max\{u_k\underline{b}_k, u_kM\}$
x_k unconstrained	$u_k = 0$	0	$-M \leq x_k \leq M$	$M\lvert u_k\rvert$

PROGRAM NOTATION

Correspondence between the notation herein and the notation of LINOPT is given in Table 2.

TABLE 2 CORRESPONDING VARIABLES

This TR	LINOPT
m	M
n	N
k_0	IOBJ
x_k	X(K)
u_k	U(K)
$\underline{b}_k$	BL(K)
$\bar{b}_k$	BU(K)
a_{ij}	A(ROW(I) + COL(J))
M	BIGM

The FORTRAN variable BIGM is included in the program for the user's convenience. It supplies a default value (which can be changed) for filling in the missing bounds. (The user must fill in all bounds, since there is no provision for keeping track of missing bounds otherwise.)

CONSTRAINT COEFFICIENT STORAGE

The constraint coefficients are referenced in an unusual but flexible way. Row and column pointer arrays ROW and COL are used to index an array A. The FORTRAN standard for array storage is by columns (a_{11}, a_{21}, a_{31},..., a_{12}, a_{22}, a_{32},..., etc.) Suppose that we have a matrix A stored in an array dimensioned 10 x 20 and we wish to study a problem whose constraint coefficients form a submatrix of A, as in

$$\begin{bmatrix} y_5 \\ y_8 \end{bmatrix} = \begin{bmatrix} a_{57} & a_{53} & a_{59} \\ a_{87} & a_{83} & a_{89} \end{bmatrix} \begin{bmatrix} z_7 \\ z_3 \\ z_9 \end{bmatrix}$$

We can set $x_1 = z_7$, $x_2 = z_3$, $x_3 = z_9$, $x_4 = y_5$, $x_5 = y_8$,

ROW(1) = 5, ROW(2) = 8,
COL(1) = 60, COL(2) = 20, COL(3) = 80.

COL(J) is set to the number of elements in the array preceeding the coefficient column for X(J)— 6 x 10 for the 7th column, 2 x 10 for the 3rd and 8 x 10 for the 9th. ROW(I) then picks off the appropriate entry in the column. We can even introduce (by using the LOCF function)columns extraneous to the array storing A. Consider the modified example:

$$\begin{bmatrix} y_5 \\ y_8 \end{bmatrix} = \begin{bmatrix} a_{57} & a_{53} \\ a_{87} & a_{84} \end{bmatrix} \begin{bmatrix} z_7 \\ z_3 \end{bmatrix} + \begin{bmatrix} b_{54} \\ b_{84} \end{bmatrix} w_4$$

in which $x_3 = w_4$. COL(3) is set by

COL(3) = LOCF (B1,4)) - LOCF (A(1,1)).

The columns of B must be structured the same way as the columns of A for this procedure to work. Instead of having a coefficient matrix stored in the array in the usual way we could have it transposed, perhaps as a result of starting with a dual problem. This corresponds to storage by rows (not the FORTRAN standard). A little reflection indicates that defining ROW the way COL is defined above, and COL the way ROW is defined handles this storage arrangement. Further examples are given in the program comments.

INPUT AND OUTPUT

Input and output variables are clearly indicated in the program comments. (The program is listed in the section titled "LISTINGS".) Arrays are passed as formal parameters. Scalers are passed by using a labelled common block /XXXLP/, which must accordingly be a common block in the calling program.

ROUNDOFF CONTROL

In the program there are three input variables which can be used to control roundoff error accummulations. EPS is a tolerance used in checking constraint violations. H is also used to zero out coefficients in the tableau which have small nonzero values (typically for a CDC 60-bit machine, on the order of 10^{-14}) which ought to vanish. Finally constraint violations less than EPS are eliminated from the optimal solution before returning. For small problems EPS = 0 is generally all right.

The other roundoff - controlling variables are invert (a logical variable) and ITMAX. When INVERT is TRUE the inverse matrix corresponding to the current key K is calculated. ITMAX is a limit on the number of iterations. When this limit is reached control is returned to the calling LINOPT again with INVERT = TRUE., one may control the building of roundoff error in the inverse matrix (which otherwise is updated by column operations every iteration). (For small problems this may not be necessary.)

After obtaining a solution (or after any return from an initial call to LINOPT) INVERT can be set to FALSE for another call to LINOPT. Certain modifications to the problem data are permissible at such a time - constraints may be added, for example. These modifications are any for which the inverse matrix would be unchanged, and include the following (Note: primary indices: K(1),...,K(N); secondary K(N + 1),...,K(N + M) - see Chapter 4.)

ADDING CONSTRAINTS

M is increased, new elements to ROW are added to point to the new constraint coefficients (which, if not already defined should be stored appropriately), and new bounds added to BL and BU.

MODIFYING CONSTRAINTS

Bounds for any secondary variable can be changed. Inactive bounds for primary variables can be changed. Active bounds for primary variables can be changed, provided the corresponding solution is also changed; e.g. if X(K(1)) = *BU(K(1)) and BU(K(1)) is changed, X(K(1)) must be changed in the same way.* Constraint coefficients for dependent variables X(N + 1),..., X(N + M) which are also secondary variables can be changed; (these may include the objective variable) or such constraints can be dropped, with appropriate changes to ROW, M, BL and BU. (If the constraint corresponding to ROW(I) is dropped, the simplest way to make these changes is to set ROW(I) = ROW(M), BL(N + I) = BL(N + M), BU(N + I) = BL(N + M), and then M = M - 1, so that the indexing for X(N + M) *is changed to* X(N + I)).

Of course, when LINOPT is recalled with INVERT = TRUE, any problem changes whatsoever are permissible.

CHAPTER 3

SUPPORT FUNCTIONS AND DUALITY IN LINEAR PROGRAMMING

The support function σ_C of a convex set C in X is a convex function defined on M by:

$$\sigma_C = (\mu) = \sup_{x \in C} \mu \cdot x \qquad (1)$$

The support function of the empty set is $-\infty$ everywhere. For nonempty C, $\sigma_C(\mu) > -\infty$ and may take the value $+\infty$; in Rockafellar's terminology, it is a proper convex function[4].

Many of the formulas of convex analysis can be simplified when they are restricted to convex polyhedra and convex polyhedral functions. One such is found in Corollary 16.4.1 of Rockafellar's book[5], from part of which we can derive the following: Let $C_1, \ldots, C_m$ be closed convex polyhedra with nonempty intersection C (also a convex polyhedron). Then

$$\sigma_C(\mu) = \begin{cases} \min \sum_{i=1}^{m} \sigma_{C_i}(\mu_i) \\ \text{subject to} \sum_{i=1}^{m} \mu_i = \mu \end{cases} \qquad (2)$$

(The general version of the corollary is required if C is empty.) Rockafellar terms the operation in (2) "infimal convolution", since for m = 2, $\sigma_{C_1 \cap C_2}(\mu) = \inf_\lambda (\sigma_{C_1}(\lambda) + \sigma_{C_2}(\mu - \lambda))$ a form reminiscent of integral convolutions. Formulas (1) and (2) express $\sigma_C(\mu)$ as the common optimal value of a pair of dual convex programming problems:

Primal Problem:

Maximize $\mu \cdot x$

subject to $x \in C_i$, $i = 1, \ldots, m$.

Dual problem:

Minimize $\sum_{i=1}^{m} \sigma_{C_i}(\mu_i)$

subject to $\sum_{i=1}^{m} \mu_i = \mu$.

4. Ibid, p. 24.
5. Ibid, p. 146.

Three solution cases arise:

1. $\sigma_C(\mu) = -\infty$: C is empty. The dual objective is unbounded below on the dual constraint set.

2. $-\infty < \sigma_C(\mu) < +\infty$: C is not empty. The optimal value, viz $\sigma_C(\mu)$, is attained in both problems.

3. $\sigma_C(\mu) = +\infty$: C is not empty. The primal objective is unbounded above on the primal constraint set. The dual objective is $+\infty$ everywhere on the dual constraint set, (i.e. $\sigma_{C_i}(\mu_i) < \infty$, $i = 1,\ldots,m \Rightarrow \Sigma_{i=1}^{m} \mu_i \neq \mu$.

Suppose that $\sigma_C(\mu)$ is finite, and let x^* solve the primal, μ_i^*, $i=1,\ldots,m$, the dual. Then

$$\begin{aligned}
\sigma_C(\mu) &= \mu \cdot x^* && \text{(primal optimality)}\\
&= (\Sigma_{i=1}^{m} \mu_i^*) \cdot x^* && \text{(dual constraint)}\\
&= \Sigma_{i=1}^{m} (\mu_i^* \cdot x^*)\\
&\leq \Sigma_{i=1} \sigma_{C_i}(\mu_i^*) && \text{(primal constraints)}\\
&= \sigma_C(\mu) && \text{(dual optimality)}
\end{aligned}$$

It follows that

$$\mu_i^* \cdot x^* = \sigma_{C_i}(\mu_i^*),\ i=1,\ldots,m, \tag{3}$$

or that μ_i^* supports C_i at x^*: μ_i^* is an outer normal to C_i at x^*. The formula is one way of expressing complementary slackness, since if $x^* \in \text{int } C_i$ then $\mu_i^* = 0$, while

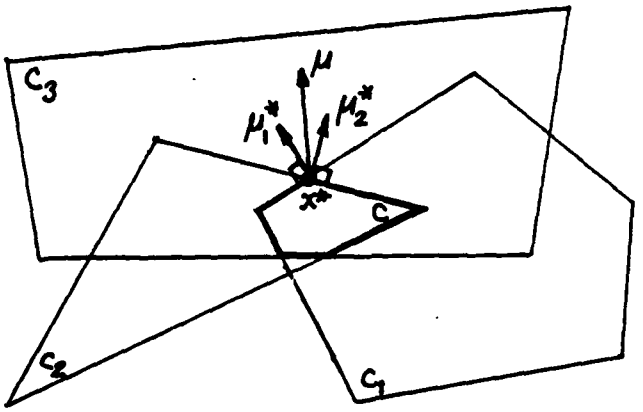


Figure 1 ILLUSTRATION OF DUAL OPTIMAL SOLUTIONS. ($\mu_3^* = 0$)

if $\mu_i^* = 0$, then the constraint $x \in C_i$ can be dropped without altering the solution. The set of constraints for which $\mu_i^* \neq 0$ are active (binding) at the optimum. (See Figure 1). There may be multiple solutions, each with a different set of active constraints.

The usefulness of (2) hinges on picking the constraint sets C_i, $i=1,\ldots,m$, to be simple enough to permit easy evaluation of their support functions. Any convex polyhedron can be expressed as the intersection of half-spaces, and any half-space can be defined by a linear inequality. Let $H := \{x: \mu \cdot x \leq b\}$, where $\mu \neq 0$. Then

$$\sigma_H(\nu) = \begin{cases} ub \text{ if } \nu = u\mu \text{ and } u \geq 0, \\ +\infty \text{ otherwise.} \end{cases}$$

In terms entirely of hyperplane constraints the primal and dual problems become
Primal problem:

Maximize $\mu \cdot x$

subject to $\mu_i \cdot x \leq b_i$, $i=1,\ldots,m$

Dual problem:

Minimize $\sum_{i=1}^{m} u_i b_i$

subject to $\sum_{i=1}^{m} u_i \mu_i = \mu$,
$u_i \geq 0$, $i=1,\ldots,m$.

The sign constraints on u_i avoid infinite values of the dual objective and keep it linear so that both problems consist of optimizing linear functionals subject to linear constraints. Alternatively we could omit the sign constraints and keep the formulation in terms of support functions. This pair of problems also illustrates the more general duality relationship cited in the INTRODUCTION. Let $A_i: X \to R$ be defined by $A_i X := \mu_i \cdot X$. Then $A_i^*: R \to M$ and $A_i^* u = u\mu_i$. Moreover,

$$\sigma_{(-\infty, b_i]}(u) = \begin{cases} ub_i \text{ if } u \geq 0, \\ +\infty \text{ if } u \leq 0. \end{cases}$$

Thus the problems are expressible as:

Primal problem:

Maximize $\mu \cdot x$

subject to $A_i x \in (-\infty, b_i]$, $i=1,\ldots,m$

Dual problem:

Minimize $\sum_{i=1}^{m} \sigma_{(-\infty, b_i]}(u_i)$

subject to $\sum_{i=1}^{m} A_i^* u_i = \mu$

We may have two-sided constraints such as

$$\underline{b} \leq \mu \cdot x \leq \bar{b}$$

(where $\underline{b} \leq \bar{b}$) defining a set S in X, which can be replaced by the pair of constraints

$$\begin{cases} \mu \cdot x \leq \bar{b}, \\ -\mu \cdot x \leq -\underline{b}. \end{cases}$$

since

$$\sigma_S(\nu) = \begin{cases} \max \{u\underline{b}, u\bar{b}\} \text{ if } \nu = u\mu, \\ +\infty \text{ otherwise,} \end{cases}$$

and $\max \{u\underline{b}, u\bar{b}\}$ is nonlinear in u (unless $\underline{b} = \bar{b}$), the dual of a problem with two-sided constraints is nonlinear. Of course it is easy to relate the two-sided and one-sided versions by using (2). Thus if u is the dual variable for the constraint $\underline{b} \leq \mu \cdot x \leq \bar{b}$, u^+ for $\mu \cdot x \leq b$ and u^- for $-\mu \cdot x \leq -\bar{b}$,

then
$$u = u^+ - u^- \tag{4}$$

while if $\underline{b} \neq \bar{b}$ either u^+ or u^- vanishes (at the solution), so that

$$u^+ = \max \{u,0\}, \; -u^- = \min \{u,0\}. \tag{5}$$

When $\underline{b} = \bar{b}$, only the difference $u = u^+ - u^-$ is determined. An alternate viewpoint in this case is that u is the dual variable for the linear equality constraint $\mu \cdot x = \bar{b}$.

The pivoting operations of the dual simplex method can be thought of as substituting one hyperplane bounding a half-space for another, and consequently are better suited for the formulation in terms of one-sided constraints. The relations (4) and (5) and some sign bookkeeping then make it easy to apply the method to two-sided constraints. Explanations of the method without the sign manipulations are more transparent. Accordingly in the next section only one-sided constraints are considered.

CHAPTER 4

DUAL SIMPLEX METHOD

In the previous section no particular coordinates were used on X. Most LP problems encountered are expressed in terms of coordinates with respect to some particular basis for X, the coordinates then forming a set of independent variables. (as indicated previously, LINOPT assumes such a formulation.) Thus, with one-sided constraints, we get a pair of problems like the following, in which we assume that the n independent variables x_j, $j\epsilon J$ are included in the m+n constrained variables x_k, $k\epsilon K$; i.e. $J\subset K$.

Primal problem:

$$\text{Maximize } x_0 := \Sigma_{j\epsilon J}\, c_j\, x_j$$

$$\text{subject to } x_k := \Sigma_{j\epsilon J}\, a_{kj}\, x_j \leq b_k,\ k\epsilon K.$$

(Note that $a_{kj} = \delta_{kj}$ for $k\epsilon J$.)

Dual problem:

$$\text{Minimize } \Sigma_{k\epsilon K}\, u_k\, b_k$$

$$\text{subject to } \Sigma_{k\epsilon K}\, u_k\, a_{kj} = c_j,\ j\epsilon J$$

$$u_k \geq 0,\ k\epsilon K.$$

Note that the n equations relating the dual variables u_k, $k\epsilon K$ can be written explicitly for u_j:

$$\Sigma_{i\epsilon K\sim J}\, u_i\, a_{ij} + u_j = c_j$$

Thus in the dual problem, u_i, $i\epsilon K\sim J$, are independent and u_j, $j\epsilon J$, are dependent. Given some other subset J' of K for which x_j, $j\epsilon J'$, are linearly independent, we can transform the constraint relations so that x_j, $j\epsilon J'$ are the independent variables through which the primal problem is phrased. Given such as index set J' we can define a corresponding basic solution. For J the definition of a basic solution is obtained by setting the independent variables x_j, $j\epsilon J$, and u_i, $i\epsilon K\sim J$, to their bounds and satisfying the constraint relations among the variables; $\bar{x}_k$ and $\bar{u}_k$ are the values of x_k, u_k at the basic solution.

$\bar{x}_j = b_j,\ j\epsilon J$	(primary primal variables)
$\bar{x}_i = \Sigma_{j\epsilon J}\, a_{ij}\, b_j,\ i\epsilon K\sim J$	(secondary primal variables)
$\bar{u}_j = c_j,\ j\epsilon J$	(secondary dual variables)
$\bar{u}_i = 0,\ i\epsilon K\sim J$	(primary dual variables)

The terms primary and secondary have been introduced instead of independent and dependent because one may wish to refer to the primary variables of the problem as initially formulated as the independent variables. The split between primary and secondary depends on the set J and changes with it. The secondary primal indices are usually called basic indices in linear programming texts, since they correspond to the indices for a column basis for the constraint matrix. This terminology is a little inappropriate here, since LINOPT makes use of a row basis corresponding to the complementary set of indices - the dual basic indices in the usual description. The use of "basic" in this sense is avoided here to prevent confusion.

Furthermore, in a problem with two-sided constraints the basic indices refer to the indexing of the equivalent one-sided problem, not the indexing of the two-sided problems, so that the basic variables for J would be x_k and $-x_k$ for $k \in K \sim J$ and either x_j or $-x_j$ (but not both) for $j \in J$. Alternatively we may retain the "primary/secondary" notation and supplement it with some way of indicating whether a primary variable is at its upper or its lower bound. (The program simply checks the solution value against the bounding values.

In a basic solution the primary variables satisfy the constraints placed on them if all primal variables satisfy the constraints, the basic solution and J are primal-feasible. If the dual constraints are satisfied, the basic solution is dual-feasible. A basic solution which is both primal- and dual-feasible is optimal. At a basic solution both primal and dual objective variables have the value $\Sigma_{j \in J} c_j b_j$. The criterion for dual feasibility is simply that $c_j \geq 0$, $j \in J$.

The transformation of the constraint coefficients accompanying a change from one set of primary variables to another can be performed explicitly when needed, or it can be expressed in terms of a nonsingular matrix relating the variables.

There are two ways of doing this. Let x_j, $j \in J$ and $x_{j'}$, $j' \in J'$ be two sets of primary primal variables. Set $I = K \sim J$, $I' = K \sim J'$, and let x_J be a column vector whose entries are x_j, $j \in J$, etc. Using matrix notation the two ways can be described as follows:

1. Solution for $x_{I'}$

$$x_I = A x_J \qquad (A \text{ is } m \times n)$$

Rearrange columns to give:

$$B\, x_{I'} = R\, x_{J'}$$

with B a nonsingular m x m submatrix of [I -A]. Then $x_{I'} = B^{-1} R\, x_{J'}$.

The columns making up B are a basis for the space spanned by the columns of [I -A]
Applied to the dual:

$$U_I A + U_J = C$$
$$U_{I'} S + U_{J'} D = C$$
$$U_{I'} SD^{-1} + U_{J'} = \bar{C}, \text{ where } \bar{C} = C\, D^{-1}.$$

The rows of D form a basis for the space spanned by the rows of $\begin{bmatrix} A \\ I \end{bmatrix}$.

2. Solution for x_J and substitution:

$$\begin{bmatrix} X_I \\ X_J \end{bmatrix} = \begin{bmatrix} A \\ I \end{bmatrix} \quad X_J$$

Rearrange rows to give:

$$\begin{bmatrix} X_{I'} \\ X_{J'} \end{bmatrix} = \begin{bmatrix} S \\ D \end{bmatrix} \quad X_J = \begin{bmatrix} SD^{-1} \\ I \end{bmatrix} \quad X_{J'}$$

(S and D are the same as above. $B^{-1}R = SD^{-1}$.)

For the dual:

$$\begin{aligned} [U_I\ U_J] &= U_I[I \quad -A] + [0 \quad C] \\ [U_{I'}\ U_{J'}] &= U_I[B \quad -R] + [C_{I'}\ C_{J'}] \\ [U_{I'}\ U_{J'}] &= U_{I'}[I \quad -B^{-1}R] + [0 \quad \bar{C}] \end{aligned}$$

where $\bar{C} = C_{J'} + C_{I'} B^{-1}R$ and is the same as before. The inverse matrix B^{-1} is the product of elementary row operations; D^{-1} is the product of elementary column operations. Either one may be used to keep track of changes. LINOPT uses D^{-1} and generates coefficients and solutions from the original *constraint coefficients by:*

$$\begin{bmatrix} X_I \\ X_J \end{bmatrix} = \begin{bmatrix} A \\ I \end{bmatrix} \quad D^{-1}\ X_{J'}$$

When a basic solution is changed, D^{-1} is updated by column operations. (We have ignored x_0: assumme that $0 \in K$ so that c is a row of $\begin{bmatrix} A \\ I \end{bmatrix}$)

The dual simplex algorithm works with dual-feasible basic solutions. Given a set J defining primary primal variables, J is altered by adding an index not in J and dropping an index in J: a secondary variable replaces a primary variable. The resulting changes in the constraint coefficients can be accomplished by Gauss-Jordan pivoting. The indices entering and leaving J are chosen in such a way that dual-feasibility is maintained and the objective function value does not increase. Proof of convergence can be found in any linear programming text.[6] The procedure is:

1. Identify violated constraints: $I' = \{ i \in K \sim J: \ X_i > b_i \}$.
2. If I' is empty, stop: basic solution is optimal.
3. Pick (by some heuristic) $k \in I'$.
4. Identify constraints which can be dropped without being violated when x_k is set to b_k: $J' := \{ j \in J: \ a_{kj} > 0 \}$
5. If J' is empty, stop: constraints are inconsistent.

[6] For example, Hadley, op. cit, or Simmonard, op. cit.

6. Identify subset of J' corresponding to constraints for which dual feasibility is maintained when dropped: $J'' = \{j \in J' : c_j/a_{kj} = \min_{j \in J} (cj/a_{kj})\}$.
7. Pick (while applying anticycling criterion, if desired) $\ell \in J''$.
8. Update solution: $J := J \cup \{k\} \sim \{\ell\}$. Update inverse D^{-1} by column operations. Calculate new basic solution.
9. Go to 1.

The coefficients a_{kj} and c_j are those corresponding to the current index set J, not the original one in terms of which the problem is phrased, and are calculated by post-multiplying an original constraint matrix row (or objective row) by D^{-1}.

In the program the heuristic used in step 3 is to pick the most violated constraint. In step 7 a tie for ℓ is broken randomly, a procedure which prevents cycling almost surely.

CHAPTER 5

EXAMPLES

TEST PROGRAM

A small test program to run the following examples is listed in Figure 2. The lines between the call to LINOPT and the call to TABLO merely do some cosmetic surgery on the output, replacing quantities near M in magnitude (actually those $\geq \sqrt{M}$) by $\pm$ R (machine infinities). Three examples are given, with NAMELIST inputs and the outputs from the program. Note that in all three, ROW and COL are defined to correspond to storage by rows, and the columns of the array A contain the rows of the constraint coefficient matrix.

EXAMPLE 1

This example is essentially the problem discussed in Section 1-3 of Hadley[7], with slightly modified coefficients.

$$\text{Maximize } x_8: = 5.0\,x_1 + 7.6\,x_2 + 8.0\,x_3 + 4.0\,x_4$$

$$\text{subject to} \quad x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0,\ x_4 \geq 0,$$

$$x_5: = 1.5\,x_1 + 1.2\,x_2 + 2.4\,x_3 + 1.2\,x_4 \leq 2100,$$

$$x_6: = 1.0\,x_1 + 4.5\,x_2 + 1.0\,x_3 + 3.0\,x_4 \leq 8000,$$

$$x_7: = 1.5\,x_1 + 3.0\,x_2 + 3.6\,x_3 + 1.0\,x_4 \leq 5000.$$

[7] Op. cit

NAMELIST INPUT:

```
$IN
IOBJ = 8
M = 4,
N = 4,
MIN = .FALSE.,
INVERT = .TRUE.,
ITMAX = 1000,
EPS = 0.,
ROW = 0, 10, 20, 30,
COL = 1, 2, 3, 4,
A(1, 1) = 1.5, 1.2, 2.4, 1.2,
A(1, 2) = 1.0, 4.5, 1.0, 3.0,
A(1, 3) = 1.5, 3.0, 3.6, 1.0,
A(1, 4) = 5.0, 7.6, 8.0, 4.0,
BL = 4*0., 4*-1.E100,
BU = 4*1.E100, 2100., 8000., 5000., 1.E100,
K = 1, 2, 3, 4, 5, 6, 7, 8,
X = 4*0.,
$END
```

(Since the objective coefficients c_j are all nonzero for the initial tableau, it is not really necessary to preset x_1, x_2, x_3 and x_4, as Example 2 will show).

Output:

TABLEAU

I	BL(I)	x(I)	BU(I)	T(I, 1)	T(I, 3)	T(I, 5)	T(I, 7)
1	0.000	0.000	R	1.000	0.000	0.000	0.000
2	0.000	1625.000	R	-.125	-.800	-.417	.500
3	0.000	0.000	R	0.000	1.000	0.000	0.000
4	0.000	125.000	R	-1.125	-1.200	1.250	-.500
5	-R	2100.000	2100.000	0.000	.000	1.000	-.000
6	-R	7687.500	8000.000	-2.938	-6.200	1.875	.750
7	-R	5000.000	5000.000	-.000	.000	.000	1.000
8	-R	12850.000	R	-.450	-2.880	1.833	1.800

The objective variable, x(8), is to be maximized. ITER = 10, IERR = 0.

The tableau gives information about the primary and secondary variables at the final iteration. The primary variables are x_1, x_3, x_5 and x_7. The rows of the tableau give the coefficients of the variables expressed in terms of the primary variables. Thus, $x_4 = -1.125\ x_1 - 1.2\ x_3 + 1.125\ x_5 - .5\ x_7$.
The dual variables are not printed explicitly but the nonzero ones can be obtained from the objective row: $u_1 = -.450$, $u_3 = -2.880$, $u_5 = 1.833$, $u_7 = 1.800$. (For a minimization problem, these should be negated.)

EXAMPLE 2

This problem also comes from Hadley[8]. It is his Problem 8-5.

Minimize x_4: = $3x_1 - 2x_2 + 4x_3$
subject to $x_1 \geq 0$, $x_2 \geq 0$, $x_3 \geq 0$,
x_5: = $3x_1 + 5x_2 + 4x_3 \geq 7$,
x_6: = $6x_1 + x_2 + 3x_3 \geq 4$,
x_7: = $7x_1 - 2x_2 - x_3 \leq 10$,
x_8: = $x_1 - 2x_2 + 5x_3 \geq 3$,
x_9: = $4x_1 + 7x_2 - 2x_3 \geq 2$.

Input:

```
   $IN
   IOBJ = 4,
   M = 6,
   N = 3,
   MIN = . TRUE.,
   INVERT = . TRUE.,
   ITMAX = 1000,
   EPS = 0.,
   ROW = 0, 10, 20, 30, 40, 50,
   COL = 1, 2, 3,
   A(1, 1) = 3., -2., 4.,
   A(1, 2) = 3., 5., 4.,
   A(1, 3) = 6., 1., 3.,
   A(1, 4) = 7., -2., -1.,
   A(1, 5) = 1., -2., 5.,
   A(1, 6) = 4., 7., -2.,
   BL = 3*0., -1.E100, 7., 4., -1.E100, 3., 2.,
   BU = 6*1.E100, 10., 2*1.E100,
   K = 1,2,3,4,5,6,7,8,9,
$END
```

Output:

TABLEAU

I	BL(I)	X(I)	BU(I)	T(I, 1)	T(I, 5)	T(I, 8)
1	0.000	0.000	R	1.000	1.000	0.000
2	0.000	R	R	-.333	.152	-.121
3	0.000	R	R	-.333	.061	.152
4	-R	-R	R	2.333	-.061	.848
5	7.000	R	R	-.000	1.000	.000
6	4.000	R	R	4.667	.333	.333
7	-R	-R	10.000	8.000	-.364	.091
8	3.000	3.000	R	.000	0.000	1.000
9	2.000	R	R	2.333	.939	-1.152

THE OBJECTIVE VARIABLE, X(4), IS TO BE MINIMIZED.
ITER = 3 IERR = 0

8. Ibid., p. 267.

This problem has an unbounded solution: $X_5 = +\infty$ and the objective value is $-\infty$.

EXAMPLE 3

This example illustrates the solution of a dual problem.

$$\text{Minimize} \sum_{k=3}^{9} |u_k|$$

$$\text{subject to} \sum_{k=3}^{9} u_k = 1,$$

$$\sum_{k=3}^{9} k\, u_k = 1.$$

(The indexing starts at 3 for convenience.) Since there is no unit matrix in the constraint coefficient matrix, we add artificial variables u_1 and u_2, which must vanish at the solution:

$$u_1 + \sum_{k=3}^{9} u_k = 1,$$

$$u_2 + \sum_{k=3}^{9} k u_k = 1.$$

Noting that $|u_k| = \max\{-u_k, u_k\}$, we can transform to the primal problem:

$$\text{Maximize } x_{10}: = x_1 + x_2$$

$$\text{subject to } -1 \le x_i \le 1 \quad , \; i = 3, \ldots, 9, \text{ where}$$

$$x_3: = x_1 + 3x_2, \qquad x_4: = x_1 + 4x_2$$

$$x_5: = x_1 + 5x_2, \qquad x_6: = x_1 + 6x_2$$

$$x_7: = x_1 + 7x_2, \qquad x_8: = x_1 + 8x_2$$

$$x_9: = x_1 + 9x_2.$$

The variables x_1 and x_2, dual to artificial variables, are not constrained directly.

Input:

```
$IN
   IOBJ = 10,
   M = 8,
   N = 2,
   MIN = .FALSE.,
   INVERT = .TRUE.,
   ITMAX = 1000,
   EPS = 0.,
   ROW = 0, 10, 20, 30, 40, 50, 60, 70,
   COL = 1, 2,
   A(1, 1) = 1., 3.,
   A(1, 2) = 1., 4.,
   A(1, 3) = 1., 5.,
   A(1, 4) = 1., 6.,
```

```
   A(1, 5) = 1., 7.,
   A(1, 6) = 1., 8.,
   A(1, 7) = 1., 9.,
   A(1, 8) = 1., 1.,
   BL = 2*-1.E100, 7*-1., -1.E100,
   BU = 2*1.E100, 7*-1.,  1.E100.
   K = 1,2,3,4,5,6,7,8,9,10,
   X = 2*1.E100,
$END
```

Output:

I	BL(I)	X(I)	BU(I)	T(I, 3)	T(I, 9)
1	-R	2.000	R	1.500	-.500
2	-R	-.333	R	-.167	.167
3	-1.000	1.000	1.000	1.000	-.000
4	-1.000	.667	1.000	.833	.167
5	-1.000	.333	1.000	.667	.333
6	-1.000	.000	1.000	.500	.500
7	-1.000	-.333	1.000	.333	.667
8	-1.000	-.667	1.000	.167	.833
9	-1.000	-1.000	1.000	.000	1.000
10	-R	1.667	R	1.333	-.333

The solution is obtained from row 10: $u_3 = 1.333$, $u_9 = -.333$, $u_4, \ldots, u_8 = 0$. The minimal value is 1.667. (Obviously, the exact solution has $u_3 = 4/3$, $u_9 = -1/3$.)

```
      PROGRAM TEST 73/74     OPT =1        FTN 4.6 + 452                80/09/25. 13.42.54

1           PROGRAM TEST (INPUT , OUTPUT)                                  TEST       2
      C                                                                    LINE       2
      C--------------------------------------------------------------------LINE       3
      C                                                                    LINE       4
5     C     TEST PROGRAM FOR LINOPT                                        TEST       4
      C                                                                    LINE       2
      C--------------------------------------------------------------------LINE       3
      C                                                                    LINE       4
            DIMENSION A(10,10). ROW (10), COL (10), BL(20), BU(20), K(20)  TEST       6
10          DIMENSION X(20), U(20), E (100), SCR(10), KORD(10)             TEST       7
            INTEGER ROW, COL                                               TEST       8
      C                                                                    TEST       9
            COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR MOD1       1
            COMMON /XXXLP/ NPI, NPM, IPIV, JPIV, NEGV                      /XXXLP/    3
15          COMMON /XXXLP/ BIGM                                            MOD1       2
            LOGICAL MIN, INVERT, NEGV                                      MOD1       3
      C                                                                    LINE       2
      C--------------------------------------------------------------------LINE       3
      C                                                                    LINE

20          NAMELIST /IN/                                                  TEST      12
           1   IOBJ, M, N, MIN, INVERT, ITMAX, EPS,                        TEST      13
           2   A, ROW, COL, BL, BU, K, X                                   TEST      14
      C                                                                    LINE       2
      C--------------------------------------------------------------------LINE       3
25    C                                                                    LINE       4
            READ IN                                                        TEST      16
            CALL LINOPT(A,ROW,COL,BL,BU,K,X,U,E,SCR)                       TEST      17
            BIGM2 = SQRT(BIGM)                                             TEST      18
            DO 20 I = 1, NPM                                               TEST      19
30              IF (BL(I).LE.-BIGM2)  BL(I) = ANEGINF                      TEST      20
                IF(BU(I).GE.BIGM2  BU(I) = -ANEGINF                        TEST      21
                IF (ABS(X(I)).GE.BIGM2 X(I) = SIGN(ANEGINF,X(I))           TEST      22
20              CONTINUE                                                   TEST      23
            CALL TABLO (A,ROW,COL,BL,BU,K,X,E,SCR,KORD)                    TEST      24
35          PRINT*, 1H                                                     TEST      25
            PRINT*, 10H    ITER = , ITER, 10H  IERR = , IERR               TEST      26
            STOP                                                           TEST      27
            END                                                            TEST      28
```

FIGURE 2 TEST PROGRAM

CHAPTER 6

LISTING

```
      SUBROUTINE LINOPT(A,ROW,COL,BL,BU,K,X,U,E,SCR)                    LINOPT     2
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
C     LINEAR PROGRAMMING BY THE DUAL SIMPLEX ALGORITHM                  LINOPT     4
C                                                                       LINOPT     5
C                                                                       LINOPT     6
C     PROBLEM--                                                         LINOPT     7
C                                                                       LINOPT     8
C        MINIMIZE OR MAXIMIZE  X(IOBJ)  SUBJECT TO                      LINOPT     9
C                                                                       LINOPT    10
C           X(N+I)  =  SUM(J = 1,...,N)  A(ROW(I)+COL(J)) * X(J),       LINOPT    11
C                                                 I = 1,...,M,          LINOPT    12
C                                                                       LINOPT    13
C           BL(J)  .LE.  X(J)  .LE.  BU(J),  J = 1,...,N+M.             LINOPT    14
C                                                                       LINOPT    15
C                      ------------                                     LINOPT    16
C                                                                       LINOPT    17
C     FURTHER DOCUMENTATION AND EXAMPLES OF USE CAN BE FOUND IN--       LINOPT    18
C                                                                       LINOPT    19
C        NSWC TR 80-413,  LINOPT, A FORTRAN ROUTINE FOR SOLVING         LINOPT    20
C        LINEAR PROGRAMMING PROBLEMS, BY J.W. WINGATE.                  LINOPT    21
C                                                                       LINOPT    22
C                      ------------                                     LINOPT    23
C                                                                       LINOPT    24
C     ARRAYS ARE PASSED AS FORMAL PARAMETERS, SIMPLE VARIABLES AS       LINOPT    25
C     ELEMENTS OF THE COMMON BLOCK /XXXLP/.                             LINOPT    26
C                                                                       LINOPT    27
C     INPUTS--THE FOLLOWING VARIABLES AND ARRAYS MUST BE DEFINED ON     LINOPT    28
C             ENTRY                                                     LINOPT    29
C                                                                       LINOPT    30
C        IOBJ       INDEX OF THE OBJECTIVE VARIABLE.          (INTEGER) LINOPT    31
C                   (NOTE THAT  X(IOBJ)  IS ALSO CONSIDERED AS          LINOPT    32
C                   A CONSTRAINED VARIABLE.)                            LINOPT    33
C                                                                       LINOPT    34
C        M          NUMBER OF DEPENDENT VARIABLES.            (INTEGER) LINOPT    35
C        N          NUMBER OF INDEPENDENT VARIABLES.          (INTEGER) LINOPT    36
C                                                                       LINOPT    37
C        MIN        .TRUE. FOR MINIMIZATION,                  (LOGICAL) LINOPT    38
C                   .FALSE. FOR MAXIMIZATION.                           LINOPT    39
C                                                                       LINOPT    40
```

```
C       INVERT      .TRUE. IF THE INVERSE MATRIX E              (LOGICAL) LINOPT     41
C                   IS TO BE CALCULATED.                                  LINOPT     42
C                   .FALSE. IF E IS ALREADY SET TO THE INVERSE FOR        LINOPT     43
C                   THE BASIS DEFINED BY K.  (FOR REOPTIMIZATION,         LINOPT     44
C                   INVERT SHOULD BE .FALSE. UNLESS REINVERSION           LINOPT     45
C                   IS DESIRED.)                                          LINOPT     46
C                                                                         LINOPT     47
C       ITMAX       MAXIMUM NUMBER OF ITERATIONS ALLOWED.       (INTEGER) LINOPT     48
C                   CONTROL IS RETURNED TO THE CALLING                    LINOPT     49
C                   PROGRAM AFTER ITMAX ITERATIONS.                       LINOPT     50
C                                                                         LINOPT     51
C       EPS         ZERO TOLERANCE.  CONSTRAINT VIOLATIONS        (REAL) LINOPT     52
C                   OR TABLEAU ENTRIES .LE. EPS IN MAGNITUDE              LINOPT     53
C                   ARE TREATED AS ZERO.                                  LINOPT     54
C                                                                         LINOPT     55
C       A           ARRAY CONTAINING THE COEFFICIENT MATRIX.      (REAL) LINOPT     56
C       ROW         ROW INDEX ARRAY.                            (INTEGER) LINOPT     57
C       COL         COLUMN INDEX ARRAY.                         (INTEGER) LINOPT     58
C                   THE COEFFICIENT OF  X(J)  IN THE EQUATION             LINOPT     59
C                   FOR  X(N+I)  IS  A(ROW(I)+COL(J)).                    LINOPT     60
C                   EITHER (CASE 1)                                       LINOPT     61
C                      ONE HAS VECTORS  AROW1,...,AROWM  WITH             LINOPT     62
C                      AROWI(COL(J))  THE COEFFICIENT OF  X(J)            LINOPT     63
C                      IN THE EQUATION FOR  X(N+I), IN WHICH CASE         LINOPT     64
C                         ROW(I) = LOCF(AROWI) - LOCF(A),  I = 1,...,M,   LINOPT     65
C                   OR (CASE 2)                                           LINOPT     66
C                      ONE HAS VECTORS  ACOL1,...,ACOLN  WITH             LINOPT     67
C                      ACOLJ(ROW(I))  THE COEFFICIENT OF  X(J)            LINOPT     68
C                      IN THE EQUATION FOR  X(N+I), IN WHICH CASE         LINOPT     69
C                         COL(J) = LOCF(ACOLJ) - LOCF(A),  J = 1,...,N.   LINOPT     70
C                   (E.G. IF  A  IS DIMENSIONED FOR  MM  ROWS             LINOPT     71
C                   AND THE COEFFICIENT MATRIX IS STORED IN               LINOPT     72
C                   THE FIRST  M  ROWS AND  N  COLUMNS OF  A (CASE 2),    LINOPT     73
C                        ROW(I) = I,  I = 1,...,M,                        LINOPT     74
C                        COL(J) = (J-1)*MM,  J = 1,...,N,                 LINOPT     75
C                   WHILE IF THE COEFFICIENT MATRIX IS STORED              LINOPT     76
C                   TRANSPOSED IN THE FIRST  N  ROWS AND  M  COLUMNS      LINOPT     77
C                   (CASE 1),                                             LINOPT     78
C                        ROW(I) = (I-1)*MM,  I = 1,...,M,                 LINOPT     79
C                        COL(J) = J,  J = 1,...,N.                        LINOPT     80
C                   ROW AND COL MAY BE PERMUTED IN ANY CONVENIENT         LINOPT     81
C                   WAY.)                                                 LINOPT     82
C                                                                         LINOPT     83
C       BL          ARRAY OF LOWER BOUNDS.                        (REAL) LINOPT     84
C       BU          ARRAY OF UPPER BOUNDS.                        (REAL) LINOPT     85
C                                                                         LINOPT     86
C       K           BASIC SOLUTION KEY.                         (INTEGER) LINOPT     87
C                   K, IN CONJUNCTION WITH X, SPECIFIES A                 LINOPT     88
C                   PARTICULAR BASIC SOLUTION.  THE EQUATIONS             LINOPT     89
C                   RELATING  X(N+I), I = 1,...,M  TO                     LINOPT     90
C                   X(J), J = 1,...,N  (THE CONSTRAINT EQUATIONS)         LINOPT     91
C                   CAN BE SOLVED FOR VARIOUS COMBINATIONS OF             LINOPT     92
```

```
C                  M  VARIABLES (SECONDARY VARIABLES) IN TERMS          LINOPT     93
C                  OF THE REMAINING  N  VARIABLES (PRIMARY              LINOPT     94
C                  VARIABLES).  K, A PERMUTATION OF  (1,...,N+M),       LINOPT     95
C                  SPECIFIES SUCH A PARTITION INTO PRIMARY AND          LINOPT     96
C                  SECONDARY VARIABLES.  K(1),...,K(N)  ARE THE         LINOPT     97
C                  INDICES OF THE PRIMARY VARIABLES.  K(N+1),...,       LINOPT     98
C                  K(N+M)  ARE THE INDICES OF THE SECONDARY             LINOPT     99
C                  VARIABLES.  FOR THE DUAL VARIABLES U(J),             LINOPT    100
C                  J = 1,...,N+M, PRIMARY AND SECONDARY INDICES         LINOPT    101
C                  SWITCH ROLES,  U(K(N+1)),...,U(K(N+M)) BEING         LINOPT    102
C                  PRIMARY.  A BASIC SOLUTION IS SPECIFIED BY           LINOPT    103
C                  SETTING EACH PRIMAL PRIMARY VARIABLE TO              LINOPT    104
C                  EITHER OF ITS BOUNDS AND EACH DUAL PRIMARY           LINOPT    105
C                  VARIABLE TO ZERO.  THE INPUT VALUES OF THE           LINOPT    106
C                  PRIMAL PRIMARY VARIABLES ARE SWITCHED TO THE         LINOPT    107
C                  OPPOSITE BOUND IF NECESSARY IN ORDER TO CREATE       LINOPT    108
C                  A DUAL-FEASIBLE BASIC SOLUTION.                      LINOPT    109
C                                                                       LINOPT    110
C        X         PRIMAL SOLUTION ARRAY.                    (REAL)     LINOPT    111
C                  X(K(J))  MUST BE SET TO EITHER  BL(K(J)) OR          LINOPT    112
C                  BU(K(J)), J = 1,...,N. THESE ARE DEFAULT             LINOPT    113
C                  VALUES TO BE USED WHEN A VANISHING  U(K(J))          LINOPT    114
C                  MAKES  X(K(J))  INDETERMINATE IN SETTING UP          LINOPT    115
C                  A DUAL-FEASIBLE SOLUTION.                            LINOPT    116
C                                                                       LINOPT    117
C     OUTPUTS--THE FOLLOWING VARIABLES AND ARRAYS ARE DEFINED           LINOPT    118
C              OR REDEFINED ON EXIT                                     LINOPT    119
C                                                                       LINOPT    120
C        ITER      NUMBER OF ITERATIONS SINCE THE LAST       (INTEGER)  LINOPT    121
C                  INVERSION.                                           LINOPT    122
C                                                                       LINOPT    123
C        IERR      ERROR FLAG.                               (INTEGER)  LINOPT    124
C                  IERR = 0--OPTIMUM FOUND.                             LINOPT    125
C                         1--INCONSISTENT CONSTRAINTS.                  LINOPT    126
C                         2--ITERATION LIMIT REACHED.                   LINOPT    127
C                         3--INVERSION FAILED (BAD INITIAL BASIS).      LINOPT    128
C                                                                       LINOPT    129
C        K         BASIC SOLUTION KEY,                       (INTEGER)  LINOPT    130
C                  SET FOR THE CURRENT BASIS.                           LINOPT    131
C                                                                       LINOPT    132
C        X         PRIMAL SOLUTION ARRAY.                       (REAL)  LINOPT    133
C                                                                       LINOPT    134
C        U         DUAL SOLUTION ARRAY.                         (REAL)  LINOPT    135
C                  U(J)  IS THE DUAL VARIABLE (LAGRANGE                 LINOPT    136
C                  MULTIPLIER) FOR THE CONSTRAINTS ON  X(J).            LINOPT    137
C                  IT IS POSITIVE IF THE UPPER BOUND IS ACTIVE,         LINOPT    138
C                  NEGATIVE IF THE LOWER BOUND IS ACTIVE.               LINOPT    139
C                                                                       LINOPT    140
C        E         INVERSE MATRIX ARRAY.                        (REAL)  LINOPT    141
C                  X(I)  =  SUM (J = 1,...,N)  E(I,J) * X(K(J)),        LINOPT    142
C                                              I = 1,...,N.             LINOPT    143
```

```
C                                                                       LINOPT   144
C       SCR         (ALIAS AROW IN SUBROUTINES) SCRATCH ARRAY.          LINOPT   145
C                                                                       LINOPT   146
C                          ------------                                 LINOPT   147
C                                                                       LINOPT   148
C     MINIMUM DECLARED ARRAY SIZES--                                    LINOPT   149
C                                                                       LINOPT   150
C       A           M * N                                               LINOPT   151
C       ROW         M                                                   LINOPT   152
C       COL         N                                                   LINOPT   153
C       BL          M + N                                               LINOPT   154
C       BU          M + N                                               LINOPT   155
C       K           M + N                                               LINOPT   156
C       X           M + N                                               LINOPT   157
C       U           M + N                                               LINOPT   158
C       E           M * N                                               LINOPT   159
C       SCR         N                                                   LINOPT   160
C                                                                       LINOPT   161
C                          ------------                                 LINOPT   162
C                                                                       LINOPT   163
C     SUBROUTINE TABLO (Q.V.) PRINTS THE FULL EXPLICIT TABLEAU.         LINOPT   164
C     IT IS NOT CALLED THROUGH LINOPT AND MUST BE CALLED SEPARATELY.    LINOPT   165
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
      DIMENSION BL(1), BU(1), K(1), X(1), U(1), SCR(1)                  LINOPT   167
C                                                                       LINOPT   168
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR    /XXXLP/    2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                         /XXXLP/    3
      COMMON /XXXLP/ BIGM                                               /XXXLP/    4
      LOGICAL MIN, INVERT, NEGV                                         /XXXLP/    5
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
C     THE VARIABLE  BIGM  REPRESENTS A VERY LARGE NUMBER.  THE DEFAULT  LINOPT   171
C     VALUE IS 1.E100.  THE USER MAY RESET THIS VALUE IF SO DESIRED.    LINOPT   172
C     BIGM OR -BIGM MAY BE USED TO FILL IN MISSING UPPER OR LOWER       LINOPT   173
C     BOUNDS.                                                           LINOPT   174
C                                                                       LINOPT   175
      DATA BIGM /1.E100/                                                LINOPT   176
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
      NP1 = N + 1                                                       LINOPT   178
      NPM = N + M                                                       LINOPT   179
      IF (.NOT. INVERT)  GO TO 10                                       LINOPT   180
         CALL SETINV(A,ROW,COL,K,E,SCR)                                 LINOPT   181
         IF (IERR.EQ.3)  RETURN                                         LINOPT   182
   10 CONTINUE                                                          LINOPT   183
      CALL GETROW(A,ROW,COL,E,IOBJ,SCR)                                 LINOPT   184
      DO 50 J = 1, N                                                    LINOPT   185
         KJ = K(J)                                                      LINOPT   186
```

```
      IF (MIN)  SCR(J) = -SCR(J)                                    LINOPT  187
      U(KJ) = SCR(J)                                                LINOPT  188
      IF (U(KJ))  20, 40, 30                                        LINOPT  189
C     NEGATIVE                                                      LINOPT  190
   20    CONTINUE                                                   LINOPT  191
         X(KJ) = BL(KJ)                                             LINOPT  192
         GO TO 40                                                   LINOPT  193
C     POSITIVE                                                      LINOPT  194
   30    CONTINUE                                                   LINOPT  195
         X(KJ) = BU(KJ)                                             LINOPT  196
   40    CONTINUE                                                   LINOPT  197
   50    CONTINUE                                                   LINOPT  198
      DO 60 J = NP1, NPM                                            LINOPT  199
      U(K(J)) = 0.                                                  LINOPT  200
   60    CONTINUE                                                   LINOPT  201
      CALL PSOL(A,ROW,COL,K,X,E)                                    LINOPT  202
      CALL DSIMP(A,ROW,COL,BL,BU,K,X,U,E,SCR)                       LINOPT  203
      IF (IERR.NE.0)  GO TO 110                                     LINOPT  204
C     ROUND X-VALUES WITHIN EPS OF BOUNDS                           LINOPT  205
      DO 100 I = 1, NPM                                             LINOPT  206
         IF (ABS(X(I)-BL(I)).LE.EPS)  X(I) = BL(I)                  LINOPT  207
         IF (ABS(X(I)-BU(I)).LE.EPS)  X(I) = BU(I)                  LINOPT  208
  100    CONTINUE                                                   LINOPT  209
  110 CONTINUE                                                      LINOPT  210
      RETURN                                                        LINOPT  211
      END                                                           LINOPT  212
```

```
      SUBROUTINE DSIMP(A,ROW,COL,BL,BU,K,X,U,E,AROW)                    DSIMP     2
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
C     DUAL SIMPLEX ALGORITHM                                            DSIMP     4
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
      DIMENSION BL(1), BU(1), K(1), X(1), U(1), AROW(1)                 DSIMP     6
C                                                                       DSIMP     7
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR    /XXXLP/   2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                         /XXXLP/   3
      COMMON /XXXLP/ BIGM                                               /XXXLP/   4
      LOGICAL MIN, INVERT, NEGV                                         /XXXLP/   5
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
      IERR = 2                                                          DSIMP    10
C     WHEN ITMAX.LT.1 THE LOOP IS PARTIALLY EXECUTED                    DSIMP    11
      DO 100 II = 1, ITMAX                                              DSIMP    12
         CALL PIVROW(BL,BU,K,X)                                         DSIMP    13
         IF (IPIV.GT.0)  GO TO 10                                       DSIMP    14
C           NO PIVOT ROW INDICATES THAT X IS OPTIMAL                    DSIMP    15
            IERR = 0                                                    DSIMP    16
            RETURN                                                      DSIMP    17
   10    CONTINUE                                                       DSIMP    18
         KROW = K(IPIV)                                                 DSIMP    19
         CALL GETROW(A,ROW,COL,E,KROW,AROW)                             DSIMP    20
         CALL PIVCOL(BL,BU,K,X,U,AROW)                                  DSIMP    21
         IF (JPIV.GT.0)  GO TO 40                                       DSIMP    22
C           NO PIVOT COLUMN INDICATES THAT THE CONSTRAINTS              DSIMP    23
C           ARE INCONSISTENT                                            DSIMP    24
            IERR = 1                                                    DSIMP    25
            RETURN                                                      DSIMP    26
   40    CONTINUE                                                       DSIMP    27
         IF (ITMAX.LT.1)  RETURN                                        DSIMP    28
C        NEW SOLUTION KEY                                               DSIMP    29
         K(IPIV) = K(JPIV)                                              DSIMP    30
         K(JPIV) = KROW                                                 DSIMP    31
         CALL NEWINV(E,AROW)                                            DSIMP    32
C        NEW DUAL SOLUTION                                              DSIMP    33
         CALL GETROW(A,ROW,COL,E,IOBJ,AROW)                             DSIMP    34
         DO 70 J = 1, N                                                 DSIMP    35
            IF (MIN)  AROW(J) = -AROW(J)                                DSIMP    36
            U(K(J)) = AROW(J)                                           DSIMP    37
   70       CONTINUE                                                    DSIMP    38
         U(K(IPIV)) = 0.                                                DSIMP    39
C        NEW PRIMAL SOLUTION                                            DSIMP    40
         X(KROW) = BU(KROW)                                             DSIMP    41
         IF (NEGV)  X(KROW) = BL(KROW)                                  DSIMP    42
         CALL PSOL(A,ROW,COL,K,X,E)                                     DSIMP    43
         ITER = ITER + 1                                                DSIMP    44
  100    CONTINUE                                                       DSIMP    45
```

```
                                                        DSIMP    46
      RETURN                                            DSIMP    47
      END
```

```
      SUBROUTINE PIVROW(BL,BU,K,X)                                      PIVROW     2
C                                                                       LINE       3
C-----------------------------------------------------------------------LINE       4
C                                                                       LINE       5
C     PIVOT ROW SELECTION                                               PIVROW     4
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
      DIMENSION BL(1), BU(1), K(1), X(1)                                PIVROW     6
C                                                                       PIVROW     7
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR    /XXXLP/    2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                         /XXXLP/    3
      COMMON /XXXLP/ BIGM                                               /XXXLP/    4
      LOGICAL MIN, INVERT, NEGV                                         /XXXLP/    5
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
      IPIV = 0                                                          PIVROW    10
      IF (NPM.LT.NP1)  RETURN                                           PIVROW    11
      VIOL = 0.                                                         PIVROW    12
      DO 50 II = NP1, NPM                                               PIVROW    13
         I = K(II)                                                      PIVROW    14
C        CHECK CONSTRAINTS ON X(I)                                      PIVROW    15
            D = X(I) - BL(I)                                            PIVROW    16
            IF (D.GE.-EPS)  GO TO 10                                    PIVROW    17
               D = -D                                                   PIVROW    18
               IF (VIOL.GT.D)  GO TO 40                                 PIVROW    19
               VIOL = D                                                 PIVROW    20
               IPIV = II                                                PIVROW    21
               NEGV = .TRUE.                                            PIVROW    22
               GO TO 40                                                 PIVROW    23
   10       CONTINUE                                                    PIVROW    24
            D = X(I) - BU(I)                                            PIVROW    25
            IF (D.LE.EPS)  GO TO 30                                     PIVROW    26
               IF (VIOL.GT.D)  GO TO 40                                 PIVROW    27
               VIOL = D                                                 PIVROW    28
               IPIV = II                                                PIVROW    29
               NEGV =.FALSE.                                            PIVROW    30
   30       CONTINUE                                                    PIVROW    31
   40    CONTINUE                                                       PIVROW    32
   50    CONTINUE                                                       PIVROW    33
      RETURN                                                            PIVROW    34
      END                                                               PIVROW    35
```

```
      SUBROUTINE PIVCOL(BL,BU,K,X,U,AROW)                                PIVCOL     2
C                                                                        LINE       2
C------------------------------------------------------------------------ LINE       3
C                                                                        LINE       4
C     PIVOT COLUMN SELECTION                                             PIVCOL     4
C                                                                        LINE       2
C------------------------------------------------------------------------ LINE       3
C                                                                        LINE       4
      DIMENSION BL(1), BU(1), K(1), X(1), U(1), AROW(1)                  PIVCOL     6
C                                                                        PIVCOL     7
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR     /XXXLP/    2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                          /XXXLP/    3
      COMMON /XXXLP/ BIGM                                                /XXXLP/    4
      LOGICAL MIN, INVERT, NEGV                                          /XXXLP/    5
C                                                                        LINE       2
C------------------------------------------------------------------------ LINE       3
C                                                                        LINE       4
      JPIV = 0                                                           PIVCOL    10
      W = BIGM                                                           PIVCOL    11
      DO 30 JJ = 1, N                                                    PIVCOL    12
         J = K(JJ)                                                       PIVCOL    13
         AA = AROW(JJ)                                                   PIVCOL    14
         IF (NEGV)  AA = -AA                                             PIVCOL    15
         IF (AA.GE.0. .AND. X(J).EQ.BL(J))  GO TO 20                     PIVCOL    16
         IF (AA.LE.0. .AND. X(J).EQ.BU(J))  GO TO 20                     PIVCOL    17
            R = U(J)/AA                                                  PIVCOL    18
            IF (R.GT.W)  GO TO 10                                        PIVCOL    19
            IF (R.EQ.W .AND. RANF(AA).GT.0.5)  GO TO 10                  PIVCOL    20
               JPIV = JJ                                                 PIVCOL    21
               W = R                                                     PIVCOL    22
   10       CONTINUE                                                     PIVCOL    23
   20    CONTINUE                                                        PIVCOL    24
   30    CONTINUE                                                        PIVCOL    25
      RETURN                                                             PIVCOL    26
      END                                                                PIVCOL    27
```

```
      SUBROUTINE GETROW(A,ROW,COL,E,KROW,AROW)                          GETROW     1
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
C     GENERATION OF CONSTRAINT COEFFICIENTS FOR THE CURRENT BASIS       GETROW     4
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
      DIMENSION A(1), ROW(1), COL(1), E(1), AROW(1)                     GETROW     6
      INTEGER ROW, COL                                                  GETROW     7
C                                                                       GETROW     8
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR    /XXXLP/    2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                         /XXXLP/    3
      COMMON /XXXLP/ BIGM                                               /XXXLP/    4
      LOGICAL MIN, INVERT, NEGV                                         /XXXLP/    5
C                                                                       LINE       2
C-----------------------------------------------------------------------LINE       3
C                                                                       LINE       4
      IF (KROW.GT.N)  GO TO 20                                          GETROW    11
C        ORIGINAL INDEPENDENT VARIABLE.  GET ROW KROW OF THE INVERSE.   GETROW    12
         JJ = 0                                                         GETROW    13
         DO 10 J = 1, N                                                 GETROW    14
            AROW(J) = E(KROW+JJ)                                        GETROW    15
            IF (ABS(AROW(J)).LE.EPS)  AROW(J) = 0.                      GETROW    16
            JJ = JJ + N                                                 GETROW    17
   10       CONTINUE                                                    GETROW    18
         GO TO 50                                                       GETROW    19
   20 CONTINUE                                                          GETROW    20
C        ORIGINAL DEPENDENT VARIABLE.                                   GETROW    21
C        MULTIPLY ORIGINAL ROW BY THE INVERSE.                          GETROW    22
         KK = ROW(KROW-N)                                               GETROW    23
         JJ = 0                                                         GETROW    24
         DO 40 J = 1, N                                                 GETROW    25
            AROW(J) = 0.                                                GETROW    26
            DO 30 I = 1, N                                              GETROW    27
               AROW(J) = AROW(J) + A(KK+COL(I))*E(I+JJ)                 GETROW    28
   30          CONTINUE                                                 GETROW    29
            IF (ABS(AROW(J)).LE.EPS)  AROW(J) = 0.                      GETROW    30
            JJ = JJ + N                                                 GETROW    31
   40       CONTINUE                                                    GETROW    32
   50 CONTINUE                                                          GETROW    33
      RETURN                                                            GETROW    34
      END                                                               GETROW    35
```

```
      SUBROUTINE PSOL(A,ROW,COL,K,X,E)                                  PSOL      2
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
C     PRIMAL SOLUTION                                                   PSOL      4
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
      DIMENSION A(1), ROW(1), COL(1), K(1), X(1), E(1)                  PSOL      6
      INTEGER ROW, COL                                                  PSOL      7
C                                                                       PSOL      8
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR    /XXXLP/   2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                         /XXXLP/   3
      COMMON /XXXLP/ BIGM                                               /XXXLP/   4
      LOGICAL MIN, INVERT, NEGV                                         /XXXLP/   5
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
      DO 30 I = NP1, NPM                                                PSOL     11
         KI = K(I)                                                      PSOL     12
         IF (KI.GT.N)  GO TO 20                                         PSOL     13
            X(KI) = 0.                                                  PSOL     14
            JJ = 0                                                      PSOL     15
            DO 10 J = 1, N                                              PSOL     16
               X(KI) = X(KI) + E(KI+JJ) * X(K(J))                       PSOL     17
               JJ = JJ + N                                              PSOL     18
   10          CONTINUE                                                 PSOL     19
   20    CONTINUE                                                       PSOL     20
   30    CONTINUE                                                       PSOL     21
      DO 60 I = NP1, NPM                                                PSOL     22
         KI = K(I)                                                      PSOL     23
         IF (KI.LE.N)  GO TO 50                                         PSOL     24
            X(KI) = 0.                                                  PSOL     25
            KK = ROW(KI-N)                                              PSOL     26
            DO 40 J = 1, N                                              PSOL     27
               X(KI) = X(KI) + A(KK+COL(J)) * X(J)                      PSOL     28
   40          CONTINUE                                                 PSOL     29
   50    CONTINUE                                                       PSOL     30
   60    CONTINUE                                                       PSOL     31
      RETURN                                                            PSOL     32
      END                                                               PSOL     33
```

```
      SUBROUTINE SETINV(A,ROW,COL,K,E,AROW)                                SETINV     2
C                                                                          LINE       2
C------------------------------------------------------------------------- LINE       3
C                                                                          LINE       4
C     INITIAL INVERSE                                                      SETINV     4
C                                                                          LINE       2
C------------------------------------------------------------------------- LINE       3
C                                                                          LINE       4
      DIMENSION K(1), E(1), AROW(1)                                        SETINV     6
C                                                                          SETINV     7
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR       /XXXLP/    2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                            /XXXLP/    3
      COMMON /XXXLP/ BIGM                                                  /XXXLP/    4
      LOGICAL MIN, INVERT, NEGV                                            /XXXLP/    5
C                                                                          LINE       2
C------------------------------------------------------------------------- LINE       3
C                                                                          LINE       4
C     SET E TO THE IDENTITY                                                SETINV    10
      JJ = 0                                                               SETINV    11
      DO 20 J = 1, N                                                       SETINV    12
         DO 10 I = 1, N                                                    SETINV    13
            E(I+JJ) = 0.                                                   SETINV    14
   10       CONTINUE                                                       SETINV    15
         E(J+JJ) = 1.                                                      SETINV    16
         JJ = JJ + N                                                       SETINV    17
   20    CONTINUE                                                         SETINV    18
C     GENERATE INITIAL INVERSE                                             SETINV    19
      DO 30 J = 1, N                                                       SETINV    20
         K(J) = -K(J)                                                      SETINV    21
   30    CONTINUE                                                         SETINV    22
      DO 90 JJ = 1, N                                                      SETINV    23
         DO 40 J = 1, N                                                    SETINV    24
            IF (K(J).LT.0)  GO TO 50                                       SETINV    25
   40       CONTINUE                                                       SETINV    26
   50    CONTINUE                                                         SETINV    27
         KROW = -K(J)                                                      SETINV    28
         CALL GETROW(A,ROW,COL,E,KROW,AROW)                                SETINV    29
         ROWMAX = 0.                                                       SETINV    30
         DO 70 L = 1, N                                                    SETINV    31
            TEST = ABS(AROW(L))                                            SETINV    32
            IF (K(L).GT.0 .OR. TEST.LT.ROWMAX)  GO TO 60                   SETINV    33
               ROWMAX = TEST                                               SETINV    34
               JPIV = L                                                    SETINV    35
   60       CONTINUE                                                       SETINV    36
   70       CONTINUE                                                       SETINV    37
         IF (ROWMAX.GT.0.)  GO TO 80                                       SETINV    38
            IERR = 3                                                       SETINV    39
            RETURN                                                         SETINV    40
   80    CONTINUE                                                         SETINV    41
```

```
            K(J) = K(JPIV)                                          SETINV    42
            K(JPIV) = KROW                                          SETINV    43
            CALL NEWINV(E,AROW)                                     SETINV    44
   90       CONTINUE                                                SETINV    45
         ITER = 0                                                   SETINV    46
         RETURN                                                     SETINV    47
         END                                                        SETINV    48
```

```
      SUBROUTINE NEWINV(E,AROW)                                          NEWINV     2
C                                                                        LINE       2
C------------------------------------------------------------------------ LINE       3
C                                                                        LINE       4
C     INVERSE UPDATE BY COLUMN OPERATIONS                                NEWINV     4
C                                                                        LINE       2
C------------------------------------------------------------------------ LINE       3
C                                                                        LINE       4
      DIMENSION E(1), AROW(1)                                            NEWINV     6
C                                                                        NEWINV     7
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR     /XXXLP/    2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                          /XXXLP/    3
      COMMON /XXXLP/ BIGM                                                /XXXLP/    4
      LOGICAL MIN, INVERT, NEGV                                          /XXXLP/    5
C                                                                        LINE       2
C------------------------------------------------------------------------ LINE       3
C                                                                        LINE       4
      JJPIV = (JPIV-1)*N                                                 NEWINV    10
      DO 20 I = 1, N                                                     NEWINV    11
         EPIV = E(I+JJPIV)/AROW(JPIV)                                    NEWINV    12
         JJ = 0                                                          NEWINV    13
         DO 10 J = 1, N                                                  NEWINV    14
            E(I+JJ) = E(I+JJ) - EPIV*AROW(J)                             NEWINV    15
            JJ = JJ + N                                                  NEWINV    16
   10       CONTINUE                                                     NEWINV    17
         E(I+JJPIV) = EPIV                                               NEWINV    18
   20    CONTINUE                                                        NEWINV    19
      RETURN                                                             NEWINV    20
      END                                                                NEWINV    21
```

```
      SUBROUTINE TABLO(A,ROW,COL,BL,BU,K,X,E,SCR,KORD)                  TABLO     2
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
C     TABLEAU PRINTOUT                                                  TABLO     4
C                                                                       TABLO     5
C     KORD IS AN ARRAY OF LENGTH AT LEAST N USED FOR REORDERING         TABLO     6
C     K(1),...,K(N) IN ASCENDING ORDER.                                 TABLO     7
C     SEE LINOPT FOR DESCRIPTIONS OF THE OTHER PARAMETERS.              TABLO     8
C                                                                       TABLO     9
C     LINOPT MUST HAVE BEEN CALLED BEFORE CALLING TABLO.                TABLO    10
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
      DIMENSION A(1), ROW(1), COL(1), BL(1), BU(1), K(1), X(1), E(1)    TABLO    12
      DIMENSION SCR(1), KORD(1)                                         TABLO    13
      INTEGER ROW, COL                                                  TABLO    14
C                                                                       TABLO    15
      COMMON /XXXLP/ IOBJ, M, N, MIN, INVERT, ITMAX, EPS, ITER, IERR    /XXXLP/   2
      COMMON /XXXLP/ NP1, NPM, IPIV, JPIV, NEGV                         /XXXLP/   3
      COMMON /XXXLP/ BIGM                                               /XXXLP/   4
      LOGICAL MIN, INVERT, NEGV                                         /XXXLP/   5
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
    1 FORMAT (1H1//T55,*T A B L E A U*//1X,*  I *,5X,*BL(I)*,           TABLO    18
     /        6X,*X(I)*,5X,*BU(I)*,1X,10A10/(36X,10A10))               TABLO    19
    2 FORMAT (1H0,I3,1X,3F10.3,1X,10F10.3/(36X,10F10.3))                TABLO    20
    3 FORMAT (2X,*T(I,*,I3,*)*)                                         TABLO    21
    4 FORMAT (///1H0,*THE OBJECTIVE VARIABLE, X(*,I3,*), IS TO BE *,A10) TABLO   22
C                                                                       LINE      2
C-----------------------------------------------------------------------LINE      3
C                                                                       LINE      4
      DO 110 J = 1, N                                                   TABLO    24
         KORD(J) = J                                                    TABLO    25
  110    CONTINUE                                                       TABLO    26
      DO 130 J = 1, N                                                   TABLO    27
         JMIN = J                                                       TABLO    29
         DO 120 JJ = J, N                                               TABLO    29
            IF (K(KORD(JJ)).LT.K(KORD(JMIN)))  JMIN = JJ                TABLO    30
  120       CONTINUE                                                    TABLO    31
         KTEMP = KORD(J)                                                TABLO    32
         KORD(J) = KORD(JMIN)                                           TABLO    33
         KORD(JMIN) = KTEMP                                             TABLO    34
  130    CONTINUE                                                       TABLO    35
      DO 10 J = 1, N                                                    TABLO    36
         ENCODE (10,3,SCR(J)) K(KORD(J))                                TABLO    37
   10    CONTINUE                                                       TABLO    38
      PRINT 1, (SCR(J), J = 1, N)                                       TABLO    39
```

```
      DO 20 I = 1, NPH                                                  TABLO   40
         CALL GETROW(A,ROW,COL,E,I,SCR)                                 TABLO   41
         PRINT 2, I, BL(I), X(I), BU(I), (SCR(KORD(J)), J = 1, N)       TABLO   42
   20    CONTINUE                                                       TABLO   43
      OPT = 10HMAXIMIZED.                                               TABLO   44
      IF (MIN)  OPT = 10HMINIMIZED.                                     TABLO   45
      PRINT 4, IOBJ, OPT                                                TABLO   46
      RETURN                                                            TABLO   47
      END                                                               TABLO   48
```

DISTRIBUTION

	Copies
Commanding Officer Naval Air Systems Command Attn: Mr. Ralph A'Harrah (Code 5301) Mr. Dale E. Hutchins (Code 53011C) Mr. Douglas Kirkpatrick (Code 320D) Mr. Thomas S. Momiyama (Code ADP018) Mr. Richard S. Niemczyk (Code 5335) Mr. James Rebel (Code 5314) Dr. G. A. Heiche (AIR 310A) Washington, DC 20361	 1 1 1 1 1 1 1
Commanding Officer Naval Air Development Center Attn: Mr. Charles R. Abrams (Code 6014) Mr. S. T. Donley (Code 6014) Mr. Carmen J. Mazza (Code 6053) Mr. Edward J. Rickner (Code 6014) Warminister, PA 18974	 1 1 1 1
Office of Naval Research 800 N. Quincy Street Attn: Dr. Stuart L. Brodsky Arlington, VA 22217	 1
Commander Naval Air Test Center Attn: Mr. Anthony Rossetti (Code SA71) Patuxent River, MD 20670	 1
Commander Naval Weapons Center Attn: Dr. R. D. Smith (Code 3911) China Lake, CA 93555	 1
Professor Anthony Calire Drexel University Philadelphia, PA 19104	 1
Chief of Naval Operations Department of the Navy Attn: Mr. R. Piacesi Washington, DC 20350	 1

DISTRIBUTION (CONT.)

	Copies
Defense Technical Information Center Cameron Station Alexandria, VA 22314	12


DATE
FILMED
2-8