# Linear Programming: Theory and Applications

Catherine Lewis

May 11, 2008

# Contents

# 1 Introduction to Linear Programming

Linear programming was developed during World War II, when a system with which to maximize the efficiency of resources was of utmost importance. New war-related projects demanded attention and spread resources thin. "Programming" was a military term that referred to activities such as planning schedules efficiently or deploying men optimally. George Dantzig, a member of the U.S. Air Force, developed the Simplex method of optimization in 1947 in order to provide an efficient algorithm for solving programming problems that had linear structures. Since then, experts from a variety of fields, especially mathematics and economics, have developed the theory behind "linear programming" and explored its applications [1].

This paper will cover the main concepts in linear programming, including examples when appropriate. First, in Section 1 we will explore simple properties, basic definitions and theories of linear programs. In order to illustrate some applications of linear programming, we will explain simplified "real-world" examples in Section 2. Section 3 presents more definitions, concluding with the statement of the General Representation Theorem (GRT). In Section 4, we explore an outline of the proof of the GRT and in Section 5 we work through a few examples related to the GRT.

After learning the theory behind linear programs, we will focus methods of solving them. Section 6 introduces concepts necessary for introducing the Simplex algorithm, which we explain in Section 7. In Section 8, we explore the Simplex further and learn how to deal with no initial basis in the Simplex tableau. Next, Section 9 discusses cycling in Simplex tableaux and ways to counter this phenomenon. We present an overview of sensitivity analysis in Section 10. Finally, we put all of these concepts together in an extensive case study in Section 11.

## 1.1 What is a linear program?

We can reduce the structure that characterizes linear programming problems (perhaps after several manipulations) into the following form:

$$
\begin{array}{lccccccccc}
\text{Minimize} & c_1x_1 & + & c_2x_2 & + & \cdots & + & c_nx_n & = & z \\
\text{Subject to} & a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\
& a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\
& \vdots & & \vdots & & & & \vdots & \vdots & \vdots \\
& a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & b_m \\
& x_1, & & x_2, & & \ldots, & & x_n & \geq & 0.
\end{array}
$$

In linear programming $z$, the expression being optimized, is called the *objective function*. The variables $x_1, x_2 \ldots x_n$ are called *decision variables*, and their values are subject to $m + 1$ constraints (every line ending with a $b_i$, plus the nonnegativity constraint). A set of $x_1, x_2 \ldots x_n$ satisfying all the constraints is called a *feasible point* and the set of all such points is called the *feasible region*. The solution of the linear program must be a point $(x_1, x_2, \ldots, x_n)$ in the feasible region, or else not all the constraints would be satisfied.

The following example from Chapter 3 of Winston [3] illustrates that geometrically interpreting the feasible region is a useful tool for solving linear programming problems with two decision variables. The linear program is:

$$
\begin{array}{lccccc}
\text{Minimize} & 4x_1 & + & x_2 & = & z \\
\text{Subject to} & 3x_1 & + & x_2 & \geq & 10 \\
& x_1 & + & x_2 & \geq & 5 \\
& x_1 & & & \geq & 3 \\
& & x_1, & x_2 & \geq & 0.
\end{array}
$$

We plotted the system of inequalities as the shaded region in Figure 1. Since all of the constraints are "greater than or equal to" constraints, the shaded region above all three lines is the feasible region. The solution to this linear program must lie within the shaded region.

Recall that the solution is a point $(x_1, x_2)$ such that the value of $z$ is the smallest it can be, while still lying in the feasible region. Since $z = 4x_1 + x_2$, plotting the line $x_1 = (z - x_2)/4$ for various values of $z$ results in *isocost* lines, which have the same slope. Along these lines, the value of $z$ is constant. In Figure 1, the dotted lines represent isocost lines for different values of $z$. Since isocost lines are parallel to each other, the thick dotted isocost line for which $z = 14$ is clearly the line that intersects the feasible region at the smallest possible value for $z$. Therefore, $z = 14$ is the smallest possible value of $z$ given

the constraints. This value occurs at the intersection of the lines $x_1 = 3$ and $x_1 + x_2 = 5$, where $x_1 = 3$ and $x_2 = 2$.

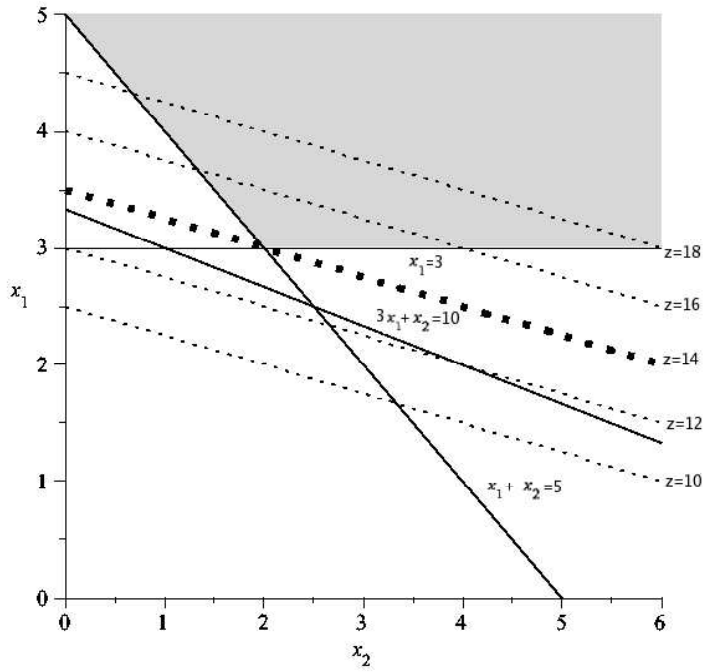


Figure 1: The shaded region above all three solid lines is the feasible region (one of the constraints does not contribute to defining the feasible region). The dotted lines are isocost lines. The thick isocost line that passes through the intersection of the two defining constraints represents the minimum possible value of $z = 14$ while still passing through the feasible region.

.

## 1.2 Assumptions

Before we get too focused on solving linear programs, it is important to review some theory. For instance, several assumptions are implicit in linear programing problems. These assumptions are:

1. **Proportionality** The contribution of any variable to the objective function or constraints is proportional to that variable. This implies no dis-

counts or economies to scale. For example, the value of $8x_1$ is twice the value of $4x_1$, no more or less.

2. **Additivity** The contribution of any variable to the objective function or constraints is independent of the values of the other variables.

3. **Divisibility** Decision variables can be fractions. However, by using a special technique called integer programming, we can bypass this condition. Unfortunately, integer programming is beyond the scope of this paper.

4. **Certainty** This assumption is also called the deterministic assumption. This means that all parameters (all coefficients in the objective function and the constraints) are known with certainty. Realistically, however, coefficients and parameters are often the result of guess-work and approximation. The effect of changing these numbers can be determined with sensitivity analysis, which will be explored later in Section 9 [3].

## 1.3 Manipulating a Linear Programming Problem

Many linear problems do not initially match the canonical form presented in the introduction, which will be important when we consider the Simplex algorithm. The constraints may be in the form of inequalities, variables may not have a nonnegativity constraint, or the problem may want to maximize $z$ instead of minimize $z$. We now consider some ways to manipulate problems into the desired form.

**Constraint Inequalities** We first consider the problem of making all constraints of a linear programming problem in the form of strict equalities. By introducing new variables to the problem that represent the difference between the left and the right-hand sides of the constraints, we eliminate this concern. Subtracting a slack variable from a "greater than or equal to" constraint or by adding an excess variable to a "less than or equal to" constraint, transforms inequalities into equalities. For example, the constraint $4x_1 + x_2 \leq 3$ becomes $4x_1 + x_2 + e_1 = 3$ with the addition of $e_1 \geq 0$. If the constraint were originally $4x_1 + x_2 \geq 3$, the additional surplus variable $s_1$ must be subtracted $(4x_1 + x_2 - s_1 = 3)$ so that $s_1$ can be a strictly nonnegative variable.

**URS Variables** If a variable can be negative in the context of a linear programming problem it is called a urs (or "unrestricted in sign") variable. By

replacing this variable $x_j$ with $x'_j - x''_j$ where $x'_j,\ x''_j \geq 0$, the problem fits the canonical form.

**Minimization/Maximization** If needed, converting a maximization problem to a minimization problem is quite simple. Given that $z$ is an objective function for a maximization problem

$$\max z = -\min\ (-z).$$

## 1.4 The Linear Algebra of Linear Programming

The example of a canonical linear programming problem from the introduction lends itself to a linear algebra-based interpretation. As a reminder, the form of a canonical problem is:

$$\begin{array}{lccccccccc}
\text{Minimize} & c_1x_1 & + & c_2x_2 & + & \cdots & + & c_nx_n & = & z \\
\text{Subject to} & a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\
& a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\
& \vdots & & \vdots & & & & \vdots & \vdots & \vdots \\
& a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & = & b_m \\
& x_1, & & x_2, & & \ldots, & & x_n & \geq & 0.
\end{array}$$

By applying some basic linear algebra, this problem becomes:

$$\begin{array}{lcccl}
\text{Minimize} & \sum_{j=1}^{n} c_j x_j & = & z & \\
\text{Subject to} & \sum_{j=1}^{n} \mathbf{a}_j x_j & = & \mathbf{b} & \\
& x_j & \geq & 0 & j = 1, 2, \ldots, n.
\end{array}$$

or, more compactly,

$$\begin{array}{lccc}
\text{Minimize} & \mathbf{cx} & = & z \\
\text{Subject to} & \mathbf{Ax} & = & \mathbf{b} \\
& \mathbf{x} & \geq & 0,
\end{array}$$

Here $\mathbf{A}$ is an $mxn$ matrix whose $j^{th}$ column is $\mathbf{a}_j$. This matrix corresponds to the coefficients on $x_1,\ x_2, \ldots, x_n$ in the constraints of a linear programming

problem. The vector $\mathbf{x}$ is a vector of solutions to the problem, $\mathbf{b}$ is the right-hand-side vector, and $\mathbf{c}$ is the cost coefficient vector. This more compact way of thinking about linear programming problems is useful especially in sensitivity analysis, which will be discussed in Section 9.

## 1.5 Convex Sets and Directions

This section defines important terms related to the feasible region of a linear program.

**Definition 1.1.** *A set $\boldsymbol{X} \in \mathbf{R}$ is a* **convex set** *if given any two points $\mathbf{x}_1$ and $\mathbf{x}_2$ in $\boldsymbol{X}$, any convex combination of these two points is also in $\boldsymbol{X}$. That is, $\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2 \in \mathbf{X}$ for any $\lambda \in [0,1]$ [2]. See Figure 4 for an example of a convex set.*

A set $\mathbf{X}$ is convex if the line segment connecting any two points in $\mathbf{X}$ is also contained in $\mathbf{X}$. If any part of the line segment not in $\mathbf{X}$, then $\mathbf{X}$ is said to be nonconvex. Figure 2 shows an example of a nonconvex set and a convex set.

The feasible region of a linear program is always a convex set. To see why this makes sense, consider the 2-dimensional region outlined on the axes in Figure 3. This region is nonconvex. In linear programming, this region could not occur because (from Figure 3) $y \leq mx + h$ for $c \leq x \leq d$ but $y \leq b$ for $d < x \leq e$. The constraint $y \leq mx + h$ doesn't hold when $d < x \leq e$ and the constraint $y \leq b$ doesn't hold when $0 \leq x \leq d$. These sorts of conditional constraints do not occur in linear programming. A constraint cannot be valid on one region and invalid on another.

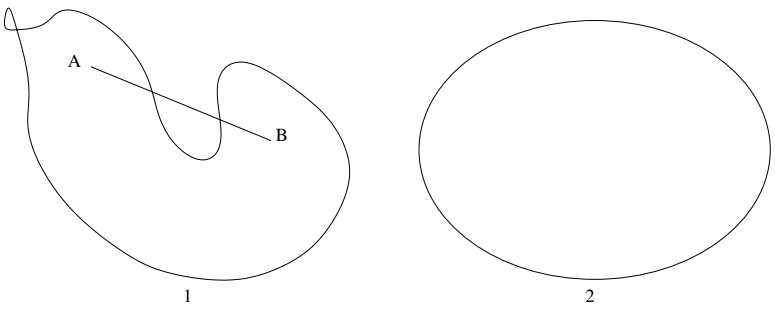


Figure 2: Set 1 is an example of a nonconvex set and set 2 is an example of a convex set. The endpoints of line segment $AB$ are in set 1, yet the entire line is not contained in set 1. No such line segment is possible in set 2.

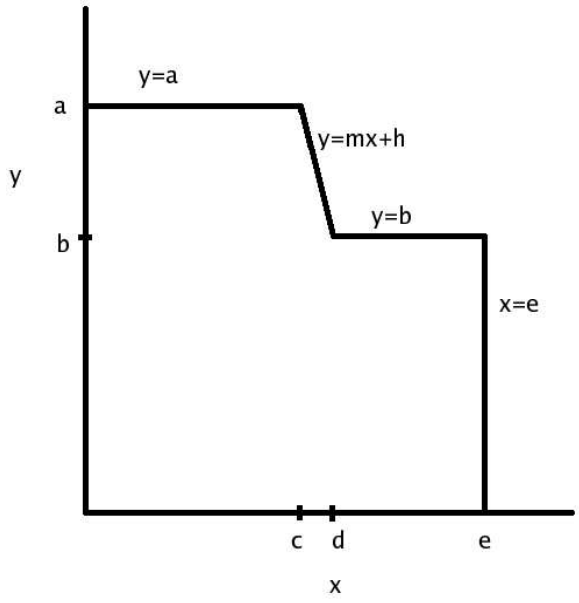


Figure 3: A nonconvex (and therefore invalid) feasible region.

**Definition 1.2.** *A point* $\mathbf{x}$ *in a convex set* $\mathbf{X}$ *is called an* **extreme point** *of* $\mathbf{X}$ *if* $\mathbf{x}$ *cannot be represented as a strict convex combination of two distinct points in* $\mathbf{X}$ *[2]. A strict convex combination is a convex combination for which* $\lambda \in (0, 1)$.

Graphically, an extreme point is a corner point. We will not prove this fact rigorously. In two dimensions, two constraints define an extreme point at their intersection. Further, if two points $\mathbf{x}'$ and $\mathbf{x}''$ make up a strict convex combination that equals $\bar{\mathbf{x}}$, then all three of these points must be defined by the same constraints, or else they could not be colinear. However, if two constraints define $\bar{\mathbf{x}}$ at their intersection, then $\bar{\mathbf{x}}$ is unique, since two lines can only intersect once (or not at all). Since this intersection is unique, yet all three points must lie on both constraints, $\mathbf{x}' = \mathbf{x}'' = \bar{\mathbf{x}}$. Therefore, the strict convex combination can't exist and hence corner points and extreme points are equivalent.

**Definition 1.3.** *Given a convex set, a nonzero vector* $\mathbf{d}$ *is a* **direction** *of the set if for each* $\mathbf{x}_0$ *in the set, the ray* $\{\mathbf{x}_0 + \lambda\mathbf{d} : \lambda \geq 0\}$ *also belongs to the set.*

**Definition 1.4.** *An* **extreme direction** *of a convex set is a direction of the set that cannot be represented as a positive combination of two distinct directions of the set.*

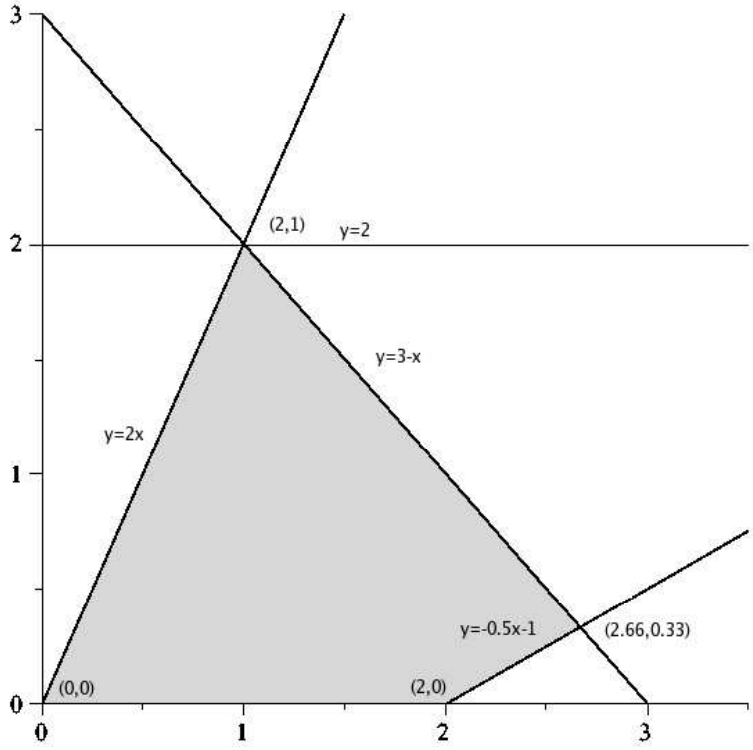


Figure 4: A bounded set with 4 extreme points. This set is bounded because there are no directions. Also, the extreme point $(2, 1)$ is a *degenerate extreme point* because it is the intersection of 3 constraints, but the feasible region is only two-dimensional. In general, an extreme point is degenerate if the number of intersecting constraints at that point is greater than the dimension of the feasible region.

Examples of a direction and an extreme direction can be seen in Figure 5.

We need all of these definitions to state the General Representation Theorem, a building-block in linear programming. The General Representation Theorem states that every point in a convex set can be represented as a convex combination of extreme points, plus a nonnegative linear combination of extreme directions. This theorem will be discussed more later.

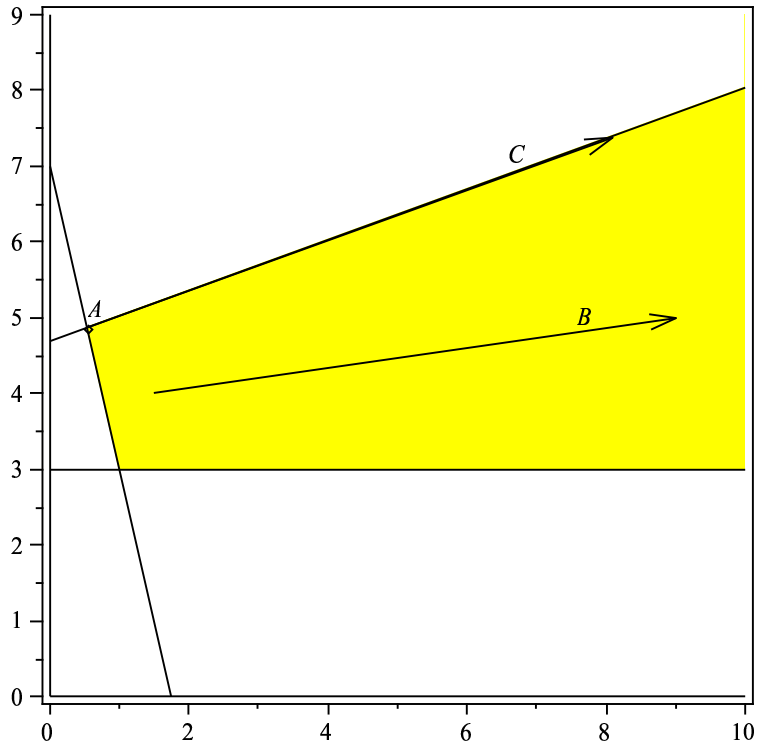


Figure 5: The shaded region in this figure is an unbounded convex set. Point A is an example of an extreme point, vector B is an example of a direction of the set and vector C is an example of an extreme direction of the set.

# 2 Examples from Bazaraa et. al. and Winston

The following examples deal with interpreting a word problem and setting up a linear program.

## 2.1 Examples

1. Consider the problem of locating a new machine to an existing layout consisting of four machines. These machines are located at the following $(x, y)$ coordinates: $(3, 0)$, $(0, -3)$, $(-2, 1)$, and $(1, 4)$. Let the coordinates of the new machine be $(x_1, x_2)$. Formulate the problem of finding an optimal location as a linear program if the sum of the distances from the new machine to the existing four machines is minimized. Use street distance; for example, the distance from $(x_1, x_2)$ to the first machine at

$(3, 0)$ is $|x_1 - 3| + |x_2|$. This means that the distance is not defined by the length of a line between two points, rather it is the sum of the lengths of the horizontal and vertical components of such a line.

**Solution** Since absolute value signs cannot be included in a linear program, recall that:

$$|x| = \max\{x, \ -x\}.$$

With this in mind, the following linear program models the problem:

$$\begin{array}{lrcl}
\text{Minimize} & z & = & (P_1 + P_2) + (P_3 + P_4) + (P_5 + P_6) + (P_7 + P_8) \\
\text{Subject to} & P_1 & \geq & -(x_1 - 3) \\
& P_1 & \geq & x_1 - 3 \\
& P_2 & \geq & -(x_2) \\
& P_2 & \geq & x_2 \\
& P_3 & \geq & -(x_1 - 1) \\
& P_3 & \geq & x_1 - 1 \\
& P_4 & \geq & -(x_2 - 4) \\
& P_4 & \geq & x_2 - 4 \\
& P_5 & \geq & -(x_1 + 2) \\
& P_5 & \geq & x_1 + 2 \\
& P_6 & \geq & -(x_2 - 1) \\
& P_6 & \geq & x_2 - 1 \\
& P_7 & \geq & -(x_1) \\
& P_7 & \geq & x_1 \\
& P_8 & \geq & -(x_2 + 3) \\
& P_8 & \geq & x_2 + 3 \\
& \text{all variables} & \geq & 0.
\end{array}$$

Each $P_{2i-1}$ represents the horizontal distance between the new machine and the $i^{th}$ old machine for $i = 1, 2, 3, 4$. Also for $i = 1, 2, 3, 4$, $P_{2i}$ represents the vertical distance between the new machine and the $i^{th}$ old machine. The objective function reflects the desire to minimize total distance between the new machine and all the others. The constraints relate the $P$ variables to the distances in terms of $x_1$ and $x_2$. Two constraints for each $P$ variable allow each $P_i$ $(i = 1, 2, \ldots, 8)$ to equal the maximum

of $x_j - c_j$ and $-(x_j - c_j)$ (for $j = 1, 2$ and where $c$ is the $j^{th}$ component of the position of one of the old machines). Since this program is a minimization problem and the smallest any of the variables can be is $\max\{(x_j - c_j),\ -(x_j - c_j)\}$, each $P_i$ will naturally equal its least possible value. This value will be the absolute value of $x_j - c_j$.

In the next problem we will also interpret a "real-world" situation as a linear program. Perhaps the most notable aspect of this problem is the concept of inventory and recursion in constraints.

2. A company is opening a new franchise and wants to try minimizing their quarterly cost using linear programming. Each of their workers gets paid \$500 per quarter and works 3 contiguous quarters per year. Additionally, each worker can only make 50 pairs of shoes per quarter. The demand (in pairs of shoes) is 600 for quarter 1, 300 for quarter 2, 800 for quarter 3, and 100 for quarter 4. Pairs of shoes may be put in inventory, but this costs \$50 per quarter per pair of shoes, and inventory must be empty at the end of quarter 4.

   **Solution** In order to minimize the cost per year, decision variables are defined. If we let

   $P_t$ – number of pairs of shoes during quarter $t$, $t = 1, 2, 3, 4$
   $W_t$ – number of workers starting work in quarter $t$, $t = 1, 2, 3, 4$
   $I_t$ – number of pairs of shoes in inventory after quarter $t$, $t = 1, 2, 3, 4$,

   the objective function is therefore

   $$\min z = 50I_1 + 50I_2 + 50I_3 + 1500W_1 + 1500W_2 + 1500W_3 + 1500W_4.$$

   Since each worker works three quarters they must be payed three times their quarterly rate. The objective function takes into account the salary paid to the workers, as well as inventory costs. Next, demand for shoes must be considered. The following constraints account for demand:

$$\begin{aligned} P_1 &\geq 600 \\ I_1 &= P_1 - 600 \\ I_2 &= I_1 + P_2 - 300 \\ I_3 &= I_2 + P_3 - 800 \\ I_4 &= I_3 + P_4 - 100 \quad = \quad 0. \end{aligned}$$

Note these constraints are recursive, meaning they can be defined by the expression:

$$I_n = I_{n_1} + P_n - D_n$$

where $D_n$ is just a number symbolizing demand for that quarter.

Also, the workers can only make 50 pairs of shoes per quarter, which gives us the constraints

$$\begin{aligned} P_1 &\leq 50W_1 + 50W_3 + 50W_4 \\ P_2 &\leq 50W_2 + 50W_4 + 50W_1 \\ P_3 &\leq 50W_3 + 50W_1 + 50W_2 \\ P_4 &\leq 50W_4 + 50W_3 + 50W_2 \end{aligned}$$

This linear program is somewhat cyclical, since workers starting work in quarter 4 can produce shoes during quarters 1 and 2. It is set up this way in order to promote long-term minimization of cost and to emphasize that the number of workers starting work during each quarter should always be the optimal value, and not vary year to year.

The next example is a similar type of problem. Again, decision variable must be identified and constraints formed to meet a certain objective. However, this problem deals with a different real-world situation and it is interesting to see the difference in the structure of the program.

In the previous problem, the concepts of inventory and scheduling were key, in the next problem, the most crucial aspect is conservation of matter. This means that several options will be provided concerning how to dispose of waste and all of the waste must be accounted for by the linear program.

3. Suppose that there are $m$ sources that generate waste and $n$ disposal sites. The amount of waste generated at source $i$ is $a_i$ and the capacity of site $j$ is $b_j$. It is desired to select appropriate transfer facilities from among $K$ candidate facilities. Potential transfer facility $k$ has fixed cost $f_k$, capacity

$q_k$ and unit processing cost $\alpha_k$ per ton of waste. Let $c_{ik}$ and $\bar{c}_{kj}$ be the unit shipping costs from source $i$ to transfer station $k$ and from transfer station $k$ to disposal site $j$ respectively. The problem is to choose the transfer facilities and the shipping pattern that minimize the total capital and operating costs of the transfer stations plus the transportation costs.

**Solution** As with the last problem, defining variables is the first step.

$$\begin{aligned} w_{ik} &= \text{tons of waste moved from } i \text{ to } k,\ 1 \le i \le m,\ 1 \le k \le K \\ y_k &= \text{a binary variable that equals 1 when transfer station } k \text{ is used} \\ &\quad \text{and 0 otherwise, } 1 \le k \le K \\ x_{kj} &= \text{tons of waste moved from } k \text{ to } j,\ 1 \le k \le K,\ 1 \le j \le n \end{aligned}$$

The objective function features several double sums in order to describe all the costs faced in the process of waste disposal. The objective function is:

$$\min\ z = \sum_i \sum_k c_{ik} w_{ik} + \sum_k \sum_j \bar{c}_{kj} x_{kj} + \sum_k f_k y_k + \sum_k \sum_i \alpha_k w_{ik}.$$

The first constraint equates the tons of waste coming from all the sources with the tons of waste going to all the disposal sites. This constraint is

$$\sum_i w_{ik} = \sum_j x_{kj}.$$

The next constraint says that the amount of waste produced equals the amount moved to all the transfer facilities:

$$\sum_k w_{ik} = \sum_i a_i.$$

Next, there must be a constraint that restricts how much waste is at transfer or disposal sites depending on their capacity. This restriction

gives:

$$\sum_k x_{kj} \leq b_j \quad \text{and} \quad \sum_i w_{ik} \leq q_k.$$

Putting these constraints all together, the linear program is:

$$\begin{array}{lllll} \text{Minimize} & z & = & \sum_i \sum_k c_{ik} w_{ik} + \sum_k \sum_j \bar{c}_{kj} x_{kj} + \sum_k f_k y_k + \sum_k \sum_i \alpha_k w_{ik} \\ \text{Subject to} & \sum_i w_{ik} & = & \sum_j x_{kj} \\ & \sum_k w_{ik} & = & \sum_i a_i \\ & \sum_k x_{kj} & \leq & b_j \\ & \sum_i w_{ik} & \leq & q_k \\ & \text{all variables} & \geq & 0. \end{array}$$

Most linear program-solving software allows the user to designate certain variables as binary, so ensuring this property of $y_k$ would not be an obstacle in solving this problem.

## 2.2 Discussion

Now let's discuss the affects altering a linear program has on the program's feasible region and optimal objective function value. Consider the typical linear program: minimize $\mathbf{cx}$ subject to $\mathbf{Ax} \geq \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$.

First, suppose that a new variable $x_{n+1}$ is added to the problem. The feasible region gets larger or stays the same. After the addition of $x_{n+1}$ the feasible region goes up in dimension. It still contains all of its original points (where $x_{n+1} = 0$) but it may now also include more.

The optimal objective function value of this program gets smaller or stays the same. Since there are more options in the feasible region, there may be one that more effectively minimizes $z$. If the new variable is added to $z$ (that is, any positive value of that variable would make $z$ larger), then that variable will equal 0 at optimality.

Now, suppose that one component of the vector $\mathbf{b}$, say $b_i$, is increased by one unit to $b_i + 1$. The feasible region gets smaller or stays the same. To see this more clearly, let $\mathbf{y}$ symbolize the row vector of coefficients of the constraint to which 1 was added. The new feasible region is everything from the old feasible region *except* points that satisfy the constraint $b_i \leq \mathbf{y} \cdot \mathbf{x} < b_i + 1$. This is either a positive number or zero, so therefore the feasible region gets smaller or stays

the same.

The optimal objective of this new program gets larger or stays the same. To satisfy the new constraint, the variables in that constraint may have to increase in value. This increase may in turn increase the value of the objective function.

In general, when changes are made in the program to increase the size of the feasible region, the optimal objective value either stays the same, or becomes more optimal (smaller for a minimization problem, or larger for a maximization problem). This is because a larger feasible region presents more choices, one of which may be better than the previous result. Conversely, when a feasible region decreases in size, the optimal objective value either stays the same, or becomes less optimal.

# 3 The Theory Behind Linear Programming

## 3.1 Definitions

Now that several examples, have been presented, it is time to explore the theory behind linear programming more thoroughly. The climax of this chapter will be the General Representation Theorem and to reach this end, more definitions and theorems are necessary.

**Definition 3.1.** *A **hyperplane** $H$ in $\mathbb{R}^n$ is a set of the form $\{\mathbf{x} : \mathbf{px} = k\}$ where $\mathbf{p}$ is a nonzero vector in $\mathbb{R}^n$ and $k$ is a scalar.*

For example, $\{(x_1, x_2) : (1, 0) \cdot (x_1, x_2) = 2\}$ is a hyperplane in $\mathbb{R}^2$. After completing the dot product, it turns out that this is just the line $x_1 = 2$ which can be plotted on the $x_1x_2$-plane.

A hyperplane in three dimensions is a traditional plane, and in two dimensions it is a line. The purpose of this definition is to generalize the idea of a plane to more dimensions.

**Definition 3.2.** *A **halfspace** is a collection of points of the form $\{\mathbf{x} : \mathbf{px} \geq k\}$ or $\{\mathbf{x} : \mathbf{px} \leq k\}$.*

Consider the two-dimensional halfspace $\{(x_1, x_2) : (1, 0) \cdot (x_1, x_2) \leq 2\}$. After completing the dot product, it is clear that this halfspace describes the region where $x_1 \leq 2$. On the $x_1, x_2$ plane, this would be everything to the left of the hyperplane $x_1 = 2$.

A halfspace in $\mathbb{R}^2$ is everything on one side of a line and, similarly, in $\mathbb{R}^3$ a halfspace is everything on one side of a a plane.

**Definition 3.3.** *A **polyhedral set** is the intersection of a finite number of halfspaces. It can be written in the form* $\{\mathbf{x} : \mathbf{Ax} \leq \mathbf{b}\}$ *where* $\mathbf{A}$ *is an* $m \times n$ *matrix (where* $m$ *and* $n$ *are integers).*

Every polyhedral set is a convex set. See Figure 6 for an example of a polyhedral set. A *proper face* of a polyhedral set $\mathbf{X}$ is a set of points that corresponds to some nonempty set of binding defining hyperplanes of $\mathbf{X}$. Therefore, the highest dimension of a proper face of $\mathbf{X}$ is equal to dim($\mathbf{X}$)-1. An *edge* of a polyhedral set is a one-dimensional face of $\mathbf{X}$. Extreme points are zero-dimensional faces of $\mathbf{X}$, which basically summarizes the next big definition:

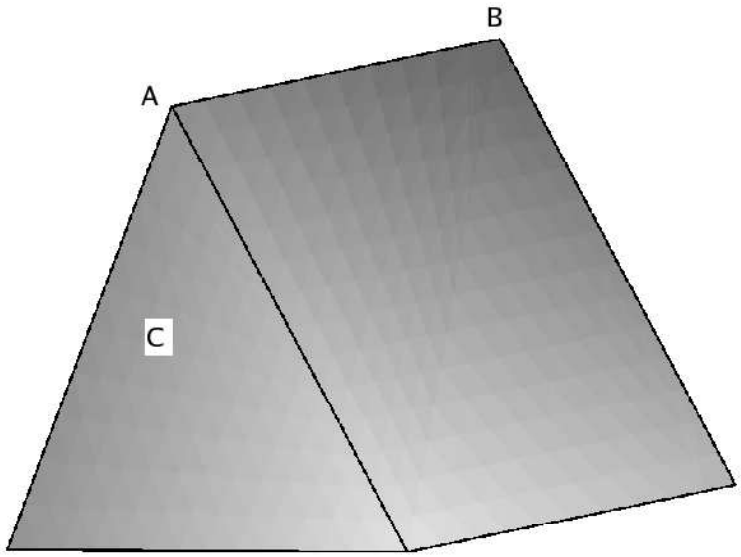


Figure 6: This "pup tent" is an example of a polyhedral set in $\mathbb{R}^3$. Point $A$ is an extreme point because it is defined by three halfspaces: everything behind the plane at the front of the tent, and everything below the two planes defining the sides of the tent. The line segment from $A$ to $B$ is called an "edge" of the polyhedral set. Set $C$ is a set of points that forms a two-dimensional proper face of the pup tent.

**Definition 3.4.** *Let* $X \in \mathbb{R}^n$*, where* $n$ *is an integer. A point* $\bar{\mathbf{x}} \in X$ *is said to be an **extreme point** of set* $X$ *if* $\bar{\mathbf{x}}$ *lies on some* $n$ *linearly independent defining*

*hyperplanes of $X$.*

Earlier, an extreme point was said to be a "corner point" of a feasible region in two dimensions. The previous definition states the definition of an extreme point more formally and generalizes it for more than two dimensions.

All of the previous definitions are for the purpose of generalizing the idea of a feasible region to more than two-dimensions. The feasible region will always be a polyhedral set, which, according to the definition, is a convex set in $n$ dimensions. The terminology presented in these definitions is used in the statement and proof of the General Representation Theorem, which is covered in the next section.

## 3.2 The General Representation Theorem

One of the most important (and difficult) theorems in linear programming is the General Representation Theorem. This theorem not only provides a way to represent any point in a polyhedral set, but its proof also lays the groundwork for understanding the Simplex method, a basic tool for solving linear programs.

**Theorem 3.1.** ***The General Representation Theorem:*** *Let $X = \{\mathbf{x} : \mathbf{Ax} \leq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ be a nonempty polyhedral set. Then the set of extreme points is not empty and is finite, say $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$. Furthermore, the set of extreme directions is empty if and only if $X$ is bounded. If $X$ is not bounded, then the set of extreme directions is nonempty and is finite, say $\{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_l\}$. Moreover, $\bar{\mathbf{x}} \in X$ if and only if it can be represented as a convex combination of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ plus a nonnegative linear combination of $\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_l$, that is,*

$$\bar{\mathbf{x}} = \sum_{j=1}^{k} \lambda_j \mathbf{x}_j + \sum_{j=1}^{l} \mu_j \mathbf{d}_j$$

$$\sum_{j=1}^{k} \lambda_j = 1,$$

$$\begin{aligned} \lambda_j &\geq 0, \qquad j = 1, 2, \ldots, k \\ \mu_j &\geq 0, \qquad j = 1, 2, \ldots, l. \end{aligned}$$

In sum, by visualizing an unbounded polyhedral set in two-dimensions, it is clear that any point in between extreme points can be represented as a convex combination of those extreme points. Any other point can be represented as

one of these convex combinations, plus a combination of multiples of extreme directions.

This theorem is called "general" since it applies to either a bounded polyhedral set (in the case that $\mu_j = 0$ for all $j$) or an unbounded polyhedral set.

# 4 An Outline of the Proof

The General Representation Theorem (GRT) has four main parts. First, it states that the number of extreme points in $\mathbf{X}$ is nonempty and finite. Next, if $\mathbf{X}$ is bounded then the set of extreme directions of $\mathbf{X}$ is empty. Further, if $\mathbf{X}$ is not bounded, then the set of extreme directions is nonempty and finite. Finally, the GRT states that $\bar{\mathbf{x}} \in \mathbf{X}$ if and only if it can be represented as a convex combination of extreme points of $\mathbf{X}$ and a nonnegative linear combination of extreme directions of $\mathbf{X}$. We will explore the proofs of these parts in sequence.

Since the last part of the GRT is a "if and only if" theorem, it must be proven both ways. Proving that if a point $x$ can be represented as a convex combination of extreme points of a set $\mathbf{X}$, plus a combination of multiples of extreme directions of $\mathbf{X}$, then $x \in \mathbf{X}$ (the "backward" proof), is simple. Before we do that, however, we must prove the first two parts of the GRT:

- First, we want to prove that there are a finite number of extreme points in the set $X$. We do this by moving along extreme directions until reaching a point that has $n$ linearly independent defining hyperplanes, where $n$ is just an integer. Thus there is at least 1 extreme point. Furthermore, $\binom{m+n}{n}$ is finite, so the number of extreme points is finite.

- Prove that there are a finite number of extreme directions in the set $X$. The argument here is similar to proving that there are a finite number of extreme points.

Next, the backwards proof:

- Given: $X = \{\mathbf{x} : \mathbf{A}\mathbf{x} \leq \mathbf{b}\, \mathbf{x} \geq \mathbf{0}\}$ is a nonempty polyhedral set. The set of extreme points of $X$ is not empty and has a finite number of points, say $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$. The set of extreme directions is either empty (if $X$ is bounded) or has a finite number of vectors, say $\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_l$. There is a point $\bar{\mathbf{x}}$ that can be written as in Equation GRT.

- Want to Prove: $\bar{\mathbf{x}} \in X$.

– First, remember that $\mathbf{A}\bar{\mathbf{x}} \leq \mathbf{b}$ if $\bar{\mathbf{x}} \in X$. From the assumptions:

$$\mathbf{A}\bar{\mathbf{x}} = \sum_{j=1}^{k} \lambda_j \mathbf{A}\mathbf{x}_j + \sum_{j=1}^{l} \mu_j \mathbf{A}\mathbf{d}_j.$$

It is clear that $\sum_{j=1}^{k} \mathbf{A}\mathbf{x}_j$ is less than $\mathbf{b}$ (since each $x_j$ is in $X$) and therefore $\sum_{j=1}^{k} \lambda_j \mathbf{A}\mathbf{x}_j \leq \mathbf{b}$ since $\sum_{j=1}^{k} \lambda_j = 1$.
But what about $\sum_{j=1}^{l} \mu_j \mathbf{A}\mathbf{d}_j$? From the definition of extreme directions, $\mathbf{d}$ is a direction of set $X$ if for each $\mathbf{x}$ in $X$, the ray $\{\mathbf{x} + \lambda\mathbf{d} : \lambda \geq 0\}$ also belongs to the set. From this definition, there are two restrictions on $\mathbf{d}$:

$$\begin{aligned} \mathbf{A}(\mathbf{x} + \lambda\mathbf{d}) &\leq \mathbf{b} \\ \mathbf{x} + \lambda\mathbf{d} &\geq \mathbf{0} \end{aligned} \tag{1}$$

for each $\lambda \geq 0$ and each $\mathbf{x} \in X$. The first inequality must hold for arbitrarily large $\lambda$, so therefore $\mathbf{A}\mathbf{d}$ must be less than or equal to $\mathbf{0}$. From the same reasoning, since $\mathbf{x} + \lambda\mathbf{d}$ can never be negative, no matter how big $\lambda$ gets, it follows that $\mathbf{d} \geq \mathbf{0}$ and $\mathbf{d} \neq \mathbf{0}$ (since $\mathbf{x}$ could have negative components). Therefore $\mathbf{d} > \mathbf{0}$.
From this information, it is clear that $\sum_{j=1}^{l} \mu_j \mathbf{A}\mathbf{d}_j$ is less than $\mathbf{0}$. Therefore, $\mathbf{A}\bar{\mathbf{x}}$ is the sum of two sums, one of which is less than or equal to $\mathbf{b}$ and one of which is less than $\mathbf{0}$. Therefore, $\mathbf{A}\bar{\mathbf{x}} \leq \mathbf{b}$.

– Next, $\bar{\mathbf{x}}$ must be nonnegative. Since $\bar{\mathbf{x}}$ is the sum of the products of all positive numbers, this must be true.

Therefore, if we can write $\bar{\mathbf{x}}$ in the form of the GRT then $\bar{\mathbf{x}} \in X$.

The "forward" proof of the GRT states that, if $x \in \mathbf{X}$, then a point $x$ can be represented as a convex combination of extreme points of a set $\mathbf{X}$, plus a combination of multiples of extreme directions of $\mathbf{X}$. Since the proof of the GRT in the forward direction is quite complicated, it will not be proven carefully. The following outline provides a summary of the steps involved with the proof of the forward direction:

- Define a bounded set that adds another constraint on to the original set $X$.
- Show that $\bar{x}$ can be written as a convex combination of extreme points in that set.

- Define the extreme directions of $X$ as the vectors created by subtracting the old extreme points (of $X$) from the newly created extreme points (of the new bounded set).
- Write a point in $X$ in terms of the extreme directions and evaluate the aforementioned convex combination in terms of this expression.
- The last step ends the proof by giving an expression that looks like the result of the GRT.

Even though this outline is not rigorous proof by any means, it still helps enhance understanding of the GRT. Figure 7 illustrates the GRT for a specific polyhedral set. It is from [2].

# 5 Examples With Convex Sets and Extreme Points From Bazaara et. al.

In the previous section, the General Representation Theorem concluded a theoretical overview of convex sets and their properties. Now it is time to apply this knowledge to a few examples. Let's see how a specific point in a specific convex set can be represented as a convex combination of extreme points. We will consider an example concerning the region $\mathbf{X}$=$\{(x_1, x_2) : x_1 - x_2 \leq 3, 2x_1 + x_2 \leq 4, x_1 \geq -3\}$ shown in Figure 8. We will find all the extreme points of $\mathbf{X}$ and express $\mathbf{x} = (0, 1)$ as a convex combination of the extreme points.

There are 3 extreme points, which are the pairwise intersections of the constraints when they are binding. Two of these points are on the line $x_1 = -3$ and these points are $(-3, -6)$ and $(-3, 10)$. In Figure 8 these points are $B$ and $C$ respectively. Point $A$, $\left(\frac{7}{3}, -\frac{2}{3}\right)$, is found by solving $x_1 - x_2 = 3$ and $2x_1 + x_2 = 4$ simultaneously.

In order to express $(1, 0)$ as a convex combination of extreme points, it is necessary to solve for $\lambda_1$ and $\lambda_2$ such that

$$\begin{aligned} -3\lambda_1 - 3\lambda_2 + \frac{7}{3}(1 - \lambda_1 - \lambda_2) &= 1 \\ -6\lambda_1 + 10\lambda_2 - \frac{2}{3}(1 - \lambda_1 - \lambda_2) &= 0. \end{aligned}$$

This is simply a system of two linear equations with two unknowns and the

solution $\lambda_1 = -\frac{11}{56}$ and $\lambda_2 = -\frac{29}{56}$. Therefore, the expression

$$\frac{2}{7}A - \frac{11}{56}B - \frac{29}{56}C = (1, 0)$$

is a convex combination of the extreme points that equals $(1, 0)$.

We can also use the GRT to show that the feasible region to a linear program in standard form is convex. Consider the linear program:

$$\begin{array}{lrcl} \text{Minimize} & \mathbf{cx} & = & z \\ \text{Subject to} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

and let $C$ be its feasible region. If $\mathbf{y}$ and $\mathbf{z} \in C$ and $\lambda$ is a real number $\in [0, 1]$, then we want to show that $\mathbf{w} = \lambda\mathbf{y} + (1 - \lambda)\mathbf{z}$ is greater than zero and that $\mathbf{Aw} = \mathbf{b}$.

It is clear that $\mathbf{w} \geq \mathbf{0}$ since both terms of $\mathbf{w}$ are products of positive numbers. To show the second condition, substitute in for $\mathbf{w}$:

$$\begin{aligned} \mathbf{Aw} &= \mathbf{A}(\lambda\mathbf{y} + (1 - \lambda)\mathbf{z}) \\ &= \lambda\mathbf{Ay} + \mathbf{Az} - \lambda\mathbf{Az} \\ &= \lambda\mathbf{b} + \mathbf{b} - \lambda\mathbf{b} \\ &= \mathbf{b}. \end{aligned}$$

This result means that $\mathbf{w} \in C$. Therefore, $C$ is a convex set.

# 6 Tools for Solving Linear Programs

## 6.1 Important Precursors to the Simplex Method

Linear programming was developed in order to obtain the solutions to linear programs. Almost always, finding a solution to a linear program is more important than the theory behind it. The most popular method of solving linear programs is called the Simplex algorithm. This section builds the groundwork for understanding the Simplex algorithm.

**Definition 6.1.** *Given the system* $\mathbf{Ax} = \mathbf{b}$ *and* $\mathbf{x} \geq 0$ *where* $\mathbf{A}$ *is an* $m \times n$ *matrix*

*and* $\mathbf{b}$ *is a column vector with* $m$ *entries. Suppose that rank(*$\mathbf{A}, \mathbf{b}$*)=rank(*$\mathbf{A}$*)=*$m$*. After possibly rearranging the columns of* $\mathbf{A}$*, let* $\mathbf{A} = [\mathbf{B}, \mathbf{N}]$ *where* $\mathbf{B}$ *is an* $m \times m$ *invertible matrix and* $\mathbf{N}$ *is an* $m \times (n-m)$ *matrix. The solution* $\mathbf{x} = \begin{bmatrix} \mathbf{x}_B \\ \mathbf{x}_N \end{bmatrix}$ *to the equations* $\mathbf{Ax} = \mathbf{b}$ *where* $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$ *and* $\mathbf{x}_N = 0$ *is called a* **basic solution** *of the system. If* $\mathbf{x}_B \geq \mathbf{0}$ *then* $\mathbf{x}$ *is called a* **basic feasible solution** *of the system. Here* $\mathbf{B}$ *is called the* **basic matrix** *and* $\mathbf{N}$ *is called the* **nonbasic matrix**. *The components of* $\mathbf{x}_B$ *are called* **basic variables** *and the components of* $\mathbf{x}_N$ *are called* **nonbasic variables**. *If* $\mathbf{x}_B > \mathbf{0}$ *then* $\mathbf{x}$ *is a* **nondegenerate basic solution**, *but if at least one component of* $\mathbf{x}_B$ *is zero then* $\mathbf{x}$ *is a* **degenerate basic solution** *[2].*

This definition might be the most important one for understanding the Simplex algorithm. The following theorems help tie together the linear algebra and geometry of linear programming. All of these theorems refer to the system

$$\begin{array}{lrcl} \text{Minimize} & \mathbf{cx} & & \\ \text{Subject to} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}. \end{array}$$

**Theorem 6.1.** *The collection of extreme points corresponds to the collection of basic feasible solutions, and both are nonempty, provided that the feasible region is not empty [2].*

**Theorem 6.2.** *If an optimal solution exists, then an optimal extreme point exists [2].*

Theorem 6.2 builds on the idea put forward in Theorem 6.1 except this time it addresses optimal points and solutions specifically. In Theorem 6.3, a "basis" refers to the set of basic variables.

**Theorem 6.3.** *For every extreme point there corresponds a basis (not necessarily unique), and, conversely, for every basis there corresponds a (unique) extreme point. Moreover, if an extreme point has more than one basis representing it, then it is degenerate. Conversely, a degenerate extreme point has more than one set of basic variables representing it if and only if the system* $\mathbf{Ax}$=$\mathbf{b}$ *itself does not imply that the degenerate basic variables corresponding to an associated basis are identically zero [2].*

All this theorem says is that a degenerate extreme point corresponds to several bases, but each basis represents only one point.

With all of this information, we might think that the best way to solve a linear programming problem is to find all the extreme points of a system and see which one correctly minimizes (or maximizes) the problem. This is realistic (though still tedious) for small problems. The number of extreme points is $\begin{pmatrix} n \\ m \end{pmatrix}$, where $m$ is the rank of $\mathbf{A}$ and $n$ is the number of variables. This number can be quite large. Also, this method does not indicate if the feasible region is unbounded or empty without first going through the whole set of extreme points. In short, this approach is not realistic. Instead, an ingenious algorithm known as the Simplex method, is the most common way to solve linear programs by hand, and is the basis for most computer software that solves linear programs.

In upcoming sections, the Simplex algorithm will be covered, including a discussion of how to manipulate programs with no starting basis. Then, sensitivity analysis will be discussed with examples. Finally, a case study concerning busing children to school will be presented and solved with computer software.

# 7 The Simplex Method In Practice

The Simplex algorithm remedies the shortcomings of the aforementioned "brute force" approach. Instead of checking *all* of the extreme points in the region, the Simplex algorithm selects an extreme point at which to start. Then, each iteration of the algorithm takes the system to the adjacent extreme point with the best objective function value. These iterations are repeated until there are no more adjacent extreme points with better objective function values. That is when the system is at optimality.

The best way to implement the Simplex algorithm by hand is through tableau form. A linear program can be put in tableau format by creating a matrix with a column for each variable, starting with $z$, the objective function value, in the far left column. For a general linear program, of the form

$$\begin{array}{lrcl} \text{Maximize} & \mathbf{cx} & & \\ \text{Subject to} & \mathbf{Ax} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

the initial tableau would look like:

| | | |
|---|---|---|
| $z$ | $-\mathbf{c}$ | 0 |
| $\mathbf{0}$ | $\mathbf{A}$ | $\mathbf{b}$ |

The idea here is that $z - \mathbf{cx} = 0$ and $\mathbf{Ax} = \mathbf{b}$ so in this format all the variables can be forgotten and represented only by columns. If we were actually completing the Simplex algorithm for this program, we would be concerned with the lack of an initial basis in the first tableau. If there is no initial basis in the tableau (an $mxm$ identity matrix), then we are not at an extreme point and the program cannot possibly be at optimality (according to Theorem 6.1).

The first iteration of the Simplex algorithm selects the column containing the most positive (in a minimization problem) coefficient in the first row (Row 0). Since Row 0 of the tableau corresponds to the objective function value $z$, the entering basic variable should be one that would decrease the value of $z$ the fastest. This variable corresponds to the column with the most positive objective function coefficient, say, column $i$.

After selecting this variable as the entering basic variable, perform the ratio test. This test consists of dividing all the numbers on the right-hand side of the tableau by their corresponding coefficients in column $i$. This test is not performed in Row 0. If a number in column $i$ is nonpositive, the ratio test doesn't apply for that row. The row $j$ with the smallest positive ratio is selected. If the ratio test is a tie, the row can be selected arbitrarily between rows with the same valued (smallest) ratios. The ratio test is necessary because it ensures that the right-hand side of the tableau (except perhaps Row 0) will stay positive as the new basic variable enters the basis.

The new basic variable selected, say $x_i$, enters the basis in row $j$. Therefore, through elementary matrix row operations, column $i$ must be transformed into a column of the identity matrix, with the one in the $j^{th}$ row. Through these operations, $x_i$ enters the basis and whichever basic variable was in row $j$ leaves the basis.

We repeat these steps until there are no longer positive numbers in Row 0 (for a minimization problem).

To help clarify these ideas, let's look at an example from [3]:

First, let's use the simplex algorithm to find the optimal solution of the

Table 1: The number with the * next to it is the entering basic variable, in this case $x_1$. The variable $s_1$ is the leaving basic variable, since the one in the $s_1$ column is in Row 1.

| Row | z | $x_1$ | $x_2$ | $s_1$ | $s_2$ | RHS | BV | ratio |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 3 | -8 | 0 | 0 | 0 | $z = 0$ | - |
| 1 | 0 | $4^*$ | 2 | 1 | 0 | 12 | $s_1 = 12$ | $\frac{12}{4} = 3$ |
| 2 | 0 | 2 | 3 | 0 | 1 | 6 | $s_2 = 6$ | $\frac{6}{2} = 3$ |

following LP:

$$\begin{aligned} \min \ z &= -3x_1 + 8x_2 \\ \text{s.t.} \ 4x_1 + 2x_2 &\leq 12 \\ 2x_1 + 3x_2 &\leq 6 \\ x_1, \ x_2 &\geq 0 \end{aligned}$$

First this LP must be in standard form. We add slack variables to transform our linear program to

$$\begin{aligned} \min \ z &= -3x_1 + 8x_2 \\ \text{s.t.} \ 4x_1 + 2x_2 + s_1 &= 12 \\ 2x_1 + 3x_2 + s_2 &= 6 \\ x_1, \ x_2, \ s_1, \ s_2 &\geq 0 \end{aligned}$$

The first tableau is shown in Table 1. The rows are labeled 0, 1, and 2. The columns are labeled according to which variable they represent. The column labeled "RHS" indicates the right-hand side of the equations. The column "BV" indicates the values of basic variables in that tableau. In the "ratio" column, we give the ratios used in the ratio test.

In Table 1 $s_1$ and $s_2$ are the initial basic variables, since their columns are columns from the identity matrix. The number in Row 0 that is most positive is the 3 in the $x_1$ column. Therefore, $x_1$ is the entering basic variable. The ratio test (shown in the far right column of Table 1), yields a tie since both ratios are 3. Row 1 is chosen arbitrarily to be the row of the entering basic variable.

Table 2: This tableaux is optimal since all the values in Row 0 (except for the $z$ column) are nonpositive.

| Row | z | $x_1$ | $x_2$ | $s_1$ | $s_2$ | RHS | BV | ratio |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | $-\frac{19}{2}$ | $-\frac{3}{4}$ | 0 | -9 | $z = -9$ | - |
| 1 | 0 | 1 | $\frac{1}{2}$ | $\frac{1}{4}$ | 0 | 3 | $x_1 = 3$ | - |
| 2 | 0 | 0 | 2 | $-\frac{1}{2}$ | 1 | 0 | $s_2 = 0$ | - |

The next tableau is shown in Table 2. This table is obtained after dividing Row 1 of Table 1 by 4, and then adding $-3$ times Row 1 to Row 0 and $-2$ times Row 1 to Row 2.

The solution shown in Table 2 is optimal because all the numbers (except for the number in the $z$ column, of course) in Row 0 are negative, which means $z$ cannot be decreased anymore. The solution is $x_1 = 3,\ x_2 = s_1 = s_2 = 0$ and $z = -9$.

Now that the basics of the Simplex algorithm have been covered, the next section will explain what to do when the initial tableau does not contain an identity matrix.

# 8 What if there is no initial basis in the Simplex tableau?

After learning the basics of the Simplex algorithm, we explore the special cases and exceptions. For example, what does one do when the initial tableau contains no starting basis? That is, what happens if the initial tableau (for a program where $\mathbf{A}$ is $m$x$n$) does not contain all the columns of an $m$x$m$ identity matrix?

This problem occurs when the constraints are in either equality or "greater than or equal to" form. In the case of an equality constraint, one cannot add a slack or an excess variable and therefore there will be no initial basic variable in that row. In the case of a "greater than or equal to" constraint, one must add an excess variable, which gives a $-1$ in that row instead of a $+1$. Simply multiplying through that row by $-1$ would not solve the problem, for then the right-hand-side variable would be negative.

There are two main methods to dealing with this problem: the Two-Phase Method and the Big-M Method. Both of these methods involve adding "artificial" variables that start out as basic variables but must eventually equal zero in

Table 3:

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | -2 | 1 | -1 | 0 | 0 | 0 |
| 0 | 2 | 1 | -2 | 0 | 1 | 8 |
| 0 | 4 | -1 | 2 | 0 | 0 | 2 |
| 0 | 2 | 3 | -1 | -1 | 0 | 4 |

Table 4:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | -1 | -1 | 0 |
| 0 | 2 | 1 | -2 | 0 | 1 | 0 | 0 | 8 |
| 0 | 4 | -1 | 2 | 0 | 0 | 1 | 0 | 2 |
| 0 | 2 | 3 | -1 | -1 | 0 | 0 | 1 | 4 |

order for the original problem to be feasible. These two techniques are outlined in the following sections.

## 8.1 The Two-Phase Method

Consider the following problem:

$$\begin{array}{lrcl} \text{Maximize} & 2x_1 - x_2 + x_3 & & \\ \text{Subject to} & 2x_1 + x_2 - 2x_3 & \leq & 8 \\ & 4x_1 - x_2 + 2x_3 & = & 2 \\ & 2x_1 + 3x_2 - x_3 & \geq & 4 \\ & x_1, x_2, x_3 & \geq & 0. \end{array}$$

To put this in standard form, all constraints must become equalities. Adding a slack variable, $s_1$, to the first constraint and subtracting an excess variable, $e_3$, from the third constraint gives the initial Simplex tableau shown in Table 3.

There is no initial basis in this tableau. To correct this, add two more variables, called "artificial variables," $a_1$ and $a_3$, to constraints two and three. This step alone however, will not solve the problem since these two extra variables cannot be in the final basis. Therefore, in addition to adding two artificial variables, the initial objective function must be altered as well. To ensure that $a_1$ and $a_3$ are not in the final basis, the initial objective function must become "minimize $z = a_1 - a_3$." Therefore, the tableau shown in Table 3 becomes the tableau shown in Table 4.

The last step before performing the Simplex algorithm as usual, is to add rows 2 and 3 to Row 0. This gives the tableau shown in Table 5. The tableau

Table 5:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 6 | 2 | 1 | -1 | 0 | 0 | 0 | 6 |
| 0 | 2 | 1 | -2 | 0 | 1 | 0 | 0 | 8 |
| 0 | 4 | -1 | 2 | 0 | 0 | 1 | 0 | 2 |
| 0 | 2 | 3 | -1 | -1 | 0 | 0 | 1 | 4 |

Table 6:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | .6 | -.3 | 0 | -.8 | -1.4 | .8 |
| 0 | 0 | 0 | -2.6 | .2 | 1 | -.25 | .2 | 6.4 |
| 0 | 1 | 0 | .3 | -.1 | 0 | .2 | .10 | .8 |
| 0 | 0 | 1 | -.8 | -.4 | 0 | -.2 | .4 | 1.2 |

in Table 5 is better because $I_3$ is in the body of the tableau.

The tableau shown in Table 5 is the initial tableau for Phase I of the Two-Phase Method. Phase I will be complete when this tableau is optimal. To get this tableau to optimality takes two steps. The final optimal tableau for Phase I is shown in Table 6. In this tableau the variables $a_1$ and $a_3$ are zero and the variables $x_1$, $x_2$ and $s_1$ are basic.

For Phase II of the Two-Phase Method, replace the current objective function with the objective function from the original problem and make all entries of Row 0 in the column of a basic variable equal to zero. Also, delete all columns associated with artificial variables. After that, proceed with the Simplex algorithm until optimality is reached. The first tableau for Phase II is shown in the first part of Table 7. After multiplying Row 2 by two, Row 3 by negative one, and adding both of the resultant rows to Row 0, the tableau becomes like the second part of Table 7. After obtaining this tableau, proceed with the Simplex algorithm as usual. Remember, now the objective function must be *maximized*. The solution to this phase is the solution to the original problem.

In summary, the Two-Phase Method can be completed using the following steps:

1. If there is no basis in the original tableau, add artificial variables to rows in which there are no slack variables.

2. Set up a new objective function that minimizes these slack variables.

3. Modify the tableau so that the Row 0 coefficients of the artificial variables are zero.

Table 7:

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | -2 | 1 | -1 | 0 | 0 | 0 |
| 0 | 0 | 0 | -2.6 | .2 | 1 | 6.4 |
| 0 | 1 | 0 | .3 | -.1 | 0 | .8 |
| 0 | 0 | 1 | -.8 | -.4 | 0 | 1.2 |
| 1 | 0 | 0 | .4 | .2 | 0 | .4 |
| 0 | 0 | 0 | -2.6 | .2 | 1 | 6.4 |
| 0 | 1 | 0 | .3 | -.1 | 0 | .8 |
| 0 | 0 | 1 | -.8 | -.4 | 0 | 1.2 |

4. Complete the Simplex algorithm as usual for a minimization problem.

5. At optimality, the artificial variables should be nonbasic variables, or basic variables equal to zero. If not, the original problem was infeasible.

6. After reaching optimality, delete all rows associated with artificial variables. Also, replace the artificial objective function with the original objective function. This is the start of Phase II.

7. Modify the tableau so that the Row 0 coefficients of the basic variables are zero.

8. Proceed with the Simplex algorithm as usual. The optimal solution to this tableau will be the optimal solution to the original problem.

## 8.2 The Big-M Method

Another way to deal with no basis in the initial tableau is called the Big-M Method. The following example from [2] will help illustrate this method:

Consider the problem

$$\begin{array}{lrcl} \text{Minimize} & x_1 - 2x_2 & & \\ \text{Subject to} & x_1 + x_2 & \geq & 2 \\ & -x_1 + x_2 & \geq & 1 \\ & x_2 & \leq & 3 \\ & x_1, x_2 & \geq & 0. \end{array}$$

Wouldn't it be nice to be able to solve this problem with artificial variables, but without having to complete two phases? Well, the Big-M method makes that possible.

Table 8:

| 1 | -1 | 2 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | -1 | 0 | 0 | 2 |
| 0 | -1 | 1 | 0 | -1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 3 |

Table 9:

| 1 | -1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | -1 | 0 | 0 | 1 | 0 | 2 |
| 0 | -1 | 1 | 0 | -1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 3 |
| 1 | -1 | 2 | 0 | 0 | 0 | -M | -M | 0 |
| 0 | 1 | 1 | -1 | 0 | 0 | 1 | 0 | 2 |
| 0 | -1 | 1 | 0 | -1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 3 |
| 1 | -1 | 2+2M | -M | -M | 0 | 0 | 0 | 3M |
| 0 | 1 | 1 | -1 | 0 | 0 | 1 | 0 | 2 |
| 0 | -1 | 1 | 0 | -1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 3 |
| 1 | 1+2M | 0 | -M | 2+M | 0 | 0 | -2-2M | -2+M |
| 0 | 2 | 0 | -1 | 1 | 0 | 1 | -1 | 1 |
| 0 | -1 | 1 | 0 | -1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | -1 | 2 |
| 1 | -1 | 0 | 0 | 0 | -2 | -M | -M | -6 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | -1 | 2 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 3 |
| 0 | -1 | 0 | 1 | 0 | 1 | -1 | 0 | 1 |

First, the initial tableau of this problem is shown in Table 6. After adding artificial variables $a_1$ and $a_2$ to rows 1 and 2, the tableau becomes the tableau in the first part of Table 8. Since the artificial variables must go to zero at optimality, there must be a way to force them to do so. By adding $Ma_1 + Ma_2$ to the objective function, where $M$ is a really *big* number (near infinity!), it is clear that this minimization problem will be optimal when $a_1$ and $a_2$ are zero. For a maximization problem, adding $-Ma_1 - Ma_2$ to the objective function produces the same effect. This step is shown in the second part of Table 9.

Next, the coefficients in Row 0 must be modified so that they are zero for all basic variables. Therefore, multiply rows 1 and 2 by $M$ and add them to Row 0. The resulting tableau is shown in the third part of Table 9.

The most positive coefficient in Row 0 is $2 + 2M$. Therefore, this column is selected for a pivot and the ratio test is performed as usual. Row 2 wins the ratio test and the next tableau is shown in the fourth part of Table 9. The term $1 + 2M$ is the most positive so that column is the pivot column. This part of the Big-M Method is just like performing the Simplex Method as usual, except with a variable $M$ floating around. This particular problem continues for a few more steps (two more pivots to be exact) before reaching optimality. The optimal tableau is shown in the fifth part of Table 9, where the solution is $x_2 = 3,\ x_3 = 1,\ x_4 = 2$ and $z = -6$.

In this tableau, there is no $M$ in the objective function value. Also, none of the artificial variables are basic variables. If an artificial variable were a basic variable in the optimal tableau, this would indicate that the the original problem was infeasible. If there were an $M$ in the objective function value, this would indicate that the original problem was unbounded.

In summary, the Big-M Method can be completed using the following steps:

1. If there is no basis in the original tableau, add artificial variables to rows in which there are no slack variables.

2. Depending on whether the problem is minimization or maximization, add or subtract $M\mathbf{x}_a$ to the objective function where $M$ is a *really* big number and $\mathbf{x}_a$ is a vector of the artificial variables.

3. Modify the tableau so that the Row 0 coefficients of the artificial variables are zero.

4. Complete the Simplex algorithm as usual.

5. At optimality, the artificial variables should be nonbasic variables and the objective function value should be finite.

# 9 Cycling

In the case of a degenerate problem, a rare phenomenon occurs when a Simplex tableau iterates through several bases all representing the same solution, only to arrive back at the original tableau. Geometrically, several linearly independent hyperplanes define a certain extreme point, and the Simplex algorithm moves with each pivot to another set of these hyperplanes that represent the same basis, and finally comes back to the beginning set of these hyperplanes and

starts over again. Thus, the tableau "cycles" around an extreme point. This phenomenon is aptly called "cycling."

As mentioned before, this phenomenon is quite rare. The following example in Table 10 is from [2]. At each iteration, the pivot is indicated with a $*$.

The two important things to note in this series of tableaux are that the solution does not change even though the basis does change and that the last tableau is the same as the first.

Cycling is undesirable because it prevents the Simplex algorithm from reaching optimality. Therefore, even though cycling is so rare, there are several rules that prevent its occurrence.

## 9.1 The Lexicographic Method

Given the problem

$$\begin{array}{lrcl} \text{Minimize} & \mathbf{cx} & & \\ \text{Subject to} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0}, \end{array}$$

where $\mathbf{A}$ is an $m \times n$ matrix of rank $m$, suppose that there is a starting basis such that the first $m$ columns of $\mathbf{A}$ form the identity. The following steps, called the *lexicographic rule* are guaranteed to prevent cycling.

1. Say that the nonbasic variable $x_k$ is chosen to enter the basis by normal means (its coefficient is the most positive number in Row 0).

2. The index $r$ of the leaving basic variable $x_r$ is in the set

$$I_0 = \left\{ r : \frac{\bar{b}_r}{y_{rk}} = \overset{\text{Minimum}}{1 \leq i \leq m} \left\{ \frac{\bar{\mathbf{b}}_i}{y_{ik}} : y_{ik} > 0 \right\} \right\}.$$

   If $I_0$ only contains one element $r$, then $x_r$ leaves the basis. If $I_0$ contains more than one element, that is, the ratio test is a tie, go to the next step.

3. Form the set $I_1$ such that

$$I_1 = \left\{ r : \frac{y_{r1}}{y_{rk}} = \overset{\text{Minimum}}{i \in I_0} \left\{ \frac{y_{i1}}{y_{ik}} : y_{ik} > 0 \right\} \right\}.$$

   If $I_1$ contains only one element $r$, then $x_r$ leaves the basis. Otherwise, this new ratio is a tie.

Table 10: An example of cycling.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | .75 | -20 | .5 | -6 | 0 |
| 0 | 1 | 0 | 0 | .25* | -8 | -1 | 9 | 0 |
| 0 | 0 | 1 | 0 | .5 | -12 | -.5 | 3 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | -3 | 0 | 0 | 0 | 4 | $\frac{7}{2}$ | -33 | 0 |
| 0 | 4 | 0 | 0 | 1 | -32 | -4 | 36 | 0 |
| 0 | -2 | 1 | 0 | 0 | 4* | 1.5 | -15 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | -1 | -1 | 0 | 0 | 0 | 2 | -18 | 0 |
| 0 | -12 | 8 | 0 | 1 | 0 | 8* | -84 | 0 |
| 0 | -.5 | .25 | 0 | 0 | 1 | $\frac{3}{8}$ | $-\frac{15}{14}$ | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 2 | -3 | 0 | -.25 | 0 | 0 | 3 | 0 |
| 0 | -1.5 | 1 | 0 | .125 | 0 | 1 | -10.5 | 0 |
| 0 | $-\frac{1}{16}$ | -.125 | 0 | $-\frac{3}{64}$ | 1 | 0 | $\frac{3}{16}$* | 0 |
| 0 | 1.5 | -1 | 1 | -.125 | 0 | 0 | 10.2 | 1 |
| 1 | 1 | -1 | 0 | .5 | -16 | 0 | 0 | 0 |
| 0 | 2* | -6 | 0 | -2.5 | 56 | 1 | 0 | 0 |
| 0 | $\frac{1}{3}$ | $-\frac{2}{3}$ | 0 | -.25 | $\frac{16}{3}$ | 0 | 1 | 0 |
| 0 | -2 | 6 | 1 | 2.5 | -56 | 0 | 0 | 1 |
| 1 | 0 | 2 | 0 | $\frac{7}{4}$ | -44 | -.5 | 0 | 0 |
| 0 | 1 | -3 | 0 | $-\frac{5}{4}$ | 28 | .5 | 0 | 0 |
| 0 | 0 | $\frac{1}{3}$* | 0 | $\frac{1}{6}$ | -4 | $-\frac{1}{6}$ | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | .75 | -20 | .5 | -6 | 0 |
| 0 | 1 | 0 | 0 | .25* | -8 | -1 | 9 | 0 |
| 0 | 0 | 1 | 0 | .5 | -12 | -.5 | 3 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |

4. If $I_1$ contains more than one element, form $I_2$. The general form for $I_j$ is

$$I_j = \left\{ r : \frac{y_{rj}}{y_{rk}} = \overset{\text{Minimum}}{i \in I_{j-1}} \left\{ \frac{y_{ij}}{y_{ik}} : y_{ik} > 0 \right\} \right\}.$$

5. Keep forming $I_j$s until one of them contains only one element $r$. Then $x_r$ leaves the basis.

The lexicographic method will alway converge (there will always be an $I_j$ that is a singleton) because otherwise the rows would be redundant.

Applying the lexicographic rule to the previous example prevents cycling. Using the lexicographic rule, optimality occurs after only two pivots!

The proof that lexicographic method prevents cycling is moderately difficult to understand, and going into its details is not especially illuminating. Instead of presenting the whole proof, a brief outline will suffice. Before going into the proof outline, however, a new definition is needed:

**Definition 9.1.** *A vector* $\mathbf{x}$ *is called* lexicographically positive *(denoted* $\mathbf{x} \succ \mathbf{0}$*) if the following hold:*

1. $\mathbf{x}$ *is not identically zero*
2. *The first nonzero component of* $\mathbf{x}$ *is positive [2].*

An outline of the proof that the lexicographic method prevents cycling is as follows:

1. Prove that none of the bases generated by the simplex method repeat.
   - Show that each row of the $m \times (m+1)$ matrix $(\bar{\mathbf{b}}, \mathbf{B}^{-1})$ is lexicographically positive at each iteration where $\bar{\mathbf{b}} = \mathbf{B}^{-1}\mathbf{b}$.
     - For a typical row $i$ where $i \neq r$ where $r$ is the pivot row.
     - For the row $i$ where $i = r$.
2. The bases developed by the Simplex algorithm are distinct (proof by contradiction).
3. Since there are a finite number of bases, finite convergence occurs.

The lexicographic rule is complicated, but there are simpler rules to prevent cycling. Bland's rule, explained next, is more straightforward.

## 9.2 Bland's Rule

Bland's rule for selecting entering and leaving basic variables is simpler than the lexicograpic method. First, the variables are ordered arbitrarily from $x_1$ to $x_n$ without loss of generality. Of all the nonbasic variables with positive coefficients in Row 0, the one with the smallest index is chosen to enter. The leaving basic variable is the variable with the smallest index of all the variables who tie in the usual minimum ratio test.

Although this rule is much simpler than the lexicographic rule, it also usually takes a lot longer for the Simplex algorithm to converge using this rule.

## 9.3 Theorem from [2]

This theorem concerns another rule to prevent cycling called Charnes' method, or the perturbation technique. Charnes' method will be shown to be equivalent to the lexicographic rule.

**Theorem** Consider the following problem.

$$\begin{array}{lrcl} \text{Minimize} & \mathbf{cx} & & \\ \text{Subject to} & \mathbf{Ax} & = & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

Assume that the first $m$ columns of $\mathbf{A}$ form the identity and assume that $\mathbf{b} \geq \mathbf{0}$. Given a basis $\mathbf{B}$ the corresponding feasible solution is nondegenerate if $\mathbf{B}^{-1}\mathbf{b} > 0$. Consider the following perturbation procedure of Charnes. Replace $\mathbf{b}$ by $\mathbf{b} + \sum_{j=1}^{m} \mathbf{a}_j \epsilon^j$ where $\epsilon$ is a very small number. Now suppose that we have a basis $\mathbf{B}$ , where $\mathbf{B}^{-1}(\mathbf{b} + \sum_{j=1}^{m} \mathbf{a}_j \epsilon^j) = \bar{\mathbf{b}} + \sum_{j=1}^{m} \mathbf{y}_j \epsilon^j > 0$. Suppose that $x_k$ is chosen to enter the basis and the following minimum ratio test is made:

$$\overset{\text{Minimum}}{1 \leq i \leq m} \left\{ \frac{\bar{\mathbf{b}}_i + \sum_{j=1}^{m} y_{ij} \epsilon^j}{y_{ik}} : y_{ik} > 0 \right\}.$$

Show that this minimum ratio occurs at a unique $r$. Show that this method is equivalent to the lexicographic rule.

**Proof** In the perturbation technique we take a degenerate extreme point with $m$ linearly independent defining hyperplanes and shift each of these hyperplanes just a little bit. By doing so, this one extreme point will become several extreme points, and each of these new extreme points will be nondegenerate.

To do this algebraically, Charnes came up with the idea of changing the

right-hand vector from $\mathbf{b} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ to $\mathbf{b}' = \begin{bmatrix} x_1 + \epsilon \\ x_2 + \epsilon^2 \\ \vdots \\ x_n + \epsilon^n \end{bmatrix}$.

Let the pivot column be column $k$. Suppose in the original problem (before perturbation), the minimum ratio was tied between two rows, row $i$ and row $l$. This means that $\bar{b}_i/y_{ik} = \bar{b}_l/y_{lk}$.

The new minimum ratio test is equivalent to

$$\frac{\bar{b}_i + y_{i1}\epsilon + +y_{i2}\epsilon^2 + \ldots + y_{it}\epsilon^t}{y_{ik}}$$

where $t$ is the number of variables in the system. Therefore, the next pivot row is the minimum of

$$\frac{\bar{b}_i + y_{i1}\epsilon + y_{i2}\epsilon^2 + \ldots + y_{it}\epsilon^t}{y_{ik}}$$

and

$$\frac{\bar{b}_l + y_{l1}\epsilon + y_{l2}\epsilon^2 + \ldots + y_{lt}\epsilon^t}{y_{lk}}.$$

At some point, two terms $y_{ir}/y_{ik}$ and $y_{lr}/y_{lk}$ will be different, or else the constraints are redundant and one of them can be eliminated altogether.

At the index $r$ where the terms are different, the minimum of those terms is the new pivot row. This is true because each successive term $y_{i(r+1)}/y_{ik}$ and $y_{l(r+1)}/y_{lk}$ is an order of magnitude smaller since is it multiplied by another $\epsilon$, and so it does not affect the sum significantly. Therefore, this minimum occurs at a unique index $r$.

This method is equivalent to the lexicographic method since the lexicographic method selects the pivot row in much the same way. Instead of involving $\epsilon$, the lexicographic method simply iterates. First, it defines

$$I_0 = \left\{ r : \frac{\bar{b}_r}{y_{rk}} = \overset{\text{Minimum}}{1 \leq i \leq m} \left\{ \frac{\bar{\mathbf{b}}_i}{y_{ik}} : y_{ik} > 0 \right\} \right\}.$$

If $I_0$ is a singleton, then this ratio occurs at a unique index and that index corresponds to the leaving basic variable. Otherwise,

$$I_j \left\{ r : \frac{y_{rj}}{y_{rk}} = \overset{\text{Minimum}}{i \in I_{j-1}} \left\{ \frac{y_{ij}}{y_{ik}} : y_{ik} > 0 \right\} \right\}.$$

Eventually $I_j$ must be a singleton for some $j \leq m$ or else the tied rows would

be repetitive and one could be eliminated. When a singleton occurs, this is equivalent to the terms in the perturbed minimum ratio sum that are different. The ratios are, in fact, only off by a factor of $\epsilon^r$, which doesn't matter since both terms in the sum have it. Then the lexicographic method terminates, and the rest of the terms in the perturbed minimum ratio sum are negligible since they are orders of magnitude less than the ones before them.

### 9.4 Which Rule to Use?

When solving a linear programming problem with a computer, which no-cycling rule is best? The answer, surprisingly, is none of them.

The lexicographic rule is computationally expensive to implement since it requires many calculations. Bland's rule is inefficient since the Simplex algorithm takes a very long time to converge. Charnes' method, or the perturbation technique, causes problems in computer programs because the value of $\epsilon$ must be predetermined. Using such a small number causes computer round-off errors.

From this information, it is clear that these rules for preventing cycling are more interesting theoretically than practically. In applications of linear programming, problems that are known to be degenerate are ignored or revised until they are nondegenerate.

## 10 Sensitivity Analysis

Sensitivity analysis deals with the effect changing a parameter in a linear program has on the linear program's solution. For example, consider the simple example of Leary Chemicals.

### 10.1 An Example

Leary Chemical produces three chemicals: A, B, and C. Only chemical A can be sold and the other two are polluting byproducts. These chemicals are produced via two production processes: 1 and 2. Running process 1 for an hour costs \$4 and yields 3 units of A, 1 of B, and 1 of C. Running process 2 for an hour costs \$1 and produces 1 unit of A, and 1 of B. To meet customer demands, at least 10 units of A must be produced daily. Also, in order to comply with government regulations, at most 5 units of B and 3 of C can be produced daily.

By assigning variables such that $x_1$ is the number of hours per day running process 1 and $x_2$ is the number of hours per day running process 2, the linear program becomes

$$\begin{array}{lrcl} \text{Minimize} & 4x_1 + x_2 & = & z \\ \text{Subject to} & 3x_1 + x_2 & \geq & 10 \\ & x_1 + x_2 & \leq & 5 \\ & x_1 & \leq & 3 \\ & x_1,\ x_2 & \geq & 0. \end{array}$$

Graphing the constraints and an isocost line for $z = 12.5$ results in Figure 9. From this figure, it is clear that the solution to this problem is $x_2 = 2.5,\ x_1 = 2.5$.

Sensitivity analysis is concerned with questions such as "for what prices of process 1 is this basis still optimal?" or "for what values of customer demand of chemical A is this basis still optimal?"

### 10.1.1 Sensitivity Analysis for a cost coefficient

We can understand and answer these questions through graphical analysis. The first question, concerning the price of process 1, deals with the coefficient $c_1 = 4$. Changing this coefficient changes the isocost line. An isocost line is a line of the form:

$$x_1 = \frac{z}{c_1} - \frac{x_2}{c_1}.$$

Therefore, changing $c_1$ will change both the slope and the $x_1$-intercept of the line. However, the question for what values of $c_1$ does the current basis remain optimal only concerns the slope. To understand this, remember that each extreme point in the feasible region corresponds to another basis. Therefore, what this question really asks is when the optimal solution is at another extreme point.

Notice that as $c_1$ decreases, the slope will become steeper. When $c_2 = 3$ the slope of the isocost line will be the same as the slope of the line corresponding to the first constraint. As seen in Figure 10, once $c_2$ becomes less than 3, the isocost line goes through extreme point 2 with a lowest objective function value. Therefore, the basis corresponding to extreme point 2 is the new basis of the optimal solution. From this analysis, it is clear that $c_1$ must be greater than 3 for the basis to remain at extreme point 1.

As $c_1$ increases, the slope of the isocost line approaches zero, that is, it becomes a horizontal line. This change will ensure that the original basis remains the basis for all $3 < c_1 < \infty$. At $c_1 = 3$ the set of optimal solutions contains all points on the line segment defined by constraint 1 between extreme points 1 and 2. That is, the set of optimal solutions contains an infinite number of points. Of course, $c_1$ cannot realistically equal $\infty$. For all practical purposes, the upper bound on $c_1$ can be interpreted as just "a really big number."

### 10.1.2 Sensitivity Analysis for a right-hand-side value

The second question "for what values of customer demand of chemical A is this basis still optimal?" deals with the right-hand-side of constraint 1. From simple analysis, it is clear that changing $b_1$ moves this constraint parallel to itself. As long as the point where the first and second constraints are binding is optimal, the current basis is optimal.

When shifting the first constraint to the left, note that it defines the optimal isocost line. When this constraint intersects the $x_2$ axis at $b_1 = 5$, three constraints are binding: the first, the second and the nonnegativity constraint on $x_2$. For $b_1 < 5$ the only the first constraint and the nonnegativity constraint on $x_2$ are binding. Therefore, for $b_1 < 5$ the current basis is no longer optimal. This can be seen in Figure 12.

When shifting the first constraint to the right, it still defines the isocost line until $b_1 = 11$. When $b_1 = 11$, the first constraint intersects with the second and the third constraints and the feasible region only contains one point (so that point is optimal). As $b_1$ gets bigger than 11 the feasible region becomes empty. This shift is illustrated in Figure 13. Therefore, if $b_1$ is between 5 and 11, the current basis remains optimal.

# 11 Case Study: Busing Children to School

Now that we have learned how to solve and analyze linear programs quite thoroughly, it is time to apply these skills to a case study. The case study below concerns how to most efficiently bus children from six areas to three schools.

We will use a software program called Lindo to solve this problem. Lindo is a program specifically designed to solve linear programs. It not only outputs an answer, but also a wide variety of sensitivity analysis information. This information will be helpful later when analysing different options available to

Table 11:

| Area | No. of Students | Number in 6th Grade | Number in 7th Grade | Number in 8th Grade | Busing Cost per Student | | |
|---|---|---|---|---|---|---|---|
| | | | | | School 1 | School 2 | School 3 |
| 1 | 450 | 144 | 171 | 135 | $300 | 0 | $700 |
| 2 | 600 | 222 | 168 | 210 | - | $400 | $500 |
| 3 | 550 | 165 | 176 | 209 | $600 | $300 | $200 |
| 4 | 350 | 98 | 140 | 112 | $200 | $500 | - |
| 5 | 500 | 195 | 170 | 135 | 0 | - | $400 |
| 6 | 450 | 153 | 126 | 171 | $500 | $300 | 0 |

us after we solve the initial problem.

## 11.1 The Problem

The Springfield school board has made the decision to close one of its middle schools (sixth, seventh and eighth grades) at the end of this school year and reassign all of next year's middle school students to the three remaining middle schools. The school district provides for all middle school students who must travel more than approximately a mile, so the school board wants a plan for reassigning the students that will minimize the total busing cost. The annual cost per student of busing from each of the six residential areas of the city to each of the schools is shown in Table 11 (along with other basic data for next year), where 0 indicates that busing is not needed and a dash indicates an infeasible assignment.

The school board also has imposed the restriction that each grade must constitute between 30 and 36 percent of each school's population. The table shows each area's middle school population for next year.

We have been hired as an operations research consultant to assist the school board in determining how many students in each area should be assigned to each school. Our job is to formulate and solve a linear program for this problem.

## 11.2 The Solution

### 11.2.1 Variables

The first step to solving this problem is to assign decision variables. We can choose variables in the following way:

$x_{ij}$ – The number of students in sixth grade from area $i$ assigned to school $j$

$s_{ij}$ – The number of students in seventh grade from area $i$ assigned to school $j$

$e_{ij}$ – The number of students in eighth grade from area $i$ assigned to school $j$

where $i = 1, 2, \ldots, 6$ and $j = 1, 2, 3$. Since a dash in Table 1 indicates an infeasible assignment, the variables $x_{21}, x_{52}, x_{43}, s_{21}, s_{52}, s_{43}, e_{21}, e_{52}$, and $e_{43}$ don't exist.

### 11.2.2 Objective Function

Now we must look at the objective function. Since this linear program should minimize busing cost, the objective function is the sum of how much is costs to bus each student. We can write this sum the following way:

$$\begin{aligned}\text{minimize } z \quad = \quad & 300(x_{11} + s_{11} + e_{11}) + 600(x_{31} + s_{31} + e_{31}) + 200(x_{41} + s_{41} + e_{41}) + 500(x_{61} + \\ & s_{61} + e_{61}) + 400(x_{22} + s_{22} + e_{22}) + 300(x_{32} + s_{32} + e_{32}) + 500(x_{42} + s_{42} + e_{42}) + \\ & 300(x_{62} + s_{62} + e_{62}) + 700(x_{13} + s_{13} + e_{13}) + 500(x_{23} + s_{23} + e_{23}) + 200(x_{33} + \\ & s_{33} + e_{33}) + 400(x_{53} + s_{53} + e_{53}).\end{aligned}$$

### 11.2.3 Constraints

There are three kinds of constraints in this linear program. One kind limits the capacity of each school, so we will call them "capacity constraints." "Grade constraints" will state how many students in each area are in each grade. Finally, there will be several "percentage constraints" limiting the percentages of students in each grade at each school.

Capacity constraints are easy to see. A school's capacity can be limited by summing over $i$ for each school and for each variable. These constraints are:

$$
\begin{aligned}
x_{11}+x_{31}+x_{41}+x_{51}+x_{61}+s_{11}+s_{31}+s_{41}+s_{51}+s_{61}+e_{11}+e_{31}+e_{41}+e_{51}+e_{61} &\leq 900\\
x_{12}+x_{22}+x_{32}+x_{42}+x_{62}+s_{12}+s_{22}+s_{32}+s_{42}+s_{62}+e_{12}+e_{22}+e_{32}+e_{42}+e_{62} &\leq 110\\
x_{13}+x_{23}+x_{33}+x_{53}+x_{63}+s_{13}+s_{23}+s_{33}+s_{53}+s_{63}+e_{13}+e_{23}+e_{33}+e_{53}+e_{63} &\leq 1000.
\end{aligned}
$$

The total number of students in each grade can also be easily read from Table 11. For sixth grade students, these constraints are:

$$
\begin{aligned}
x_{11}+x_{12}+x_{13} &= 144\\
x_{22}+x_{23} &= 222\\
x_{31}+x_{32}+x_{33} &= 165\\
x_{41}+x_{42} &= 98\\
x_{51}+x_{53} &= 195\\
x_{61}+x_{62}+x_{63} &= 153.
\end{aligned}
$$

For seventh grade students, these constraints are:

$$
\begin{aligned}
s_{11}+s_{12}+s_{13} &= 171\\
s_{22}+s_{23} &= 168\\
s_{31}+s_{32}+s_{33} &= 176\\
s_{41}+s_{42} &= 140\\
s_{51}+s_{53} &= 170\\
s_{61}+s_{62}+s_{63} &= 126.
\end{aligned}
$$

For eighth grade students, these constraints are:

$$\begin{aligned} e_{11} + e_{12} + e_{13} &= 135 \\ e_{22} + e_{23} &= 210 \\ e_{31} + e_{32} + e_{33} &= 209 \\ e_{41} + e_{42} &= 112 \\ e_{51} + e_{53} &= 135 \\ e_{61} + e_{62} + e_{63} &= 171. \end{aligned}$$

Some of these constraints are shorter than others because of the infeasibility of assigning students from area 2 to school 1, or students from area 4 to school 3, or students from area 5 to school 2.

The percentage constraints are probably the most difficult part of this problem. To give an example of how these are formulated, let's consider the percentage of sixth graders at school 1. This number must be in between 30 and 36 percent. In rough form, this constraint is:

$$.3 \leq \frac{x_{11} + x_{31} + x_{41} + x_{51} + x_{61}}{x_{11} + x_{31} + x_{41} + x_{51} + x_{61} + s_{11} + s_{31} + s_{41} + s_{51} + s_{61} + e_{11} + e_{31} + e_{41} + e_{51} + e_{61}} \leq .36$$

After multiplying both sides by the denominator of the middle term, splitting this inequality into its two parts, and subtracting until all variables are on one side, this constraint is *two* constraints in standard form. These constraints are:

$$\begin{aligned} .64x_{11} + .64x_{31} + .64x_{41} + .64x_{51} + .63x_{61} - .36s_{11} - .36s_{31} & \\ -.36s_{41} - .36s_{51} - .36s_{61} - .36e_{11} - .36e_{31} - .36e_{41} - .36e_{51} - .36e_{61} &\leq 0 \\ .7x_{11} + .7x_{31} + .7x_{41} + .7x_{51} + .7x_{61} - .3s_{11} - .3s_{31} & \\ -.3s_{41} - .3s_{51} - .3s_{61} - .3e_{11} - .3e_{31} - .3e_{41} - .3e_{51} - .3e_{61} &\geq 0. \end{aligned}$$

Doing this for sixth graders at all schools, the percentage constraints are:

$$
\begin{aligned}
.64x_{11} + .64x_{31} + .64x_{41} + .64x_{51} + .63x_{61} - .36s_{11} - .36s_{31} & \\
-.36s_{41} - .36s_{51} - .36s_{61} - .36e_{11} - .36e_{31} - .36e_{41} - .36e_{51} - .36e_{61} & \leq 0 \\
.7x_{11} + .7x_{31} + .7x_{41} + .7x_{51} + .7x_{61} - .3s_{11} - .3s_{31} & \\
-.3s_{41} - .3s_{51} - .3s_{61} - .3e_{11} - .3e_{31} - .3e_{41} - .3e_{51} - .3e_{61} & \geq 0. \\
.64x_{12} + .64x_{22} + .64x_{32} + .64x_{42} + .63x_{62} - .36s_{12} - .36s_{22} - .36s_{32} & \\
-.36s_{42} - .36s_{62} - .36e_{12} - .36e_{22} - .36e_{32} - .36e_{42} - .36e_{62} & \leq 0 \\
.7x_{12} + .7x_{22} + .7x_{32} + .7x_{42} + .7x_{62} - .3s_{12} - .3s_{22} - .3s_{32} & \\
-.3s_{42} - .3s_{62} - .3e_{12} - .3e_{22} - .3e_{32} - .3e_{42} - .3e_{62} & \geq 0. \\
.64x_{13} + .64x_{23} + .64x_{33} + .64x_{53} + .64x_{63} - .36s_{13} - .36s_{23} - .36s_{33} & \\
-.36s_{53} - .36s_{63} - .36e_{13} - .36e_{23} - .36e_{33} - .36e_{53} - .36e_{63} & \leq 0 \\
.7x_{13} + .7x_{23} + .7x_{33} + .7x_{53} + .7x_{63} - .3s_{13} - .3s_{23} - .3s_{33} & \\
-.3s_{53} - .3s_{63} - .3e_{13} - .3e_{23} - .3e_{33} - .3e_{53} - .3e_{63} & \geq 0.
\end{aligned}
$$

For all seventh graders at all schools, the percentage constraints are:

$$
\begin{aligned}
.64s_{11} + .64s_{31} + .64s_{41} + .64s_{51} + .64s_{61} - .36x_{11} - .36x_{31} & \\
-.36x_{41} - .36x_{51} - .36x_{61} - .36e_{11} - .36e_{31} - .36e_{41} - .36e_{51} - .36e_{61} & \leq 0 \\
.7s_{11} + .7s_{31} + .7s_{41} + .7s_{51} + .7s_{61} - .3x_{11} - .3x_{31} & \\
-.3x_{41} - .3x_{51} - .3x_{61} - .3e_{11} - .3e_{31} - .3e_{41} - .3e_{51} - .3e_{61} & \geq 0. \\
.64s_{12} + .64s_{22} + .64s_{32} + .64s_{42} + .64s_{62} - .36x_{12} - .36x_{22} - .36x_{32} & \\
-.36x_{42} - .36x_{62} - .36e_{12} - .36e_{22} - .36e_{32} - .36e_{42} - .36e_{62} & \leq 0 \\
.7s_{12} + .7s_{22} + .7s_{32} + .7s_{42} + .7s_{62} - .3x_{12} - .3x_{22} - .3x_{32} & \\
-.3x_{42} - .3x_{62} - .3e_{12} - .3e_{22} - .3e_{32} - .3e_{42} - .3e_{62} & \geq 0. \\
.64s_{13} + .64s_{23} + .64s_{33} + .64s_{53} + .64s_{63} - .36x_{13} - .36x_{23} - .36x_{33} & \\
-.36x_{53} - .36x_{63} - .36e_{13} - .36e_{23} - .36e_{33} - .36e_{53} - .36e_{63} & \leq 0 \\
.7s_{13} + .7s_{23} + .7s_{33} + .7s_{53} + .7s_{63} - .3x_{13} - .3x_{23} - .3x_{33} & \\
-.3x_{53} - .3x_{63} - .3e_{13} - .3e_{23} - .3e_{33} - .3e_{53} - .3e_{63} & \geq 0.
\end{aligned}
$$

For all eighth graders at all schools, the percentage constraints are:

$$\begin{aligned}
.64e_{11} + .64e_{31} + .64e_{41} + .64e_{51} + .64e_{61} - .36x_{11} - .36x_{31} & \\
-.36x_{41} - .36x_{51} - .36x_{61} - .36s_{11} - .36s_{31} - .36s_{41} - .36s_{51} - .36s_{61} &\leq 0 \\
.7e_{11} + .7e_{31} + .7e_{41} + .7e_{51} + .7e_{61} - .3x_{11} - .3x_{31} & \\
-.3x_{41} - .3x_{51} - .3x_{61} - .3s_{11} - .3s_{31} - .3s_{41} - .3s_{51} - .3s_{61} &\geq 0. \\
.64e_{12} + .64e_{22} + .64e_{32} + .64e_{42} + .64e_{62} - .36x_{12} - .36x_{22} - .36x_{32} & \\
-.36x_{42} - .36x_{62} - .36s_{12} - .36s_{22} - .36s_{32} - .36s_{42} - .36s_{62} &\leq 0 \\
.7e_{12} + .7e_{22} + .7e_{32} + .7e_{42} + .7e_{62} - .3x_{12} - .3x_{22} - .3x_{32} & \\
-.3x_{42} - .3x_{62} - .3s_{12} - .3s_{22} - .3s_{32} - .3s_{42} - .3s_{62} &\geq 0. \\
.64e_{13} + .64e_{23} + .64e_{33} + .64e_{53} + .64e_{63} - .36x_{13} - .36x_{23} - .36x_{33} & \\
-.36x_{53} - .36x_{63} - .36s_{13} - .36s_{23} - .36s_{33} - .36s_{53} - .36s_{63} &\leq 0 \\
.7e_{13} + .7e_{23} + .7e_{33} + .7e_{53} + .7e_{63} - .3x_{13} - .3x_{23} - .3x_{33} & \\
-.3x_{53} - .3x_{63} - .3s_{13} - .3s_{23} - .3s_{33} - .3s_{53} - .3s_{63} &\geq 0.
\end{aligned}$$

## 11.3 The Complete Program

Putting all of these constraints together in standard form, the final linear program is:

minimize

$$\begin{aligned}
z = {} & 300x_{11} + 300s_{11} + 300e_{11} + 600x_{31} + 600s_{31} + 600e_{31} + 200x_{41} + 200s_{41} + 200e_{41} + \\
& 500x_{61} + 500s_{61} + 500e_{61} + 400x_{22} + 400s_{22} + 400e_{22} + 300x_{32} + 300s_{32} + 300e_{32} + \\
& 500x_{42} + 500s_{42} + 500e_{42} + 300x_{62} + 300s_{62} + 300e_{62} + 700x_{13} + 700s_{13} + 700e_{13} + \\
& 500x_{23} + 500s_{23} + 500e_{23} + 200x_{33} + 200s_{33} + 200e_{33} + 400x_{53} + 400s_{53} + 400e_{53}
\end{aligned}$$

such that

$$\begin{aligned}
x_{11} + x_{31} + x_{41} + x_{51} + x_{61} + s_{11} + s_{31} + s_{41} + s_{51} + s_{61} + e_{11} + e_{31} + e_{41} + e_{51} + e_{61} &\leq 900 \\
x_{12} + x_{22} + x_{32} + x_{42} + x_{62} + s_{12} + s_{22} + s_{32} + s_{42} + s_{62} + e_{12} + e_{22} + e_{32} + e_{42} + e_{62} &\leq 110 \\
x_{13} + x_{23} + x_{33} + x_{53} + x_{63} + s_{13} + s_{23} + s_{33} + s_{53} + s_{63} + e_{13} + e_{23} + e_{33} + e_{53} + e_{63} &\leq 1000
\end{aligned}$$

$$
\begin{aligned}
x_{11} + x_{12} + x_{13} &= 144 \\
x_{22} + x_{23} &= 222 \\
x_{31} + x_{32} + x_{33} &= 165 \\
x_{41} + x_{42} &= 98 \\
x_{51} + x_{53} &= 195 \\
x_{61} + x_{62} + x_{63} &= 153 \\
s_{11} + s_{12} + s_{13} &= 171 \\
s_{22} + s_{23} &= 168 \\
s_{31} + s_{32} + s_{33} &= 176 \\
s_{41} + s_{42} &= 140 \\
s_{51} + s_{53} &= 170 \\
s_{61} + s_{62} + s_{63} &= 126 \\
e_{11} + e_{12} + e_{13} &= 135 \\
e_{22} + e_{23} &= 210 \\
e_{31} + e_{32} + e_{33} &= 209 \\
e_{41} + e_{42} &= 112 \\
e_{51} + e_{53} &= 135 \\
e_{61} + e_{62} + e_{63} &= 171 \\
.64x_{11} + .64x_{31} + .64x_{41} + .64x_{51} + .63x_{61} - .36s_{11} - .36s_{31} & \\
-.36s_{41} - .36s_{51} - .36s_{61} - .36e_{11} - .36e_{31} - .36e_{41} - .36e_{51} - .36e_{61} &\leq 0 \\
.7x_{11} + .7x_{31} + .7x_{41} + .7x_{51} + .7x_{61} - .3s_{11} - .3s_{31} & \\
-.3s_{41} - .3s_{51} - .3s_{61} - .3e_{11} - .3e_{31} - .3e_{41} - .3e_{51} - .3e_{61} &\geq 0. \\
.64x_{12} + .64x_{22} + .64x_{32} + .64x_{42} + .63x_{62} - .36s_{12} - .36s_{22} - .36s_{32} & \\
-.36s_{42} - .36s_{62} - .36e_{12} - .36e_{22} - .36e_{32} - .36e_{42} - .36e_{62} &\leq 0 \\
.7x_{12} + .7x_{22} + .7x_{32} + .7x_{42} + .7x_{62} - .3s_{12} - .3s_{22} - .3s_{32} & \\
-.3s_{42} - .3s_{62} - .3e_{12} - .3e_{22} - .3e_{32} - .3e_{42} - .3e_{62} &\geq 0. \\
.64x_{13} + .64x_{23} + .64x_{33} + .64x_{53} + .64x_{63} - .36s_{13} - .36s_{23} - .36s_{33} & \\
-.36s_{53} - .36s_{63} - .36e_{13} - .36e_{23} - .36e_{33} - .36e_{53} - .36e_{63} &\leq 0 \\
.7x_{13} + .7x_{23} + .7x_{33} + .7x_{53} + .7x_{63} - .3s_{13} - .3s_{23} - .3s_{33} & \\
-.3s_{53} - .3s_{63} - .3e_{13} - .3e_{23} - .3e_{33} - .3e_{53} - .3e_{63} &\geq 0. \\
.64s_{11} + .64s_{31} + .64s_{41} + .64s_{51} + .64s_{61} - .36x_{11} - .36x_{31} & \\
-.36x_{41} - .36x_{51} - .36x_{61} - .36e_{11} - .36e_{31} - .36e_{41} - .36e_{51} - .36e_{61} &\leq 0 \\
.7s_{11} + .7s_{31} + .7s_{41} + .7s_{51} + .7s_{61} - .3x_{11} - .3x_{31} & \\
-.3x_{41} - .3x_{51} - .3x_{61} - .3e_{11} - .3e_{31} - .3e_{41} - .3e_{51} - .3e_{61} &\geq 0. \\
.64s_{12} + .64s_{22} + .64s_{32} + .64s_{42} + .64s_{62} - .36x_{12} - .36x_{22} - .36x_{32} & \\
-.36x_{42} - .36x_{62} - .36e_{12} - .36e_{22} - .36e_{32} - .36e_{42} - .36e_{62} &\leq 0
\end{aligned}
$$

$$
\begin{aligned}
.7s_{12} + .7s_{22} + .7s_{32} + .7s_{42} + .7s_{62} - .3x_{12} - .3x_{22} - .3x_{32} & \\
-.3x_{42} - .3x_{62} - .3e_{12} - .3e_{22} - .3e_{32} - .3e_{42} - .3e_{62} &\geq 0. \\
.64s_{13} + .64s_{23} + .64s_{33} + .64s_{53} + .64s_{63} - .36x_{13} - .36x_{23} - .36x_{33} & \\
-.36x_{53} - .36x_{63} - .36e_{13} - .36e_{23} - .36e_{33} - .36e_{53} - .36e_{63} &\leq 0 \\
.7s_{13} + .7s_{23} + .7s_{33} + .7s_{53} + .7s_{63} - .3x_{13} - .3x_{23} - .3x_{33} & \\
-.3x_{53} - .3x_{63} - .3e_{13} - .3e_{23} - .3e_{33} - .3e_{53} - .3e_{63} &\geq 0. \\
.64e_{11} + .64e_{31} + .64e_{41} + .64e_{51} + .64e_{61} - .36x_{11} - .36x_{31} & \\
-.36x_{41} - .36x_{51} - .36x_{61} - .36s_{11} - .36s_{31} - .36s_{41} - .36s_{51} - .36s_{61} &\leq 0 \\
.7e_{11} + .7e_{31} + .7e_{41} + .7e_{51} + .7e_{61} - .3x_{11} - .3x_{31} & \\
-.3x_{41} - .3x_{51} - .3x_{61} - .3s_{11} - .3s_{31} - .3s_{41} - .3s_{51} - .3s_{61} &\geq 0. \\
.64e_{12} + .64e_{22} + .64e_{32} + .64e_{42} + .64e_{62} - .36x_{12} - .36x_{22} - .36x_{32} & \\
-.36x_{42} - .36x_{62} - .36s_{12} - .36s_{22} - .36s_{32} - .36s_{42} - .36s_{62} &\leq 0 \\
.7e_{12} + .7e_{22} + .7e_{32} + .7e_{42} + .7e_{62} - .3x_{12} - .3x_{22} - .3x_{32} & \\
-.3x_{42} - .3x_{62} - .3s_{12} - .3s_{22} - .3s_{32} - .3s_{42} - .3s_{62} &\geq 0. \\
.64e_{13} + .64e_{23} + .64e_{33} + .64e_{53} + .64e_{63} - .36x_{13} - .36x_{23} - .36x_{33} & \\
-.36x_{53} - .36x_{63} - .36s_{13} - .36s_{23} - .36s_{33} - .36s_{53} - .36s_{63} &\leq 0 \\
.7e_{13} + .7e_{23} + .7e_{33} + .7e_{53} + .7e_{63} - .3x_{13} - .3x_{23} - .3x_{33} & \\
-.3x_{53} - .3x_{63} - .3s_{13} - .3s_{23} - .3s_{33} - .3s_{53} - .3s_{63} &\geq 0.
\end{aligned}
$$

After solving this program on Lindo software, we obtain the objective function value of \$555555.60.

## 11.4 Road Construction and Portables

### 11.4.1 Construction

Reconsider the previous problem of busing students from 6 areas to 3 schools. Now the school board is concerned about the ongoing road construction in area 6. The construction may increase busing costs for students in area 6 by up to 10 percent. Using the sensitivity report given by Lindo when we solved the problem in the previous section, we can account for construction costs. This report is shown in Table 12.

Since the construction will affect only the busing cost for students in area 6,

the relevant information in the sensitivity report in Table 14 concerns the allowable increase (AI) of the objective function coefficient for variables $x_{61}$, $x_{62}$ and $x_{63}$. The current cost coefficient of $x_{61}$ is \$500 and its AI is \$33.33. Therefore, if the cost of busing students in area 6 increased 10% (\$50), this basis would not remain optimal. The current cost coefficient of $x_{62}$ is \$300 and its AI is infinity. Therefore, increasing the cost coefficient of $x_{62}$ by 10% would not affect this basis. The current cost coefficient of $x_{63}$ is \$0, and therefore increasing this cost coefficient by any percentage would still yield \$0. Therefore, the only area 6 cost coefficient that would change the basis if it were increased is the cost coefficient of $x_{61}$.

Solving the problem again, this time with the objective function coefficients of $x_{61}$ and $x_{62}$ increased by 10%, we obtain a new optimal solution for the school board. This new result is shown in Table 13. This adjustment increases the objective function value by over \$10,000. Also, some basic variables have become nonbasic and vice versa.

### 11.4.2 Portable Classrooms

The next day the school board calls and says they can buy portable classrooms for \$2,500 per year, that hold 20 students each. Now it must be determined whether or not increasing each school's capacity by multiples of 20 (at a cost of \$2,500 for each multiple) would increase or decrease busing costs. What are being changed here are the right-hand-side values of the capacity constraints. Analyzing the shadow prices and AIs of these constraints would solve the problem, but it may actually be easier to make minor adjustments to the linear program itself. Defining new variables $p_1$, $p_2$, and $p_3$ to represent the number of portables bought for schools 1, 2 and 3 respectively, everything about the program stays the same except the capacity constraints and the objective function. The objective function becomes:

$$\begin{aligned} \min z \quad = \quad & 300x_{11} + 600x_{31} + 200x_{41} + 500x_{61} + 400x_{22} + 300x_{32} + 500x_{42} + \\ & 300x_{62} + 700x_{13} + 500x_{23} + 200x_{33} + 400x_{53} + 2500p_1 + 2500p_2 + 2500p_3. \end{aligned}$$

The capacity constraints become:

Table 12:

* ORIGINAL OBJECTIVE FUNCTION VALUE: 555555.6

| VARIABLE | VALUE | REDUCED COST |
|---|---|---|
| X11 | 0.000000 | 177.777771 |
| X31 | 0.000000 | 11.111114 |
| X41 | 350.000000 | 0.000000 |
| X61 | 83.333336 | 0.000000 |
| X22 | 422.222229 | 0.000000 |
| X32 | 227.777771 | 0.000000 |
| X42 | 0.000000 | 366.666656 |
| X62 | 0.000000 | 200.000000 |
| X13 | 0.000000 | 266.666687 |
| X23 | 177.777771 | 0.000000 |
| X33 | 322.222229 | 0.000000 |
| X53 | 133.333328 | 0.000000 |
| X51 | 366.666656 | 0.000000 |
| X12 | 450.000000 | 0.000000 |
| X63 | 366.666656 | 0.000000 |
| X21 | 0.000000 | 0.000000 |
| X43 | 0.000000 | 0.000000 |
| X52 | 0.000000 | 0.000000 |
| ROW | SLACK OR SURPLUS | DUAL PRICES |
| 2) | 100.000000 | 0.000000 |
| 3) | 0.000000 | 177.777771 |
| 4) | 0.000000 | 144.444443 |
| 5) | 0.000000 | 0.000000 |
| 6) | 0.000000 | 0.000000 |
| 7) | 0.000000 | 0.000000 |
| 8) | 0.000000 | -177.777771 |
| 9) | 0.000000 | -577.777771 |
| 10) | 0.000000 | -477.777771 |
| 11) | 0.000000 | -311.111115 |
| 12) | 0.000000 | 55.555557 |
| 13) | 0.000000 | -277.777771 |
| 14) | 29.333334 | 0.000000 |
| 15) | 18.666666 | 0.000000 |
| 16) | 38.555557 | 0.000000 |
| 17) | 27.444445 | 0.000000 |
| 18) | 39.111111 | 0.000000 |
| 19) | 20.888889 | 0.000000 |
| 20) | 48.000000 | 0.000000 |
| 21) | 0.000000 | 2777.777832 |
| 22) | 32.111111 | 0.000000 |
| 23) | 33.888889 | 0.000000 |
| 24) | 0.888889 | 0.000000 |
| 25) | 59.111111 | 0.000000 |
| 26) | 2.666667 | 0.000000 |
| 27) | 45.333332 51 | 0.000000 |
| 28) | 39.333332 | 0.000000 |
| 29) | 26.666666 | 0.000000 |
| 30) | 60.000000 | 0.000000 |
| 31) | 0.000000 | 6666.666504 |

* NO. ITERATIONS= 13

Table 13: after changing the cost coefficients for $x_{61}$ and $x_{62}$

| * OBJECTIVE FUNCTION VALUE: 567000.0 | | |
|---|---|---|
| VARIABLE | VALUE | REDUCED COST |
| X11 | 0.000000 | 420.000000 |
| X31 | 185.000000 | 0.000000 |
| X41 | 350.000000 | 0.000000 |
| X61 | 0.000000 | 70.000015 |
| X22 | 600.000000 | 0.000000 |
| X32 | 50.000000 | 0.000000 |
| X42 | 0.000000 | 40.000000 |
| X62 | 0.000000 | 350.000000 |
| X13 | 0.000000 | 320.000000 |
| X23 | 0.000000 | 20.000002 |
| X33 | 315.000000 | 0.000000 |
| X53 | 170.000000 | 0.000000 |
| X51 | 330.000000 | 0.000000 |
| X12 | 450.000000 | 0.000000 |
| X63 | 450.000000 | 0.000000 |
| X21 | 0.000000 | 0.000000 |
| X43 | 0.000000 | 0.000000 |
| X52 | 0.000000 | 0.000000 |

$$\begin{aligned} x_{11} + x_{31} + x_{41} + x_{51} + x_{61} - 20p_1 &\leq 900 \\ x_{12} + x_{22} + x_{32} + x_{42} + x_{62} - 20p_2 &\leq 1100 \\ x_{13} + x_{23} + x_{33} + x_{53} + x_{63} - 20p_3 &\leq 1000. \end{aligned}$$

Solving this linear program yields the solution shown below in Table 15. This objective function value is almost a $2,000 improvement. Note that the basis for this linear program is the same basis as in the original linear program. This solution is clearly not realistic since the values of the variables are not integers. In Lindo, this is easy enough to fix with one simple command, but the theory behind the complexities of integer programming is beyond the scope of this problem. However, in the next section, bit integer programming must be used to solve a new problem with assigning students.

Table 14: RANGES IN WHICH THE BASIS IS UNCHANGED:

| | | * OBJ COEFFICIENT RANGES | |
|---|---|---|---|
| VARIABLE | CURRENT COEF | ALLOWABLE INCREASE | ALLOWABLE DECREASE |
| X11 | 300.000000 | INFINITY | 177.777786 |
| X31 | 600.000000 | INFINITY | 11.111118 |
| X41 | 200.000000 | 366.666656 | INFINITY |
| X61 | 500.000000 | 33.333256 | 166.666672 |
| X22 | 400.000000 | 34.210526 | 4.545444 |
| X32 | 300.000000 | 4.545444 | 34.210526 |
| X42 | 500.000000 | INFINITY | 366.666656 |
| X62 | 300.000000 | INFINITY | 200.000000 |
| X13 | 700.000000 | INFINITY | 266.666687 |
| X23 | 500.000000 | 4.545444 | 34.210526 |
| X33 | 200.000000 | 34.210526 | 7.692289 |
| X53 | 400.000000 | 108.333328 | 16.666624 |
| X51 | 0.000000 | 16.666624 | 108.333328 |
| X12 | 0.000000 | 177.777786 | INFINITY |
| X63 | 0.000000 | 166.666672 | 33.333256 |
| X21 | 0.000000 | INFINITY | INFINITY |
| X43 | 0.000000 | INFINITY | INFINITY |
| X52 | 0.000000 | INFINITY | INFINITY |
| | | * RIGHTHAND SIDE RANGES | |
| ROW | CURRENT RHS | ALLOWABLE INCREASE | ALLOWABLE DECREASE |
| 2 | 900.000000 | INFINITY | 100.000000 |
| 3 | 1100.000000 | 36.363636 | 3.773585 |
| 4 | 1000.000000 | 42.105259 | 3.883495 |
| 5 | 0.000000 | INFINITY | 0.000000 |
| 6 | 0.000000 | INFINITY | 0.000000 |
| 7 | 0.000000 | INFINITY | 0.000000 |
| 8 | 450.000000 | 3.773585 | 36.363636 |
| 9 | 600.000000 | 3.773585 | 36.363636 |
| 10 | 550.000000 | 3.773585 | 36.363636 |
| 11 | 350.000000 | 72.727272 | 6.451613 |
| 12 | 500.000000 | 12.903226 | 145.454544 |
| 13 | 450.000000 | 3.225806 | 36.363636 |
| 14 | 0.000000 | 29.333334 | INFINITY |
| 15 | 0.000000 | INFINITY | 18.666666 |
| 16 | 0.000000 | 38.555557 | INFINITY |
| 17 | 0.000000 | INFINITY | 27.444445 |
| 18 | 0.000000 | 39.111111 | INFINITY |
| 19 | 0.000000 | INFINITY | 20.888889 |
| 20 | 0.000000 | 48.000000 | INFINITY |
| 21 | 0.000000 | 0.258065 | 2.909091 |
| 22 | 0.000000 | 32.111111 | INFINITY |
| 23 | 0.000000 | INFINITY | 33.888889 |
| 24 | 0.000000 | 0.888889 | INFINITY |
| 25 | 0.000000 | INFINITY | 59.111111 |
| 26 | 0.000000 | 2.666667 | INFINITY |
| 27 | 0.000000 | INFINITY | 45.333332 |
| 28 | 0.000000 | 39.333332 | INFINITY |
| 29 | 0.000000 | INFINITY | 26.666666 |
| 30 | 0.000000 | 60.000000 | INFINITY |
| 31 | 0.000000 | 5.333333 | 0.666667 |

Table 15:

| * OBJECTIVE FUNCTION VALUE: 553636.4 | | |
|---|---|---|
| VARIABLE | VALUE | REDUCED COST |
| X11 | 0.000000 | 243.750000 |
| X31 | 0.000000 | 37.500004 |
| X41 | 350.000000 | 0.000000 |
| X61 | 95.454544 | 0.000000 |
| X22 | 600.000000 | 0.000000 |
| X32 | 86.363640 | 0.000000 |
| X42 | 0.000000 | 287.500000 |
| X62 | 0.000000 | 200.000000 |
| X13 | 0.000000 | 295.454559 |
| X23 | 0.000000 | 10.795456 |
| X33 | 463.636353 | 0.000000 |
| X53 | 181.818176 | 0.000000 |
| P1 | 0.000000 | 2500.000000 |
| P2 | 1.818182 | 0.000000 |
| P3 | 0.000000 | 522.727295 |
| X51 | 318.181824 | 0.000000 |
| X12 | 450.000000 | 0.000000 |
| X63 | 354.545441 | 0.000000 |
| X21 | 0.000000 | 0.000000 |
| X43 | 0.000000 | 0.000000 |
| X52 | 0.000000 | 0.000000 |

## 11.5 Keeping Neighborhoods Together

Now the school board wants to prohibit the splitting of residential areas among multiple schools. That is, each of the six areas must be assigned to a single school. This problem can be solved with bit integer programming, a type of linear programming in which the variables can only take on values of 0 or 1. For this problem, we can define the variables $b_{ij}$ as 1 if students from area i are assigned to school j, 0 if they are not. In this problem, we can abandon the percentage constraints. The objective function for this problem is:

$$\begin{aligned} \min z \quad = \quad & 135000b_{11} + 330000b_{31} + 70000b_{41} + 225000b_{61} + 240000b_{22} + 165000b_{32} + 175000b_{42} \\ & +135000b_{62} + 315000b_{13} + 3000000b_{23} + 110000b_{33} + 200000b_{53}. \end{aligned}$$

The school capacity constraints are:

$$\begin{aligned} 450b_{11} + 550b_{31} + 350b_{41} + 500b_{51} + 450b_{61} &< 900 \\ 450b_{12} + 550b_{22} + 350b_{32} + 500b_{42} + 450b_{62} &< 1100 \\ 450b_{13} + 550b_{23} + 350b_{33} + 500b_{53} + 450b_{63} &< 1000. \end{aligned}$$

The next constraints ensure that each area is assigned to only school. They are:

$$\begin{aligned} 450b_{11} + 450b_{12} + 450b_{13} &= 450 \\ 600b_{22} + 600b_{23} &= 600 \\ 550b_{31} + 550b_{32} + 550b_{33} &= 550 \\ 350b_{41} + 350b_{42} &= 350 \\ 500b_{51} + 500b_{53} &= 500 \\ 450b_{61} + 450b_{62} + 450b_{63} &= 450. \end{aligned}$$

The solution to this linear program is that students from areas 4 and 5 go to school 1, students from areas 1 and 2 go to school 2, and students from areas 3

and 6 go to school 3. By inspecting the busing costs associated with each area and school, this result seems reasonable.

## 11.6 The Case Revisited

Recall that the school board has the option of adding portable classrooms to increase the capacity of one or more of the middle schools. Each portable classrooms holds 20 students and costs $2,500 per year. The operations research team must decide if any of the schools should have these portable classrooms, and how many they should have.

Earlier, this was accomplished by adding a few new variables to the linear program and resolving. Using the Lindo software, this procedure was actually quite simple. The result was that 1.818 classrooms should be added to school 2 (or, $p_2 = 1.818$), as seen in Table 15.

### 11.6.1 Shadow Prices

Now, the operations research team will verify this solution using something called a *shadow price*.

**Definition 11.1.** *In linear programming problem, the* shadow price *of the $i^{th}$ constraint is the amount by which the optimal $z$-value is improved if the right-hand side of the $i^{th}$ constraint is increased by 1 (assuming the current basis remains optimal) [3].*

For a minimization problem, the new $z$-value is given by:

$$z_{new} = z_{old} - SP_i \Delta b_i,$$

where $SP_i$ is the shadow price of the $i^{th}$ constraint and $\Delta b_i$ is the amount by which the right-hand side of the $i^{th}$ constraint changes.

This problem concerns the shadow prices of the capacity constraints of the linear program. The school board would like to know how increasing these constraints by multiples of twenty would affect the optimal $z$-value.

Shadow prices in the Lindo sensitivity analysis output screen are called "Dual Prices." For a "greater than or equal to" constraint, these values are nonpositive, since increasing the right-hand side of this constraint will eliminate points from the feasible region. Therefore, the new optimal solution can stay the same, or get worse. Similar reasoning shows that a "less than or equal to" constraint

will have a nonnegative shadow price, and an equality constraint's shadow price can be either positive, negative, or zero [3].

### 11.6.2 The New Result

In this case, the operations research team is concerned only with the shadow prices of the school capacity constraints. For the capacity constraints these values are 0, 177.78, and 144.44 for schools 1, 2, and 3 respectively. Just from looking at these shadow prices, it is clear that changing the capacity of school 1 will have no effect on the objective function. This school also is not filled to capacity in the current basis, and therefore it is bad business sense to consider increasing its capacity.

Since school 2 has a larger-valued shadow price, increasing the capacity of school 2 seems favorable to increasing the capacity of school 3. Since the solutions found last week are decimals and not integers, this solution should be decimals as well for the sake of comparison.

Notice that the allowable increase of constraint 2 is 36.36. If we divide this by 20, the result, 1.818, is the number of portable classrooms that can be added at school 2. This is precisely the number obtained in Table 15 for the variable $p_2$. According to the equation in section 2, the new $z$-value is

$$555555.6 - 177.78 \cdot 36.36 = 548091.42.$$

This isn't the same as the $z$-value found previously, because the price of the portables hasn't yet been added. Adding $(2500 \cdot 1.818)$ gives 553636.4, which is our final answer shown in Table 15. Since this answer is less than then answer to the original problem, we can conclude that adding portables is a viable solution.

# 12 Conclusion

As a next step, we reccommend studying integer programming. With this tool, we could obtain more practical results. For example, in the case study, 1.818 portable classrooms is an unrelastic answer, but 2 portable classrooms would be a better answer because 2 is an integer. Also, the theory of binary integer programming would be helpful for understanding how we kept the neighborhoods together in Section 11.5.

Now we have a solid background in linear programming. We are acquainted with the theory behind linear programs and we know the basis tools used to

solve them. However, the field of linear programming is so large that we have only touched the tip of the iceberg.

# References


[1] E. F. Robertson J. J. O'Connor. George Dantzig. *http://www-history.mcs.st-andrews.ac.uk/Biographies/Dantzig_George.html*, University of St. Andrews, 2003.

[2] John J. Jarvis Mokhtar S. Bazaraa and Hanif D. Sherali. *Linear Programming and Network Flows*. John Wiley & Sons, second edition, 1990.

[3] Wayne L. Winston. *Operations Research: Applications and Algorithms*. Duxbury Press, fourth edition, 2003.

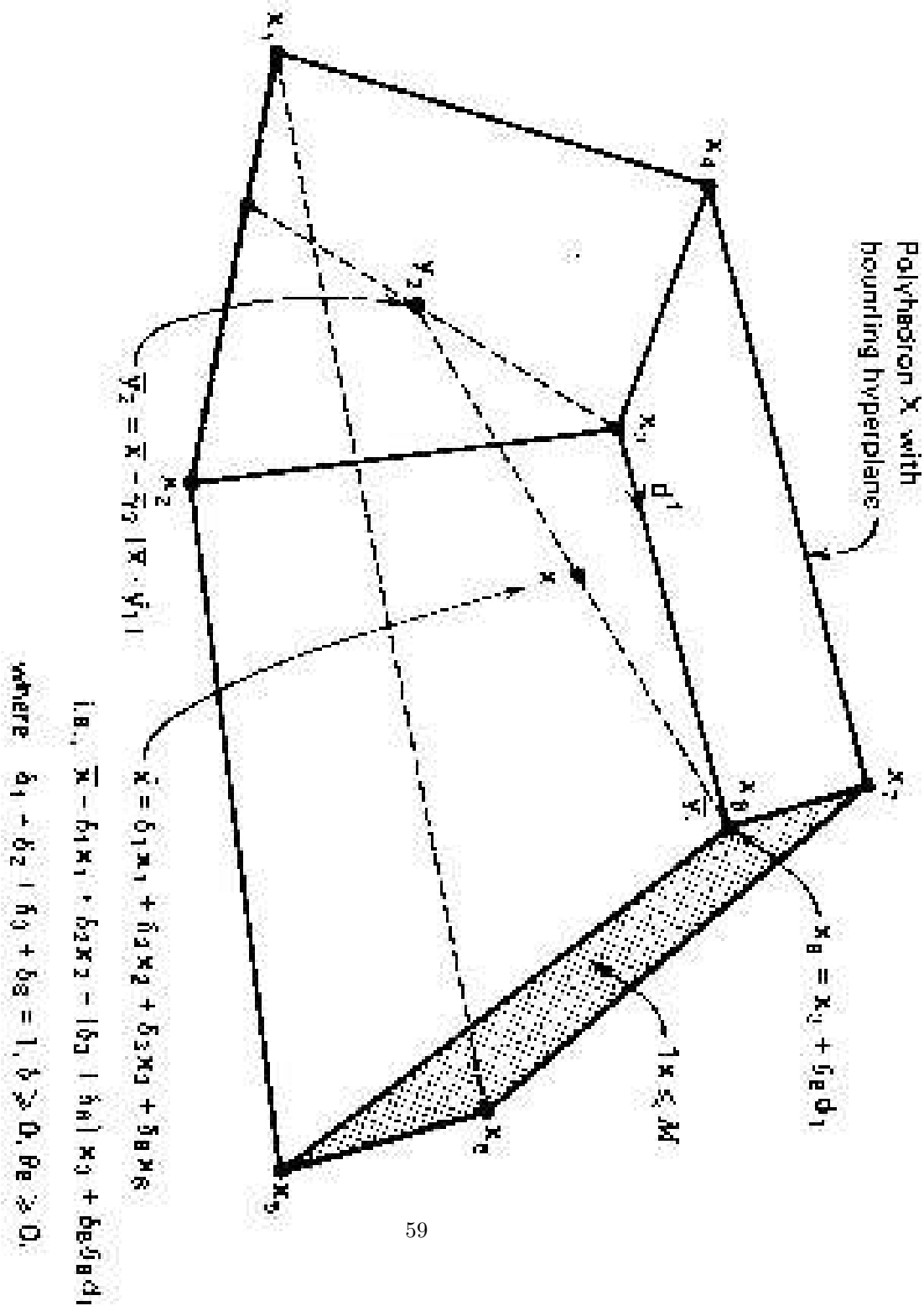


Figure 7: An illustration of the General Representation Theorem.

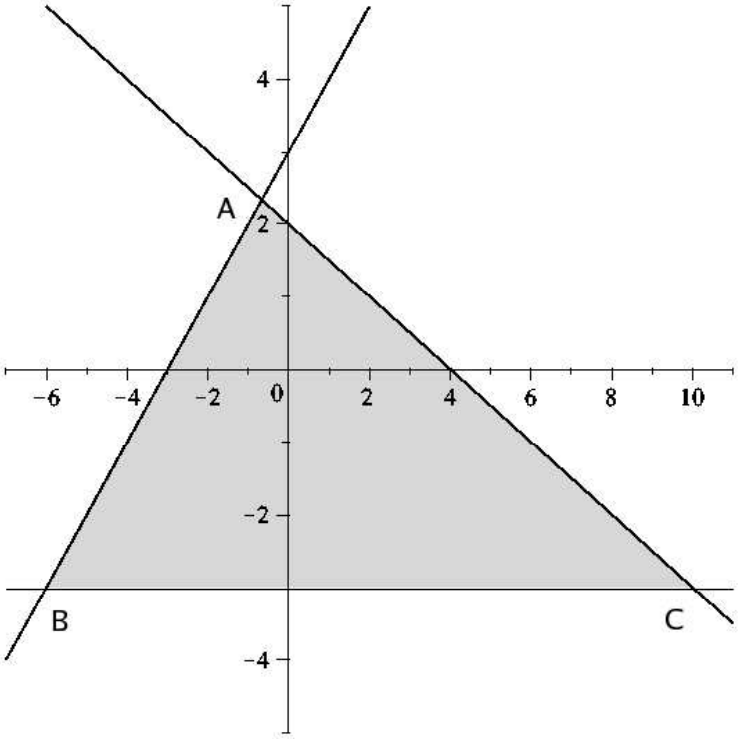


Figure 8: This figure shows the region from problem 2.

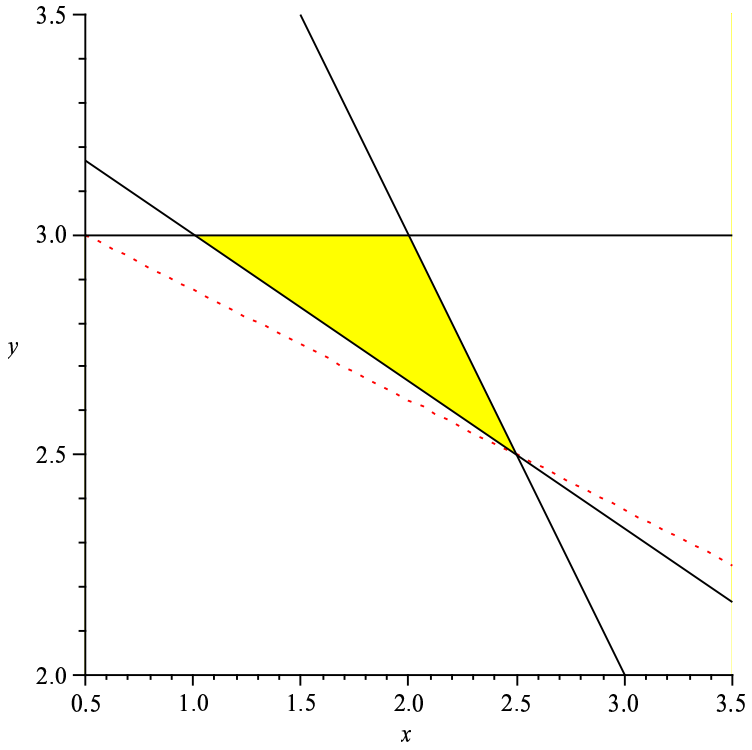


Figure 9: This figure is for the original Leary chemical problem. The shaded region is the feasible region: it satisfies all constraints on Leary Chemical's production.

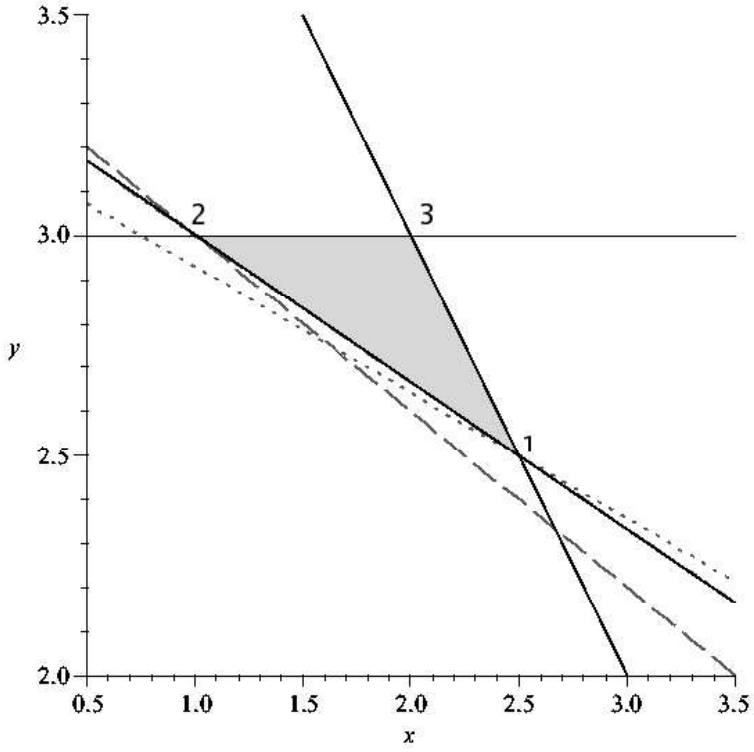


Figure 10: The shaded region is the feasible region. The dashed line is an isocost line for $c_1 = 2.5$ and $z = 8.5$ and the dotted line is the isocost line for $c_1 = 3.5$ and $z = 11.25$.

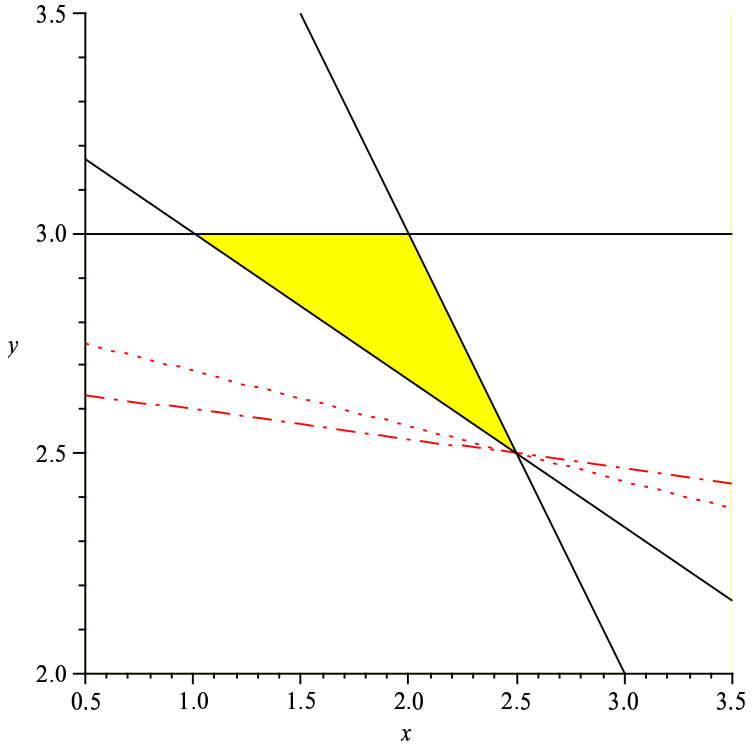


Figure 11: The shaded region is the feasible region. The dash-dotted line is an isocost line for $c_1 = 15$ and $z = 40$ and the dotted line is the isocost line for $c_1 = 8$ and $z = 22.5$.

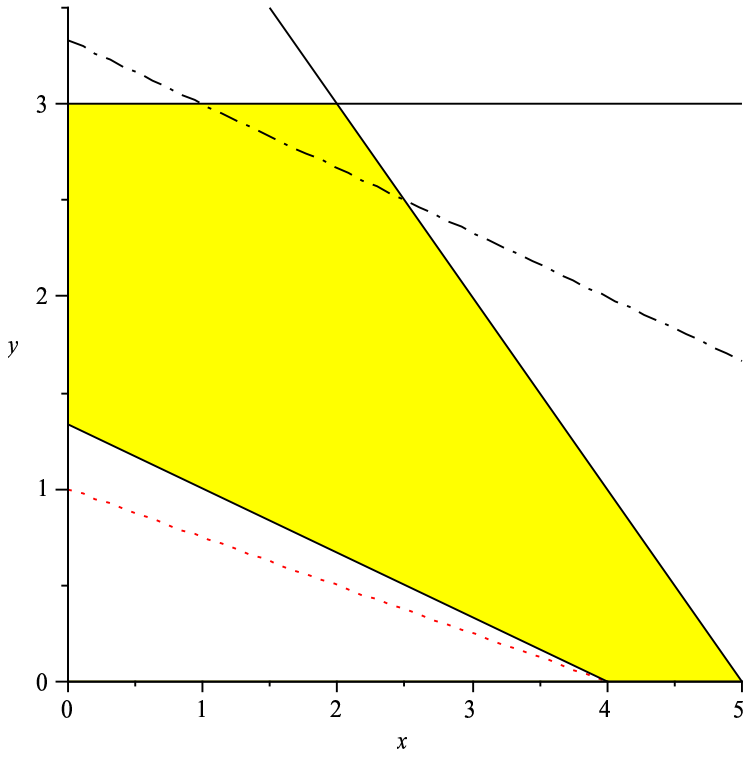


Figure 12: The shaded region is the feasible region with $b_1$ modified to equal 4. The dash-dotted line is the original constraint 1 for $b_1 = 10$. The dotted line is the isocost line for $z = 4$.

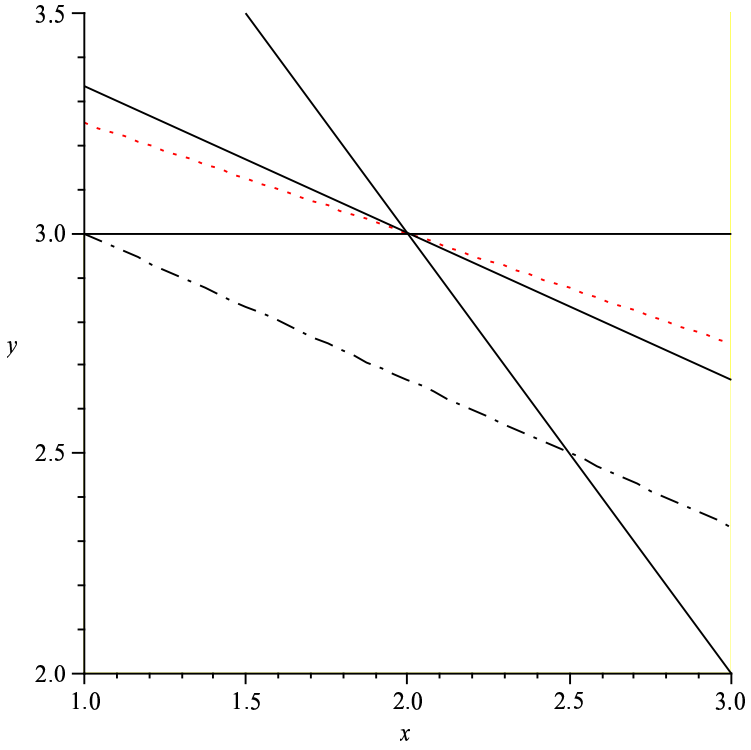


Figure 13: The shaded region is the feasible region with $b_1$ modified to equal 11. The dash-dotted line is the original constraint 1 for $b_1 = 10$. The dotted line is the isocost line for $z = 14$.