Chapter 3

Linear Programs

A linear program involves optimization (i.e., maximization or minimization) of a linear function subject to linear constraints. A linear inequality constraint on a vector $x \in \Re^N$ takes the form $a^T x \leq b$ or $a_1 x_1 + a_2 x_2 + \ldots + a_N x_N \leq b$ for some $a \in \Re^N$ and $b \in \Re$. If we have a collection of constraints $(a^1)^T x \leq b_1, (a^2)^T x \leq b_2, \ldots, (a^M)^T x \leq b_M$, we can group them together as a single vector inequality $Ax \leq b$ where

$$A = \left[\begin{array}{c} (a^1)^T \\ \vdots \\ (a^M)^T \end{array} \right] \quad \text{and} \quad b = \left[\begin{array}{c} b_1 \\ \vdots \\ b_M \end{array} \right].$$

When we write that the vector Ax is less than or equal to b, we mean that each component of Ax is less than or equal to the corresponding component of b. That is, for each i, $(Ax)_i \leq b_i$. Sometimes, in a slight abuse of language, we refer to the ith row A_{i*} of the matrix A as the ith constraint, and b_i as the value of the constraint.

In mathematical notation, a linear program can be expressed as follows:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b. \end{array}$$

The maximization is over $x \in \Re^N$. Each component x_j is referred to as a *decision variable*. The matrix $A \in \Re^{M \times N}$ and vector $b \in \Re^M$ specify a set of M *inequality constraints*, one for each row of A. The ith constraint comes from the ith row and is $(A_{i*})^T x \leq b_i$. The vector $c \in \Re^N$ is a vector of values for each decision variable. Each c_j represents the benefit of increasing x_j by 1. The set of vectors $x \in \Re^N$ that satisfy $Ax \leq b$ is called the feasible region. A linear program can also be defined to minimize the objective:

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \leq b, \end{array}$$

in which case c_j represents the cost of increasing x_j by 1.

3.1 Graphical Examples

To generate some understanding of linear programs, we will consider two simple examples. These examples each involve two decision variables. In most interesting applications of linear programming there will be many more decision variables – perhaps hundreds, thousands, or even hundreds of thousands. However, we start with cases involving only two variables because it is easy to illustrate what happens in a two dimensional space. The situation is analogous with our study of linear algebra. In that context, it was easy to generate some intuition through two-dimensional illustrations, and much of this intuition generalized to spaces of higher dimension.

3.1.1 Producing Cars and Trucks

Let us consider a simplified model of an automobile manufacturer that produces cars and trucks. Manufacturing is organized into four departments: sheet metal stamping, engine assembly, automobile assembly, and truck assembly. The capacity of each department is limited. The following table provides the percentages of each department's monthly capacity that would be consumed by constructing a thousand cars or a thousand trucks:

Department	Automobile	Truck
metal stamping	4%	2.86%
engine assembly	3%	6%
automobile assembly	4.44%	0%
truck assembly	0%	6.67%

The marketing department estimates a profit of \$3000 per car produced and \$2500 per truck produced. If the company decides only to produce cars, it could produce $22,500$ of them, generating a total profit of \$67.5 million. On the other hand, if it only produces trucks, it can produce $15,000$ of them, with a total profit of \$37.5 million. So should the company only produce cars? No. It turns out that profit can be increased if the company produces a combination of cars and trucks.

Let us formulate a linear program that will lead us to the optimal solution. Define decision variables x_1 and x_2 to be the number in thousands of cars and trucks, respectively, to produce each month. Together, they can be thought of as a vector $x \in \Re^2$. These quantities have to be positive, so we introduce a constraint $x \geq 0$. Several additional constraints arise from capacity limitations. The car assembly and truck assembly departments limit production according to

$$4.44x_1 \leq 100 \quad \text{and} \quad 6.67x_2 \leq 100.$$

The metal stamping and engine assembly activities also introduce constraints:

$$4x_1 + 2.86x_2 \leq 100 \quad \text{and} \quad 3x_1 + 6x_2 \leq 100.$$

The set of vectors $x \in \Re^2$ that satisfy these constraints is illustrated in Figure 3.1(a).

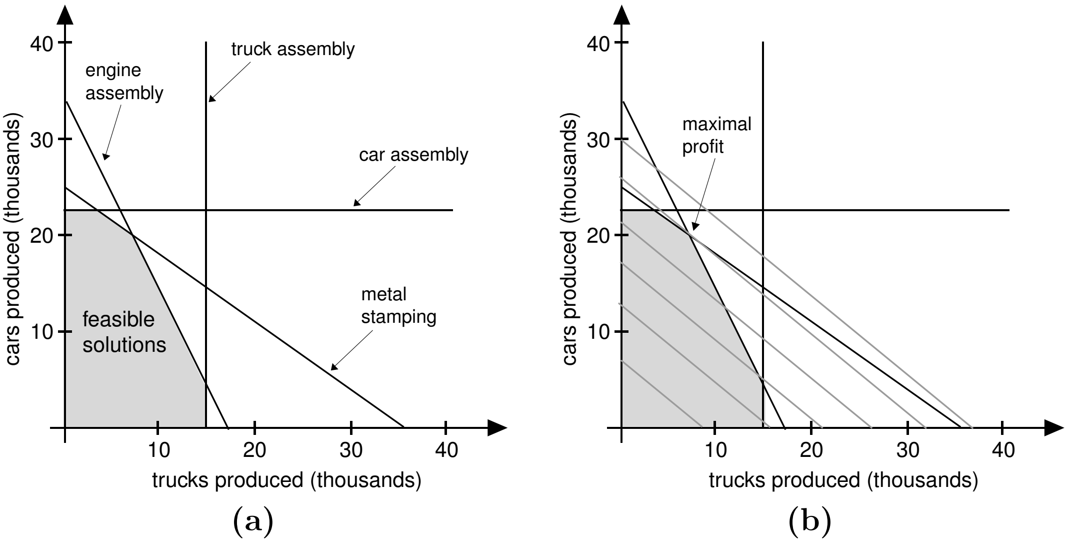

Figure 3.1: (a) Feasible solutions for production of cars and trucks. (b) Finding the solution that maximizes profit.

The anticipated profit in thousands of dollars associated with production quantities x_1 and x_2 is $3x_1 + 2.5x_2$. In Figure 3.1(b), each gray line superimposed on the set of solutions represents a subset for which the associated profit takes on a particular value. In other words, each line represents solutions of the equation $3x_1 + 2.5x_2 = \alpha$ for some value of α. The diagram also identifies the feasible solution that maximizes profit, which is given approximately by $x_1 = 20.4$ and $x_2 = 6.5$. Note that this solution involves making use of the entire capacity available for metal stamping and engine assembly, but does not maximize use of capacity to assemble either cars or trucks. The optimal profit is over \$77.3 million per month, which exceeds by about \$10 million the profit associated with producing only cars.

3.1.2 Feeding an Army

Suppose that two basic types of food are supplied to soldiers in an army: meats and potatoes. Each pound of meats costs \$1, while each pound of potatoes costs \$0.25. To minimize expenses, army officials consider serving only potatoes. However, there are some basic nutritional requirements that call for meats in a soldier's diet. In particular, each soldier should get at least 400 grams of carbohydrates, 40 grams of dietary fiber, and 200 grams of protein in their daily diet. Nutrients offered per pound of each of the two types of food, as well as the daily requirements, are provided in the following table:

Nutrient	Meats	Potatoes	Daily Requirement
carbohydrates	40 grams	200 grams	400 grams
dietary fiber	5 grams	40 grams	40 grams
protein	100 grams	20 grams	200 grams

Consider the problem of finding a minimal cost diet comprised of meats and potatoes that satisfies the nutritional requirements. Let x_1 and x_2 denote the number of pounds of meat and potatoes to be consumed daily. These quantities cannot be negative, so we have a constraint $x \geq 0$. The nutritional requirements impose further constraints:

$$\begin{array}{rcrll} 40x_1 & + & 200x_2 & \geq 400 & \text{(carbohydrates)} \\ 5x_1 & + & 40x_2 & \geq 40 & \text{(dietary fiber)} \\ 100x_1 & + & 20x_2 & \geq 200 & \text{(protein).} \end{array}$$

The set of feasible solutions is illustrated in Figure 3.2(a).

In Figure 3.2(b), superimposed lines identify sets that lead to particular daily costs. Unlike the automobile manufacturing problem we considered in the previous section, we are now *minimizing* cost rather than *maximizing* profit. The optimal solution involves a diet that includes both meats and potatoes, and is given approximately by $x_1 = 1.67$ and $x_2 = 1.67$. The associated daily cost per soldier is about \$2.08. Note that the constraint brought about by dietary fiber requirements does not affect the feasible region. This is because – based on our data – any serving of potatoes that offers sufficient carbohydrates will also offer sufficient dietary fibers.

3.1.3 Some Observations

There are some interesting observations that one can make from the preceding examples and generalize to more complex linear programs. In each case, the set of feasible solutions forms a polygon. By this we mean that the boundary of each is made up of a finite number of straight segments, forming corners

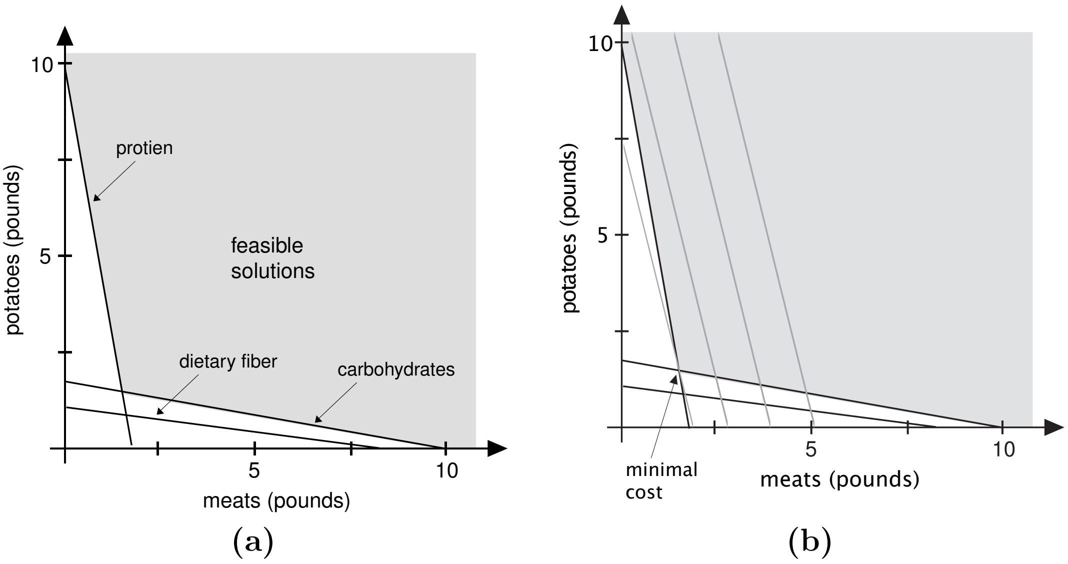

Figure 3.2: (a) Feasible solutions for an army diet. (b) Finding the solution that minimizes cost.

where they connect. In the case of producing cars and trucks, this polygon is bounded. In the case of feeding an army, the polygon is unbounded: two of the sides continue out to infinity. But in both cases they are polygons.

Another key commonality between the examples is that optimal solutions appear at corners of the polygon. To see why this is the case when we have two decision variables, consider the line given by $\{x \in \Re^2 | c^T x = \alpha\}$ for some $\alpha \in \Re$. As we change α we move the line continuously across $\Re^2$. To make α as large as possible, and hence maximize $c^T x$, we keep moving the line (increasing α) until any more movement will mean the line no longer intersects the feasible region. At this point, the line must be touching a corner.

In the next two sections, we will formalize and generalize these observations, so that we can make statements that apply to linear programs involving arbitrary numbers of variables and constraints. In higher dimensions, we will be dealing with polyhedra as opposed to polygons, and we will find that optimal solutions still arise at "corners."

3.2 Feasible Regions and Basic Feasible Solutions

The set of vectors $x \in \Re^N$ that satisfies constraints of the form $Ax \leq b$ is called a *polyhedron*. In three dimensions, the boundaries of the set are formed by "flat faces." In two dimensions, the boundaries are formed by line segments, and a polyhedron is a polygon.

Note that the feasible region of a linear program is a polyhedron. Hence, a linear program involves optimization of a linear objective function over a polyhedral feasible region. One way to view a polyhedron is as the intersection of a collection of half-spaces. A *half-space* is a set of points that satisfy a single inequality constraint. Hence, each constraint $(A_{i*})^T x \leq b_i$ defines a half-space, and the polyhedron characterized by $Ax \leq b$ is the intersection of M such half-spaces.

As an example, consider the problem of producing cars and trucks described in Section 3.1.1. Each constraint restricts feasible solutions to the half-space on one side of a line. For instance, the constraint that the number of cars produced must be nonnegative restricts the feasible region to vectors in the half-space on the right side of the horizontal axis in Figure 3.1(a). Note that, though this constraint was represented with a greater-than sign ($x_1 \geq 0$), it can also be represented with a less-than sign ($-x_1 \leq 0$) to be consistent with the form of $Ax \leq b$. The constraint introduced by the capacity to assemble engines also restricts solutions to a half-space – the set of points below a diagonal line. The intersection of half-spaces associated with the six constraints produces the polyhedron of feasible solutions.

In this section, we develop some understanding of the structure of polyhedra. We will later build on these ideas to establish useful properties of optimal solutions.

3.2.1 Convexity

Given two vectors x and y in $\Re^N$, a vector $z \in \Re^N$ is said to be a *convex combination* of x and y if there exists a scalar $\alpha \in [0,1]$ such that $z = \alpha x + (1-\alpha)y$. Intuitively, a convex combination of two vectors is in between the vectors, meaning it lies directly on the line segment joining the two vectors. In fact, the line segment connecting two vectors is the set of all convex combinations of the two vectors. We generalize this to more than two vectors by saying y is a convex combination of $x^1, x^2, \ldots, x^M$ if there are some $\alpha_1, \alpha_2, \ldots, \alpha_M \geq 0$ such that $y = \alpha_1 x^1 + \alpha_2 x^2 + \ldots + \alpha_M x^M$ and $\alpha_1 + \alpha_2 + \ldots + \alpha_M = 1$.

A set $U \subseteq \Re^N$ is said to be *convex* if any convex combination of any two elements of U is in U. In other words, if we take two arbitrary points in U, the line segment connecting those points should stay inside U. Figure 3.3 illustrates three subsets of $\Re^2$. The first is convex. The others are not.

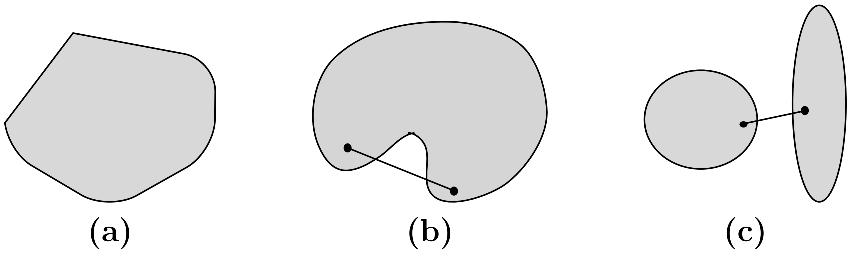

Figure 3.3: (a) A convex set in $\Re^2$. (b and c) Nonconvex sets in $\Re^2$ – in each case, a segment connecting points in the set includes points outside the set.

Given a set of vectors $x^1, \ldots, x^K \in \Re^N$, their *convex hull* is the smallest convex set containing $x^1, \ldots, x^K \in \Re^N$. Because any intersection of convex sets is convex, there is no ambiguity in this definition. In particular, the convex hull can be thought of as the intersection of all convex sets containing $x^1, \ldots, x^K \in \Re^N$.

Convex sets are very important in many areas of optimization, and linear programming is no exception. In particular, polyhedra are convex. To see why this is so, consider a polyhedron $U = \{x \in \Re^N | Ax \leq b\}$. If $z = \alpha x + (1-\alpha)y$ for some $\alpha \in [0,1]$ and $x, y \in U$ then

$$Az = A(\alpha x + (1-\alpha)y) = \alpha Ax + (1-\alpha)Ay \leq \alpha b + (1-\alpha)b = b$$

so that z is an element of U.

3.2.2 Vertices and Basic Solutions

Let $U \subseteq \Re^N$ be a polyhedron. We say $x \in U$ is a vertex of U if x is not a convex combination of two other points in U. Vertices are what we think of as "corners."

Suppose $U = \{x \in \Re^N | Ax \leq b\}$ is the feasible region for a linear program and that $y \in U$. If $(A_{i*})^T y = b_i$ then we say the ith constraint is *binding* or *active* at y. If we think of a polyhedron as a collection of half spaces, then for a constraint to be active, the point in question lies on the hyperplane forming the border of the half space. In three dimensions, it must lie on the face associated with the constraint, and in two dimensions, it must lie on the edge associated with the constraint. If a collection of constraints

$(a^1)^T x \leq \beta_1, \ldots, (a^K)^T x \leq \beta_K$ are active at a vector $\overline{x} \in \Re^N$, then $\overline{x}$ is a solution to $Bx = \beta$, where

$$B = \begin{bmatrix} (a^1)^T \\ \vdots \\ (a^K)^T \end{bmatrix} \quad \text{and} \quad \beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_K \end{bmatrix}.$$

A collection of linear constraints $(a^1)^T x \leq \beta_1, \ldots, (a^K)^T x \leq \beta_K$ is said to be linearly independent if $a^1, \ldots, a^K$ are linearly independent. For a given linear program, if there are N linearly independent constraints that are active at a vector $\overline{x} \in \Re^N$ then we say that $\overline{x}$ is a *basic solution*. To motivate this terminology, note that the active constraints form a basis for $\Re^N$. Note also that, because the active constraints form a basis, given N active constraints $(a^1)^T x \leq \beta_1, \ldots, (a^K)^T x \leq \beta_K$, the square matrix

$$B = \begin{bmatrix} (a^1)^T \\ \vdots \\ (a^K)^T \end{bmatrix},$$

has full rank and is therefore invertible. Hence, $Bx = \beta$ has a unique solution, which is a basic solution of the linear program.

If a basic solution $\overline{x} \in \Re^N$ is feasible, then we say $\overline{x}$ is a *basic feasible solution*. In two dimensions, a basic feasible solution is the intersection of two boundaries of the polygonal feasible region. Clearly, in two dimensions, basic feasible solutions and vertices are equivalent. The following theorem establishes that this remains true for polyhedra in higher-dimensional spaces.

Theorem 3.2.1. *Let U be the feasible region of a linear program. Then, $x \in U$ is a basic feasible solution if and only if x is a vertex of U.*

Proof: Suppose x is a basic feasible solution and also that x is a convex combination of y and z, both in U. Let C be a matrix whose rows are N linearly independent active constraints at x, and c be the vector of corresponding constraint values. Because C has linearly independent rows, it has full rank and is invertible. Also $Cy \leq c, Cz \leq c$ and $Cx = c$.

x is a convex combination of y and z so that $x = \alpha y + (1 - \alpha) z$ for some $\alpha \in [0, 1]$. This means $Cx = \alpha Cy + (1 - \alpha) Cz \leq \alpha c + (1 - \alpha) c = c$ can only equal c if at least one of Cy and Cz are equal to c. But because C is invertible, there is only one solution to $Cy = c$, namely $y = x$. Similarly $Cz = c$ gives $z = x$. This means x cannot be expressed as a convex combination of two points in U unless one of them is x, so that x is a vertex.

Conversely, suppose x is not a basic feasible solution. We let C be the matrix of all the active constraints at x. Because C has less than N linearly independent rows, it has a non-empty null space. Let d be a non zero vector in $\mathcal{N}(C)$. Then for small ϵ, we have that $x \pm \epsilon d$ is still feasible ($C(x \pm \epsilon d) = Cx = c$ and for small ϵ, non-active constraints will still be non-active). But $x = \frac{1}{2}(x + \epsilon d) + \frac{1}{2}(x - \epsilon d)$, so that x is a convex combination of two other vectors in U. So, x is not a vertex. □

If the feasible region of a linear program is $\{x \in \Re^N | Ax \leq b\}$ then any N linearly independent active constraints identify a unique basic solution $\overline{x} \in \Re^N$. To see why, consider a square matrix $B \in \Re^{N \times N}$ whose rows are N linearly independent active constraints. Any vector $x \in \Re^N$ for which these constraints are active must satisfy $Bx = b$. Since its rows of B are linearly independent, B has full rank and therefore a unique inverse B^{-1}. Hence, $B^{-1}b$ is the unique point at which the N constraints are active. Let us capture the concept in a theorem.

Theorem 3.2.2. *Given a polyhedron $\{x \in \Re^N | Ax \leq b\}$ for some $A \in \Re^{M \times N}$ and $b \in \Re^M$, any set of N linearly independent active constraints identifies a unique basic solution.*

Each basic solution corresponds to N selected constraints. There are M constraints to choose from and only finitely many ways to choose N from M. This implies the following theorem.

Theorem 3.2.3. *There are a finite number of basic solutions and a finite number of basic feasible solutions.*

Note that not every combination of N constraints corresponds to a basic solution. The constraints are required to be linearly independent.

3.2.3 Bounded Polyhedra

A polyhedron $U \subseteq \Re^N$ is said to be *bounded* if there is a scalar α such that, for each $x \in U$ and each j, $-\alpha \leq x_j \leq \alpha$. In other words, each component of a vector in U is restricted to a bounded interval, or U is contained in a "hyper-cube." The following theorem presents an alternative way to represent bounded polyhedra.

Theorem 3.2.4. *If U is a bounded polyhedron, it is the convex hull of its vertices.*

Proof: Let $U = \{x \in \Re^N | Ax \leq b\}$, and let H be the convex hull of the vertices of U. Each vertex of U is in U and U is convex. Hence, $H \subseteq U$.

We now have to show $U \subseteq H$. We will do this by showing that any $x \in U$ is a convex combination of the vertices of U, and hence is in H. We will do this by backwards induction on the number of linearly independent active constraints at x.

If the number of linearly independent active constraints is N, then x is a basic feasible solution, and so a vertex. A vertex is a convex combination of itself, and hence is in U. Thus all points with N linearly independent active constraints are in H.

Suppose all points with $K+1$ or more linearly independent active constraints are in H, and that x has K linearly independent active constraints. Let C be a matrix whose rows are the active constraints at x, and let c be the vector whose components are the corresponding constraint values. Because C has rank $K < N$, we know that its null space is non-empty. Take any non-zero vector $n \in \mathcal{N}(C)$ and consider the line $x + \alpha n$ for different α. For small α, the points on the line are inside U, but because U is bounded, for sufficiently large positive and negative values of α, the points on the line will not be in U. Take the most positive and negative values of α such that $x+\alpha n$ is in U, and let the corresponding points be y and z. Note $Cy = c = Cz$ so that all the constraints active for x are still active for y and z. However each one of them much also have an additional active constraint because n is in the null space of C and so changing α will not change $C(x + \alpha n)$. Thus each of y and z must have at least $K+1$ linearly independent active constraints. x lies on the line segment connecting y and z and so is a convex combination of them. By the inductive hypothesis, each of y and z are convex combinations of vertices, and hence so is x. Hence all points with K linearly independent active constraints are in H.

By induction, each $x \in U$ is also in H, so $U \subseteq H$. □

3.3 Optimality of Basic Feasible Solutions

Consider the linear program

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b. \end{array}$$

If $x^1, x^2, \ldots, x^K$ are the vertices of the feasible region, we say that x^k is an *optimal basic feasible solution* if $c^T x^k \geq c^T x^\ell$ for every ℓ. That is x^k is the vertex with the largest objective value among vertices. Note that the optimal basic feasible solution need not be unique.

An *optimal solution* is a feasible solution x such that $c^T x \geq c^T y$ for every other feasible solution y. As discussed in Section 3.1, in two dimensions, it is easy to see that an optimal basic feasible solution is also an optimal solution. In this section, we generalize this observation to polyhedra in higher-dimensional spaces.

3.3.1 Bounded Feasible Regions

We first consider linear programs where the feasible region is bounded.

Theorem 3.3.1. *If x^* is an optimal basic feasible solution of a linear program for which the feasible region is bounded, then it is an optimal solution.*

Proof: If x is a convex combination of y and z then $c^T x \leq \max\{c^T y, c^T z\}$. Similarly, if x is a convex combination of $x^1, \ldots, x^K$, then $c^T x \leq \max_{\ell \in \{1,\ldots,K\}} c^T x^\ell$. Any x is a convex combination of the vertices, and so $c^T x$ must attain its largest value at a vertex. □

3.3.2 The General Case

The result also applies to unbounded polyhedra, though the associated analysis becomes more complicated.

Theorem 3.3.2. *If x^* is an optimal basic feasible solution of a linear program that has an optimal solution, then it is an optimal solution.*

Before proving the theorem, we will establish a helpful lemma. Recall that a line is a one dimensional affine space. A set $U \subseteq \Re^N$ is said to *contain a line* if there exists a vector $x \in S$ and a vector $d \neq 0$ such that $x + \alpha d \in S$ for all $\alpha \in \Re$.

Lemma 3.3.1. *Consider a polyhedron $U = \{x \in \Re^N | Ax \leq b\}$ that is not empty. Then the following statements are equivalent:*

1. *The polyhedron U has at least one vertex.*

2. *The polyhedron U does not contain a line.*

Proof of lemma: Suppose that U contains the line $\{x + \alpha d | \alpha \in \Re\}$ where $d \neq 0$. Then for all α, we have $A(x + \alpha d) \leq b$, or rearranging $\alpha A d \leq b - Ax$. For this to be true for all α it must be that $Ad = 0$, so that A is not full rank, and so cannot have N linearly independent rows. Thus there are no vertices.

Suppose that U does not contain any line. We use a similar line of reasoning to Theorem 3.2.4. Let x be a point with the maximum number of linearly independent active constraints. Let the number of linearly independent active constraints be K. If $K = N$ then x is a vertex, and we are done.

Suppose that $K < N$, the consider the line $L = \{x + \alpha d | \alpha \in \Re\}$ for some d that is perpendicular to the constraints active at x. d must exist because the matrix whose rows are the constraints active at x is not full rank, and hence has a non-zero null space. All the points in L satisfy the K constraints at x. Because L cannot be contained in U, there must be some α for which an additional constraint is active. The point $x + \alpha d$ has $K + 1$ linearly independent active constraints contradicting the fact that K was the maximum attainable. □

Proof of theorem: Note that the fact that the linear program has a basic solution means that is can contain no lines. Let x be an optimal solution with the largest number of linearly independent active constraints, and the number of linearly independent active constraints at x be K.

If $K = N$ then x is a vertex satisfying the conclusion of the theorem. Suppose $K < N$, then take d orthogonal to the constraints active at x. The same reasoning as given in Theorem **??** shows that all points of the form $x + \alpha d$ have all the same constraints active as x, and also that for some α^*, an additional constraint is satisfied.

But, for sufficiently small α, $x \pm \alpha d$ is still feasible. Because $c^T(x + \alpha d) = c^T x + \alpha c^T d$ can be no larger than $c^T x$, we have that $c^T d = 0$ and that $c^T(x + \alpha d) = c^T x$ for all α. But this means $x + \alpha^* d$ is an optimal solution with more than K linearly independent active constraints, contradicting the maximality of K. □

3.3.3 Searching through Basic Feasible Solutions

For any linear program, there are a finite number of basic feasible solutions, and one of them is optimal if the linear program has an optimum. This motivates a procedure for solving linear programs: enumerate all basic feasible solutions and select the one with the largest objective value. Unfortunately, such a procedure is not effective for large problems that arise in practice because there are usually far too many basic feasible solutions. As mentioned earlier, the number of basic solutions is the number of ways of choosing N linearly independent constraints from the entire collection of M constraints. There are $M!/(N!(M-N)!)$ choices. This is an enormous number – if $N = 20$ and $M = 100$, the number of choices $M!/(N!(M-N)!)$ exceeds half a billion trillion. Though many of these choices will not be linearly independent or feasible, the number of them that are basic feasible solutions is usually still enormous.

In Chapter **??**, we will study the simplex method, which is a popular linear programming algorithm that searches through basic feasible solutions for an optimal one. It does not enumerate all possibilities but instead intelligently traverses a sequence of improving basic feasible solutions in a way that arrives quickly at an optimum. We will also study interior point methods, another very efficient approach that employs a different strategy. Instead of considering basic feasible solutions, interior point methods generate a sequence of improving solutions in the interior of the feasible region, converging on an optimum.

3.4 Greater-Than and Equality Constraints

We have focused until now on less-than constraints, each taking the form $a^T x \leq b$ for some $a \in \Re^N$ and $b \in \Re$. Two other forms of constraints commonly used to describe polyhedra are greater-than and equality constraints. Greater-than constraints take the form $a^T x \geq b$, while equality constraints take the form $a^T x = b$.

Both greater-than and equality constraints can be replaced by equivalent less-than constraints. A greater than constraint $a^T x \geq b$ is equivalent to $(-a)^T x \leq -b$, whereas an equality constraint $a^T x = b$ is equivalent to a pair of less-than constraints: $a^T x \leq b$ and $(-a)^T x \geq -b$. Hence, any set of constraints that includes less-than, greater-than, and equality constraints is equivalent to a set of less-than constraints and therefore represents a polyhedron.

In matrix notation, we can define a polyhedron involving all types of constraints by

$$S = \{x \in \Re^N | A^1 x \leq b, A^2 x \geq b^2, A^3 x = b^3\}.$$

This is the same polyhedron as one characterized by $Ax \leq b$, where

$$A = \begin{bmatrix} A^1 \\ -A^2 \\ A^3 \\ -A^3 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} b^1 \\ -b^2 \\ b^3 \\ -b^3 \end{bmatrix}.$$

The vertices of a polyhedron do not change if we change the way it is represented. This is because the notion of a vertex is geometric; that is, it only depends on which vectors are inside or outside the set. Our definition of *basic feasible solutions*, on the other hand, does rely on the algebraic representation. In particular, our definition determines whether a solution is basic depending on which of the M constraints represented by $Ax \leq b$ are active.

Theorem 3.2.1 establishes that vertices and basic feasible solutions are equivalent. Hence, the theorem relates a geometric, representation-independent concept to an algebraic, representation-dependent one. It is convenient to extend the definition of a basic feasible solution to situations where the representation of a polyhedron makes use of greater-than and equality constraints. This extended definition should maintain the equivalence between basic feasible solutions and vertices.

Let us now provide the generalized definition of basic and basic feasible solutions. Given a set of equality and inequality constraints defining a polyhedron $S \in \Re^N$, we say that a vector $\overline{x} \in \Re^N$ is a *basic solution* if all equality constraints are active, and among all constraints that are active at $\overline{x}$, N of them are linearly independent. A *basic feasible solution* is a basic solution that satisfies all constraints.

To see that basic feasible solutions still correspond to vertices, consider a polyhedron

$$S = \{x \in \Re^N | A^1x \leq b^1, A^2x \geq b^2, A^3x = b^3\}.$$

Recall that $S = \{x \in \Re^N | Ax \leq b\}$, where

$$A = \begin{bmatrix} A^1 \\ -A^2 \\ A^3 \\ -A^3 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} b^1 \\ -b^2 \\ b^3 \\ -b^3 \end{bmatrix}.$$

From Theorem 3.2.1, we know that basic feasible solutions of $Ax \leq b$ are equivalent to vertices of S. We will show that basic feasible solutions of the inequalities $A^1x \leq b, A^2x \geq b^2, A^3x = b^3$ are equivalent to basic feasible solutions of $Ax \leq b$.

Consider a basic feasible solution $\overline{x}$ of $A^1x \leq b^1, A^2x \geq b^2, A^3x = b^3$. All the equality constraints must be active at $\overline{x}$, and there must be a set of N linearly independent constraints that are active at $\overline{x}$. Let $\mathcal{I}$ denote this set of N linearly independent constraints. Let N_1, N_2, and N_3, be the number of less-than, greater-than, and equality constraints represented by $\mathcal{I}$. Hence, $N_1 + N_2 + N_3 = N$. Now consider the inequality $Ax \leq b$, which we write in a partitioned form

$$\begin{bmatrix} A^1 \\ -A^2 \\ A^3 \\ -A^3 \end{bmatrix} x \leq \begin{bmatrix} b^1 \\ -b^2 \\ b^3 \\ -b^3 \end{bmatrix}.$$

Since $A^3\overline{x} = b^3$, all rows of A associated with A^3 and $-A^3$ correspond to active constraints. Only $2N_3$ of these correspond to equality constraints in $\mathcal{I}$, and among these $2N_3$ constraints, only only N_3 of them can be linearly independent (since each row of $-A_3$ is linearly dependent on the corresponding row of A_3).

Another $N_1 + N_2$ constraints in $\mathcal{I}$ lead to linearly independent active constraints in rows of A associated with A^1 and $-A^2$. This makes for a total of N linearly independent constraints. Therefore, any basic feasible solution of $A^1x \leq b^1, A^2x \geq b^2, A^3x = b^3$ is also a basic feasible solution of $Ax = b$ and therefore a vertex of S. The converse – that a basic feasible solution of $Ax = b$ is a basic feasible solution of $A^1x \leq b^1, A^2x \geq b^2, A^3x = b^3$ – can be shown by reversing preceding steps. It follows that basic feasible solutions of $A^1x \leq b^1, A^2x \geq b^2, A^3x = b^3$ are equivalent to vertices of S.

3.5 Production

We now shift gears to explore a few of the many application domains of linear programming. A prime application of linear programming is to the allocation of limited resources among production activities that can be carried out at a firm. In this context, linear programming is used to determine the degree to which the firm should carry out each activity, in the face of resource constraints. In this section, we discuss several types of production problems that can be modeled and solved as linear programs.

3.5.1 Single-Stage Production

In a single stage production problem, there is stock in M types of resources and N activities, each of which transforms resources into a type of product. The available stock in each ith resource is denoted by b_i, which is a component of a vector $b \in \Re^M$. The level to which activity j is carried out is a decision variable x_j, which a component of a vector $x \in \Re^N$. This quantity x_j represents the number of units of product type j generated by the activity.

In producing each unit of product j, A_{ij} units of each ith resource are consumed. This gives us a matrix $A \in \Re^{M \times N}$. The activity levels are constrained to be nonnegative ($x_j \geq 0$), and in aggregate, they cannot consume more resources than available ($Ax \leq b$). Each unit of product j generates a profit of c_j, which is a component of a vector $c \in \Re^N$. The objective is to maximize profit. This gives rise to a linear program:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

Let us revisit the petroleum production problem of Chapter 1, which is an example of a single-stage production problem.

Example 3.5.1. *Crude petroleum extracted from a well contains a complex mixture of component hydrocarbons, each with a different boiling point. A refinery separates these component hydrocarbons using a distillation column. The resulting components are then used to manufacture consumer products such as low, medium, and high octane gasoline, diesel fuel, aviation fuel, and heating oil.*

Suppose we are managing a company that manufactures N petroleum products and have to decide on the number of liters x_j, $j \in \{1, \ldots, n\}$ of each product to manufacture next month. We have M types of resources in the form of component hydrocarbons. A vector $b \in \Re^M$ represents the quantity in liters of each ith resource to be available to us next month. Each petroleum product is manufactured through a separate activity. The jth activity consumes A_{ij} liters of the ith resource per unit of the jth product manufactured.

Our objective is to maximize next month's profit. Each jth product garners c_j dollars per liter. Hence, the activity levels that maximize profit solve the following linear program:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

There are typically many possible activities that can be carried out to produce a wide variety of products. In this event, the number of activities N may far exceed the number M of resource types. Is it advantageous to carry out so many activities in parallel? Remarkably, the following theorem establishes that it is not:

Theorem 3.5.1. *If a single-stage production problem with M resource types and N activities has an optimal solution, then there is an optimal solution that involves use of no more than M activities.*

This result follows from Theorem 3.3.2, as we will now explain. Consider a basic feasible solution $\overline{x} \in \Re^N$. At $\overline{x}$, there must be N linearly independent binding constraints. Up to M of these constraints can be associated with the M rows of A. If $N > M$, we need at least $N - M$ additional binding constraints – these must be nonnegativity constraints. It follows that at least $N - M$ components of $\overline{x}$ are equal to zero. In other words, there are at most M activities that are used at a basic feasible solution.

By Theorem 3.3.2, if there is an optimal solution to a linear program, there is one that is a basic feasible solution. Hence, there is an optimal solution that entails carrying out no more than M activities. This greatly simplifies implementation of an optimal production strategy. No matter how many activities are available, we will make use of no more than M of them.

Before closing this section, it is worth discussing how to deal with capacity constraints. In particular, production activities are often constrained not only by availability of raw materials, but also by the capacity of manufacturing facilities. In fact, in many practical production activities, capacity is the only relevant constraint. Conveniently, capacity can simply be treated as an additional resource consumed by manufacturing activities. We illustrate this point through a continuation to our petroleum production example.

Example 3.5.2. *Recall that our petroleum production problem led to a linear program*

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

Given an optimal solution $x^* \in \Re^N$, each component x_j^* tells us the quantity in liters of the jth petroleum product that we should manufacture next month.

Suppose that we have two factories that support different manufacturing processes. Each of our N manufacturing activities requires capacity from one or both of the factories. In particular, the manufacturing of each liter of the jth product requires a fraction a_j^1 of next month's capacity from factory 1 and a fraction a_j^2 of next month's capacity from factory 2. Hence, we face capacity constraints:

$$(a^1)^T x \leq 1 \quad \text{and} \quad (a^2)^T x \leq 1.$$

Let

$$\overline{A} = \left[\begin{array}{c} A \\ (a^1)^T \\ (a^2)^T \end{array} \right] \quad \text{and} \quad \overline{b} = \left[\begin{array}{c} b \\ 1 \\ 1 \end{array} \right].$$

A new linear program incorporates capacity constraints:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & \overline{A}x \leq b \\ & x \geq 0. \end{array}$$

Note that the capacity constraints play a role entirely analogous to the constraints imposed by limitations in stock of component hydrocarbons. The capacity at each factory is just a resource that leads to an additional constraint.

3.5.2 Multi-Stage Production

Manufacturing of sophisticated products typically entails multiple stages of production activity. For example, in manufacturing a computer, chips are fabricated, then they are connected on a printed circuit board, and finally, printed circuit boards, casing, and other components are assembled to create a finished product. In such a process, not all activities deliver finished products. Some activities generate materials that serve as resources for other activities. Multi-stage production activities of this sort can still be formulated as linear programs. Let us illustrate this with an example.

Example 3.5.3. *Consider a computer manufacturer with two CPU chip fabrication facilities and one computer assembly plant. Components such as keyboards, monitors, casing, mice, disk drives, and other chips such as SRAM and DRAM are purchased from other companies. There are three grades of CPU chips manufactured by the company, and they are used to produce three models of computers. Fabrication facility 1 can produce chips of grades 1 and 2, while fabrication facility 2 can produce chips of grade 2 and 3. Completed chips are transported to the assembly plant where they are combined with other components to produce finished products. The only relevant constraints on manufacturing are capacities of the two fabrication facilities and the manufacturing plant.*

Consider the decision of how to allocate resources and conduct manufacturing activities over the next month of operation. To formulate the problem as a linear program, we introduce the following decision variables:

x_1	*quantity of model 1 produced*
x_2	*quantity of model 2 produced*
x_3	*quantity of model 3 produced*
x_4	*quantity of grade 1 chip produced at factory 1*
x_5	*quantity of grade 2 chip produced at factory 1*
x_6	*quantity of grade 2 chip produced at factory 2*
x_7	*quantity of grade 3 chip produced at factory 2*

There are 6 *types of resources to consider:*
(1) *capacity at fabrication facility 1;*
(2) *capacity at fabrication facility 2;*

(3) *capacity at the assembly plant;*
(4) *grade 1 chips;*
(5) *grade 2 chips;*
(6) *grade 3 chips.*
Let the quantities available next month be denoted by a vector $b \in \Re^6$.

For each jth activity and ith resource, let A_{ij} denote the amount of the ith resource consumed or produced per unit of activity j. If $A_{ij} > 0$, this represents an amount consumed. If $A_{ij} < 0$, this represents an amount produced. Consider, as an example, the constraint associated with capacity at the assembly plant:

$$A_{31}x_1 + A_{32}x_2 + A_{33}x_3 + A_{34}x_4 + A_{35}x_5 + A_{36}x_6 + A_{37}x_7 \leq b_3.$$

The coefficients A_{34}, A_{35}, A_{36}, and A_{37} are all equal to zero, since capacity at the assembly plant is not used in manufacturing chips. On the other hand, producing the three models of computers does require assembly, so A_{31}, A_{32}, and A_{33} are positive. Let us now consider, as a second example, the constraint associated with stock in grade 2 chips, which is an endogenously produced resource:

$$A_{51}x_1 + A_{52}x_2 + A_{53}x_3 + A_{54}x_4 + A_{55}x_5 + A_{56}x_6 + A_{57}x_7 \leq b_5.$$

The coefficients A_{51}, A_{52}, and A_{53} are zero or positive, depending on how many grade 2 chips are used in manufacturing units of each model. The coefficients A_{54} and A_{57} are zero, because manufacturing of grade 1 and grade 3 chips does not affect our stock in grade 2 chips. Finally, $A_{55} = -1$ and $A_{56} = -1$, because each unit of activities 5 and 6 involves production of one grade 2 chip. The value of b_5 represents the number of chips available in the absence of any manufacturing. These could be chips acquired from an exogenous source or left over from the previous month. If $b_5 = 0$, all chips used to produce computers next month must be manufactured during the month.

The profit per unit associated with each of the three models is given by a vector $c \in \Re^7$, for which only the first three components are nonzero. With an objective of maximizing profit, we have a linear program:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

Note that the basic difference between linear programs arising from multistage – as opposed to single-stage – production problems is that elements of the matrix A are no longer nonnegative. Negative elements are associated with the production of resources. There is another way to represent such linear programs that is worth considering. Instead of having a matrix A that can have negative elements, we could define matrices C and P such that both have only nonnegative elements and $A = C - P$. The matrix C represents quantities consumed by various activities, while P represents quantities produced. Then, the multi-stage production problem takes the form

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Cx - Px \leq b \\ & x \geq 0. \end{array}$$

As in the single-stage case, a basic feasible solution of the multi-stage production problem with M resource types – including materials produced for use as resources to later stages of production – must have M linearly independent binding constraints. If the number N of activities exceeds the number M of resources, at least $N - M$ activities must be inactive at a basic feasible solution. We therefore have an extension of Theorem 3.5.1:

Theorem 3.5.2. *If a multi-stage production problem with M resource types and N activities has an optimal solution, then there is an optimal solution that involves use of no more than M activities.*

3.5.3 Market Stratification and Price Discrimination

In all the production models we have considered, each unit of a product generated the same profit. Consider, for example, a single-stage production problem:

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

Each unit of each jth product offers a profit of c_j.

In some situations, it is desirable to stratify a market and charge different prices for different classes of customers. For example, coupons can be directed at a certain segment of the market that will only purchase a product if it is below the advertised price. This allows a firm to sell at a high price to those who find it acceptable without loosing the profit it can obtain from the portion of the market that requires a lower price.

When there is a single price c_j associated with each jth product, the profit generated by manufacturing x_j units is $c_j x_j$. Suppose that the market is segmented and price discrimination is viable. In particular, suppose that we can sell up to K units each jth product at price c_j^1 and the rest at price $c_j^2 < c_j^1$. Then, the objective should be to maximize $\sum_{j=1}^N f_j(x_j)$, where

$$f_j(x_j) = \begin{cases} c_j^1 x_j, & \text{if } x_j \leq K \\ c_j^1 K + c_j^2(x_j - K), & \text{otherwise.} \end{cases}$$

In fact, suppose that this were the case for every product. Then, the single-stage production problem becomes:

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^N f_j(x_j) \\ \text{subject to} & Ax \leq b \\ & x \geq 0. \end{array}$$

This optimization problem is not a linear program, but fortunately, it can be converted to one, as we now explain.

We introduce new decision variables $x^1, x^2 \in \Re^N$. For each j, let $x_j^1 = \min(x_j, K)$ and $x_j^2 = \min(x_j - K, 0)$. Hence, $x_j = x_j^1 + x_j^2$. Each x_j^1 represents the quantity of product j manufactured and sold for profit c_j^1, while x_j^2 represents the quantity manufactured and sold for profit c_j^2. A linear program leads to optimal values for these new decision variables:

$$\begin{array}{ll} \text{maximize} & (c^1)^T x^1 + (c^2)^T x^2 \\ \text{subject to} & A(x^1 + x^2) \leq b \\ & x^1 \leq Ke \\ & x^1 \geq 0 \\ & x^2 \geq 0. \end{array}$$

Recall that e denotes the vector with every component equal to 1.

This idea generalizes to any number of market segments. If there are L different profit vectors $c^1 \geq c^2 \geq \cdots \geq c^L$, let $K_1, \ldots, K_{L-1}$ denote numbers of customers that will purchase each jth product at prices $c_j^1, \ldots, c_j^{L-1}$, respectively. Then, the profit-maximizing linear program is given by

$$\begin{array}{lll} \text{maximize} & \sum_{k=1}^L (c^k)^T x^k & \\ \text{subject to} & A\left(\sum_{k=1}^L x^k\right) \leq b & \\ & x^k \leq K_k e, & \text{for } k \in \{1, \ldots, L-1\} \\ & x^k \geq 0, & \text{for } k \in \{1, \ldots, L\}. \end{array}$$

There are LN decision variables in the above linear program, so at a basic feasible solution $\overline{x}^1, \ldots, \overline{x}^L$, there must be LN linearly independent active constraints. Among these, at most M can be associated with resource constraints, and at most $(L-1)N$ can be associated with the constraints

$x^k \leq K_k e$. If $N > M$, this leaves at least $N - M$ additional constraints that must be active. These must be nonnegativity constraints. It follows that at least $N - M$ components among $\overline{x}^1, \ldots, \overline{x}^L$ are equal to zero. We therefore have the following theorem.

Theorem 3.5.3. *If a production problem with M resource types, N activities, and multiple market segments associated with different profits has an optimal solution, then there is an optimal solution that involves use of no more than M activities.*

3.6 Contingent Claims

In Section 2.4, we introduced the study of contingent claims. We showed how structured products can sometimes be replicated and priced and discussed the notion of arbitrage. In this section, we revisit the topic of contingent claims, bringing to bear our understanding of linear programming. This will allow us to broaden the range of situations where a bank can sell structured products while protecting itself against risk and to identify arbitrage opportunities, when they exist.

3.6.1 Structured Products in an Incomplete Market

As in Section 2.4, consider a collection of N assets in a world with M possible outcomes. The possible payoffs of each jth asset is represented by a vector $a^j \in \Re^M$. A payoff matrix

$$P = \left[\begin{array}{ccc} a^1 & \ldots & a^N \end{array} \right],$$

represents payoff vectors for all assets. The price per unit of each asset is given by the corresponding component of a vector $\rho \in \Re^N$.

When a bank sells a structured product, it is desirable to protect against risks. This can be accomplished by finding portfolio holdings $x \in \Re^N$ that replicate the product, and purchasing the associated quantities of assets. The process of replication also guides pricing of the structured product. In particular, the price that the bank charges should exceed the cost $\rho^T x$ of the replicating portfolio.

It is not always possible to find a replicating portfolio. A structured product with a payoff vector $b \in \Re^M$ can only be replicated if there is a vector $x \in \Re^N$ of portfolio holdings such that $Px = b$. Such a vector exists only when b is in $\mathcal{C}(P)$. This is true for all b only if the market is complete

– that is, if P has rank M. Otherwise, the market is said to be *incomplete*, and some structured products cannot be replicated.

How can a bank protect against risks when it sells a structured product that cannot be replicated? One way involves *super-replicating* the structured product. A portfolio with holdings x is said to *super-replicate* a structured product with payoff function b if $Px \geq b$. If a bank sells a structured product and purchases a super-replicating portfolio, its net cash flow $(Px)_i - b_i$ for any possible future outcome i is nonnegative. The price $\rho^T x$ of this super-replicating portfolio can also be used as a lower bound for the price to charge the customer for the structured product. We illustrate the concept of super-replication with an example.

Example 3.6.1. (Super-Replicating a Currency Hedge) *Recall the structured product described in Example 2.4.2. Suppose that the current value of the foreign currency is* $r^0 = 0.5$ *dollars, that* $r^* = 0.1$ *dollars and* $p = 10$ *million dollars, and that the currency value one year from now will be in the set* $\{0.01, 0.02, \ldots, 0.99, 1.0\}$. *Hence, the payoff vector* b *for the structured product is in* $\Re^{100}$, *with* b_i *being the value of the product one year from now in the event that the currency is valued at* i *cents. This payoff vector is illustrated in Figure 3.4.*

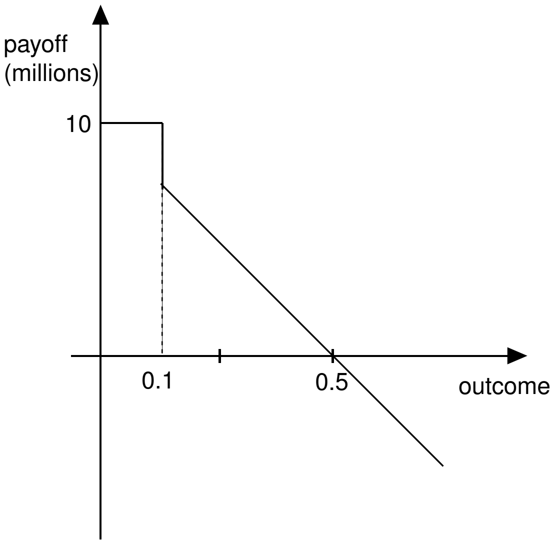

Figure 3.4: The payoff function for a protective structured product.

Consider a situation where a bank sells this structured product and wishes to protect against associated risks. Suppose there are several assets in the market that can be traded for this purpose:

(a) The currency. *If a unit of foreign currency is sold one year from now, the payoff vector* $a^1 \in \Re^{100}$ *is given by* $a^1_i = i/100$ *for each* i.
(b) A zero-coupon bond. *The payoff vector* $a^2 \in \Re^{100}$ *is given by* $a^2_i = 1$ *for each* i.
(c) A European call with strike 0.1**.** *The payoff vector* $a^3 \in \Re^{100}$ *is given by* $a^3_i = \max(a^1_i - 0.1, 0)$ *for each* i.
(d) A European call with strike 0.2**.** *The payoff vector* $a^4 \in \Re^{100}$ *is given by* $a^3_i = \max(a^1_i - 0.2, 0)$ *for each* i.
Payoff functions for these four assets are illustrated in Figure 3.5. The structured product payoff vector b *is not in the span of* $a^1, \ldots, a^4$*, and therefore, the product cannot be replicated.*

There are many ways to super-replicate the structured product. One involves purchasing 10 *million units of the bond. This leads to a portfolio that pays* 10 *million dollars in any outcome. This amount always exceeds what the bank will have to pay the customer. Somehow this feels like overkill, since in many outcomes, the value of the bond portfolio will far exceed the value of the structured product.*

An alternative replicating portfolio is constructed by purchasing 10 *million units of the bond and* 20 *million call options at a strike of* 0.2*, and short selling* 40 *million call options at a strike of* 0.1*. The payoff vector of this super-replicating portfolio, as well as the one consisting only of bonds, is illustrated in Figure 3.6. Note that the payoffs associated with the second super-replicated portfolio are dominated by those offered by the first. It is natural to expect that the second super-replicating portfolio should offer a less expensive way of protecting against risks brought about by selling the structured product.*

Given multiple super-replicating portfolios, as in the above example, which one should the bank purchase? The cheapest one, of course! The price of a replicating portfolio with holdings x is $\rho^T x$. To find the cheapest replicating portfolio, the bank can solve a linear program:

$$\begin{array}{ll} \text{minimize} & \rho^T x \\ \text{subject to} & Px \geq b. \end{array}$$

The constraints ensure that the resulting portfolio holdings do indeed generate a super-replicating portfolio. We illustrate the process in the context of our currency hedge.

Example 3.6.2. (Cheapest Super-Replicating Portfolio) *Suppose that the prices per unit of the currency, bond, and European call options are* \$0.5*,* \$0.9*,* \$0.4*, and* \$0.35*, respectively. Then, letting* $\rho = [0.5\ 0.9\ 0.4\ 0.35]^T$*,* *solving the linear program*

$$\begin{array}{ll} \text{minimize} & \rho^T x \\ \text{subject to} & Px \geq b, \end{array}$$

leads to portfolio holdings of $x = 10^6 \times [0\ 10\ \ -40\ 20]^T$*. This corresponds to the second super-replicating portfolio considered in Example 3.6.1.*

By Theorem 3.3.2, if the linear program

$$\begin{array}{ll} \text{minimize} & \rho^T x \\ \text{subject to} & Px \geq b. \end{array}$$

has an optimal solution and there is a basic feasible solution, then there is a basic feasible solution that is an optimal solution. Let x^* be an optimal basic feasible solution. Then, there must be at least N constraints active at x^*. In other words, the payoff vector Px^* is equal to the payoff vector b of the structured product for at least N outcomes. This means that if we plot the payoff vectors associated with the structured product and this optimal super-replicating portfolio on the same graph, they will touch at no less than N points.

3.6.2 Finding Arbitrage Opportunities

In Section 2.4, we introduced the notion of arbitrage. An arbitrage opportunity was defined to be a vector $x \in \Re^N$ of portfolio holdings with a negative cost $\rho^T x < 0$ and nonnegative payoffs $Px \geq 0$. By purchasing assets in quantities given by the portfolio weights, we receive an amount $-\rho^T x > 0$, and in every possible future event, we are not committed to pay any money. Sounds good. But why stop at buying the quantities identified by x? Why not buy quantities $100x$ and garner an income of $-100\rho^T x$? Indeed, an arbitrage opportunity offers the possibility of making unbounded sums of money.

We have been talking about arbitrage opportunities for some time, but now that we understand linear programming, we can actually identify arbitrage opportunities, if they exist. Consider solving the following linear program:

$$\begin{array}{ll} \text{minimize} & \rho^T x \\ \text{subject to} & Px \geq 0. \end{array}$$

It is clear that this linear program will look for the most profitable arbitrage opportunity. However, because there is no bound to the sum of money that can be made when an arbitrage opportunity presents itself, this linear program will have an unbounded solution when an arbitrage opportunity exists.

In order to generate a more meaningful solution, consider an alternative linear program:

$$\begin{array}{ll} \text{minimize} & \rho^T x \\ \text{subject to} & Px \geq 0 \\ & \rho^T x = -1. \end{array}$$

This one identifies a portfolio that offers an initial income of \$1. If an arbitrage opportunity exists, we can use is to make any amount of money, so we could use it to generate a \$1 income. Hence, existence of an arbitrage opportunity implies that the feasible region of this linear program is nonempty. Once this linear program identifies a portfolio that generates \$1, we can make an arbitrarily large amount of money by purchasing multiples of this portfolio. Note that the objective function here could be anything – any feasible solution is an arbitrage opportunity.

There are other linear programs we could consider in the search for arbitrage opportunities. An example is:

$$\begin{array}{ll} \text{minimize} & e^T(x^+ + x^-) \\ \text{subject to} & P(x^+ - x^-) \geq 0 \\ & \rho^T(x^+ - x^-) = -1 \\ & x^+ \geq 0 \\ & x^- \geq 0. \end{array}$$

This linear program involves two decision vectors: $x^+, x^- \in \Re^N$. The idea is to view the difference $x^+ - x^-$ as a vector of portfolio weights. Note that if (x^+, x^-) is a feasible solution to our new linear program, the portfolio $x^+ - x^-$ is an arbitrage opportunity, since $P(x^+ - x^-) \geq 0$ and $\rho^T(x^+ - x^-) = -1$. Further, if there exists an arbitrage opportunity x that offers a \$1 profit, there is a feasible solution (x^+, x^-) such that $x = x^+ - x^-$. Hence, our new linear program finds the \$1 arbitrage opportunity that minimizes $e^T(x^+ + x^-)$, which is the number of shares that must be traded in order to execute the opportunity. This objective is motivated by a notion that if there are multiple arbitrage opportunities, one that minimizes trading activity may be preferable.

3.7 Pattern Classification

Many engineering and managerial activities call for automated classification of observed data. Computer programs that classify observations are typically developed through *machine learning.* We discuss an example involving breast cancer diagnosis.

Example 3.7.1. (Breast Cancer Diagnosis) *Breast cancer is the second largest cause of cancer deaths among women. A breast cancer victim's chances for long-term survival are improved by early detection of the disease. The first sign of breast cancer is a lump in the breast. The majority of breast lumps are benign, however, so other means are required to diagnose breast cancer – that is, to distinguish malignant lumps from benign ones. One approach to diagnosing breast cancer involves extracting fluid from a lump and photographing cell nuclei through a microscope. Numerical measurements of the sizes and shapes of nuclei are recorded and used for diagnosis.*

An automated system for diagnosing breast cancer based on these numerical measurements has proven to be very effective. This system was developed through machine learning. In particular, a computer program processed a large collection of data samples, each of which was known to be associated with a malignant or benign lump. Through this process, the computer program "learned" patterns that distinguish malignant lumps from benign ones to produce a system for classifying subsequent samples.

Pattern classification can be thought of in terms of mapping a feature vector $a \in \Re^K$ to one of a finite set C of classes. Each feature vector is an encoding that represents an observation. For example, in breast cancer diagnosis, each feature vector represents measurements associated with cell nuclei from a breast lump. In this example, there are two classes corresponding to malignant and benign lumps.

Machine learning involves processing a set of feature vectors $u^1, \ldots, u^L \in \Re^K$ labeled with known classes $z_1, \ldots, z_L \in C$ to produce a mapping from $\Re^K$ to C. This mapping is then used to classify additional feature vectors. Machine learning has been used to generate pattern classifiers in many application domains other than breast cancer diagnosis. Examples include:

1. automatic recognition of handwritten alphabetical characters;
2. detection of faults in manufacturing equipment based on sensor data;
3. automatic matching of finger prints;
4. automatic matching of mug shots with composite sketches;
5. prediction of outcomes in sports events;
6. detection of favorable investment opportunities based on stock market data;
7. automatic recognition of spoken phonemes.

In this section, we discuss an approach to machine learning and pattern classification that involves formulation and solution of a linear program. We will only consider the case of two classes. This comes at no loss of generality, since methods that address two-class problems can also handle larger numbers of classes. In particular, for a problem with n classes, we could generate n classifiers, each distinguishing one class from the others. The combination of these classifiers addresses the n-class problem.

3.7.1 Linear Separation of Data

We consider two classes with labels $C = \{1, -1\}$. Samples in class 1 are referred to as *positive*, while samples in class -1 are *negative*. To develop a classifier we start with positive samples $u^1, \ldots, u^K \in \Re^N$ and negative samples $v^1, \ldots, v^L \in \Re^N$. The positive and negative samples are said to be *linearly separable* if there exists a hyperplane in $\Re^N$ such that all positive samples are on one side of the hyperplane and all negative samples are on the other. Figure 3.7 illustrates for $N = 2$ cases of samples that are linearly separable and samples that are not.

Recall that each hyperplane in $\Re^N$ takes the form $\{y | x^T y = \alpha\}$ for some $x \in \Re^N$ and $\alpha \in \Re$. In mathematical terms, the positive and negative samples are linearly separable if there is a vector $x \in \Re^N$ and a scalar $\alpha \in \Re$ such that $x^T u^k > 0$ and $x^T v^\ell < 0$ for all $k \in \{1, \ldots, K\}$ and $\ell \in \{1, \ldots, L\}$.

If positive and negative samples are linearly separable, a hyperplane that separates the data provides a classifier. In particular, given parameters $x \in \Re^N$ and $\alpha \in \Re$ for such a hyperplane, we classify a sample $y \in \Re^N$ as positive if $x^T y > \alpha$ and as negative if $x^T y < \alpha$. But how can we find appropriate parameters x and α? Linear programming offers one approach.

To obtain a separating hyperplane, we need to compute $x \in \Re^N$ and $\alpha \in \Re$ such that $x^T u^k > \alpha$ and $x^T v^\ell < \alpha$ for all k and ℓ. This does not quite fit into the linear programming framework, because the inequalities are strict. However, the problem can be converted to a linear program, as we now explain. Suppose that we have parameters $\overline{x}$ and $\overline{\alpha}$ satisfying the desired strict inequalities. They then also satisfy $\overline{x}^T u^k - \overline{\alpha} > 0$ and $\overline{x}^T v^\ell - \overline{\alpha} < 0$ for all k and ℓ. For a sufficiently large scalar β, we have $\beta(\overline{x}^T u^k - \overline{\alpha}) \geq 1$ and $\beta(\overline{x}^T v^\ell - \overline{\alpha}) \leq -1$, for all k and ℓ. Letting $x = \beta\overline{x}$ and $\alpha = \beta\overline{\alpha}$, we have $x^T u^k - \alpha \geq 1$ and $x^T v^\ell - \alpha \leq -1$, for all k and ℓ. It follows that – if there is a separating hyperplane – there is a hyperplane characterized by parameters x and α satisfying $x^T u^k - \alpha \geq 1$ and $x^T v^\ell - \alpha \leq -1$, for all k and ℓ.

Let

$$U = \begin{bmatrix} (u^1)^T \\ \vdots \\ (u^K)^T \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} (v^1)^T \\ \vdots \\ (v^K)^T \end{bmatrix},$$

so that our inequalities can be written more compactly as $Ux - \alpha e \geq e$ and $Vx - \alpha e \leq -e$, where e is the vector with every component equal to 1. A linear program with constraints $Ux - \alpha e \geq e$ and $Vx - \alpha e \leq -e$ and any objective function will find a hyperplane that separates positive and negative values, if one exists.

3.7.2 Minimizing Violations

What should we do when the positive and negative samples cannot be separated by a hyperplane? One might aim at minimizing the number of misclassifications. A *misclassification* is either a positive sample u^k such that $x^T u^k < 0$ or a negative sample v^ℓ such that $x^T v^\ell > 0$. Unfortunately, the problem of minimizing the number of misclassifications is very hard. In fact, there are no known methods for efficiently finding a hyperplane that minimizes the number of misclassifications.

One alternative is to find a hyperplane that minimizes the "extent" of misclassifications. In particular, given a hyperplane parameterized by $x \in \Re^N$ and $\alpha \in \Re$, for each kth positive sample, define the *violation* $\delta_k^+ = \max(-Ux + \alpha e + e, 0)$. Similarly, for each ℓth negative sample, define the *violation* $\delta_\ell^- = \max(Vx - \alpha e + e, 0)$. We therefore have two vectors: $\delta^+ \in \Re^K$ and $\delta^- \in \Re^L$. The violation associated with a sample exceeds 1 if and only if the sample is misclassified. The following linear program introduces decision variables δ^+ and δ^-, in addition to x and α, and minimizes the sum of violations:

$$\begin{array}{ll} \text{minimize} & e^T(\delta^+ + \delta^-) \\ \text{subject to} & \delta^+ \geq -Ux + \alpha e + e \\ & \delta^+ \geq 0 \\ & \delta^- \geq Vx - \alpha e + e \\ & \delta^- \geq 0. \end{array}$$

Figure 3.8 presents an example with $N = 2$ of a hyperplane produced by this linear program when positive and negative samples are not linearly separable.

3.8 Notes

The line of analysis presented in Sections 3.2 and 3.3 is adapted from Chapter 2 of *Introduction to Linear Optimization*, by Bertsimas and Tsitsiklis (1997).

The example of breast cancer diagnosis and the linear programming formulation for minimizing violations is taken from Mangasarien, Street, and Wolberg (1994), who have developed using linear programming a pattern classifier that is in use at University of Wisconsin Hospitals.

3.9 Exercises

Question 1

Add a single inequality constraint to $x \leq 0, y \leq 0$ so that the feasible region contains only one point.

Question 2

How many faces does the feasible set given by $x \geq 0, y \geq 0, z \geq 0, x+y+z = 1$ have. What common polyhedron is it? What is the maximum of $x + 2y + 3z$ over this polyhedron?

Question 3

Show that the feasible set constrained by $x \geq 0, y \geq 0, 2x + 5y \leq 3, -3x + 8y \leq -5$ is empty.

Question 4

Is $\Re^N$ convex? Show that it is, or explain why not.

Question 5

Draw a picture of a polyhedron in $\Re^2$ where one of the points in the polyhedron has 3 constraints active at the same time.

Question 6

In a particular polyhedron in $\Re^3$, one point has 3 constraints active at once. Does this point have to be a vertex? Why, or if not, give an example.

Question 7

Consider the following problem. Maximize $x + y$ subject to the constraints $x \geq 0, y \geq 0, -3x + 2y \leq -1, x - y \leq 2$. Is the feasible region bounded or unbounded? For a particular $L \geq 0$, find an x and y so that $[x\ y]^T$ is feasible, and also $x + y \geq L$. Note that x and y will depend on L.

Question 8

Dwight is a retiree who raises pigs for supplemental income. He is trying to decide what to feed his pigs, and is considering using a combination of feeds from some local suppliers. He would like to feed the pigs at minimum cost while making sure that each pig receives an adequate supply of calories and vitamins. The cost, calorie content, and vitamin supply of each feed is given in the table below.

$Contents$	$FeedTypeA$	$FeedtypeB$
$Calories(perpound)$	800	1000
$Vitamins(perpound)$	$140units$	$70units$
$Cost(perpound)$	\$0.40	\$0.80

Each pig requires at least 8000 calories per day and at least 700 units of vitamins. A further constraint is that no more than one-third of the diet (by weight) can consist of Feed Type A, since it contains an ingredient that is toxic if consumed in too large a quantity.

Formulate as a linear program and solve this in Excel. What is the resulting daily cost per pig.

Question 9

The Apex Television Company has to decide on the number of 27- and 20-inch sets to be produced at one of its factories. Market research indicates that at most 40 of the 27-inch sets and 10 of the 20-inch sets can be sold per month. The maximum number of work-hours available is 500 per month. a 27-inch set requires 10 work hours, and a 20-inch set requires 10 work-hours. Each 27-inch set sold produces a profit of \$120 and each 20-inch set produces a profit of \$80. A wholesaler has agreed to purchase all the television sets (at the market price) produced if the numbers do not exceed the amounts indicated by the market research.

Formulate as a linear program and solve for the maximum profit in Excel.

Question 10

The Metalco Company desires to blend a new alloy of 40 percent tin, 35 percent zinc and 25 percent lead from several available alloys having the following properties.

$Property$	$Alloy1$	$Alloy2$	$Alloy3$	$Alloy4$	$Alloy5$
$Percentageoftin$	60	25	45	20	50
$Percentageofzinc$	10	15	45	50	40
$Percentageoflead$	30	60	10	30	10
$Cost(\$perlb)$	22	20	25	24	27

The objective is to determine the proportions of these alloys that should be blended to produce the new allow at a minimum cost.

a) Formulate this as a linear program, and solve in Excel. b) How many alloys are used in the optimal solution?

Question 11

Suppose that two constraints in a system are $c^T x \leq 1$ and $d^T x \leq 1$, where c and d are linearly *dependent*. A constraint is called *redundant* if removing it does not change the feasible region.

a) If $c^T d \geq 0$ does this mean that one of c or d is redundant? If so, explain why. If not, give an example

b) If $c \leq d$ does this mean that one of c or d is redundant? If so, explain why. If not, give an example

Question 12

A paper mill makes rolls of paper that are 80" (80 inches) wide. Rolls are marketed in widths of 14", 31" and 36". An 80" roll may be cut (like a loaf of bread) into any combination of widths whose sum does not exceed 80". Suppose there are orders for 216 rolls of width 14", 87 rolls of width 31" and 341 rolls of with 36". The problem is to minimize the total number of 80" rolls required to fill the orders.

There are six ways – called "cuts" – in which we might consider to cut each roll into widths of 14", 31" and 36". The number of 14", 31" and 36" rolls resulting from each cut are given in the following table:

cut	14"	31"	36"
1	5	0	0
2	3	1	0
3	1	2	0
4	3	0	1
5	0	1	1
6	0	0	2

(a) Let $x_1, x_2, x_2, x_4, x_5, x_6$ be decision variables, each representing the number of rolls cut in one of the six ways. Describe a linear program that determines the minimum number of 80" rolls required to fill the orders (ignore the requirement that each x_i should be an integer).
(b) Solve the linear program using Excel.
(c) Suppose the orders for next month are yet to be decided. Can we determine in advance how many types of cuts will be needed? Can we determine in advance any cuts that will or will not be used? For each question if your answer is affirmative, then explain why, and if not, explain why not.

Question 13

MSE Airlines (pronounced "messy") needs to hire customer service agents. Research on customer demands has lead to the following requirements on the minimum number of customer service agents that need to be on duty at various times in any given day:

Time Period	**Staff Required**
6am to 8am	68
8am to 10am	90
10am to noon	56
Noon to 2pm	107
2pm to 4pm	80
4pm to 6pm	93
6pm to 8pm	62
8pm to 10pm	56
10pm to midnight	40
Midnight to 6am	15

The head of personnel would like to determine least expensive way to meet these staffing requirements. Each agent works an 8 hour shift, but not all shifts are available. The following table gives the available shifts and daily wages for agents working various shifts:

Shift	**Daily Wages**
6am-2pm	\$ 180
8am-4pm	\$ 170
10am-6pm	\$ 160
Noon-8pm	\$ 190
2pm-10pm	\$ 200
4pm-Midnight	\$ 210
10pm-6am	\$ 225
Midnight-8am	\$ 210

(a) Write a linear program that determines the least expensive way to meet staffing requirements.
(b) Solve the linear program using Excel.

Question 14

Consider the multi-stage production problem of producing chips and computers given in the lecture notes (Example 3.5.3). Suppose the net profits of selling one unit of Model 1, 2 and 3 are \$600, \$650 and \$800 respectively. Production of one unit of Model 1 consumes 1.5 % of the capacity of the assembly plant. Model 2 consumes 2% and Model 3, 2.5%. Production of one unit of Chip 1 and 2 use 2% of the capacity of fabrication facility 1, each. Production of one unit of Chip 2 uses 3% of the capacity of fabrication facility 2. Production of Chip 3 uses 4%. Model 1 needs one unit of Chip 1 and one unit of Chip 2, Model 2 needs one unit of Chip 1 and one unit of Chip 3, and Model 3 needs one unit of Chip 2 and one unit of Chip 3. Initially there are no chips in stock.

Formulate as a linear program and solve in Excel. What is the optimal production plan? How many activities are used? Is this the maximal number of activities that would be used for any basic feasible solution? Is this the minimal number of activities that would be used for any basic feasible solution?

Question 15

In the early 1900s, Edgar Anderson collected data on different species of iris' to study how species evolve to differentiate themselves in the course of evolution. In class, we studied three species – iris sesota, iris versicolor, and iris verginica – and used a linear program to show how iris sesota could be distinguished from the others based on sepal and petal dimensions. In particular, we showed that data on iris sesota was linearly separable from the other species data. The Excel file used in class is available on the course web site.

Is iris versicolor data linearly separable from the other species' data? If so determine parameters (x and α) for a separating hyperplane. If not, determine parameters for a hyperplane that minimizes the violation metric discussed in class and used in the spreadsheet, and determine the number of misclassifications resulting from this hyperplane.

Question 16

Consider a stock that in one year can take on any price within $\{1, 2, \ldots, 99, 100\}$ dollars. Suppose that there are four European put options available on the stock, and these are the only assets that we can trade at the moment. Their strike prices and current market prices for buying/selling are provided in the following table:

Strike Price	Market Price
10	1.5
20	2.55
30	4.59
40	8.72

Use Excel to show whether or not there is an arbitrage opportunity involving trading of these four assets.

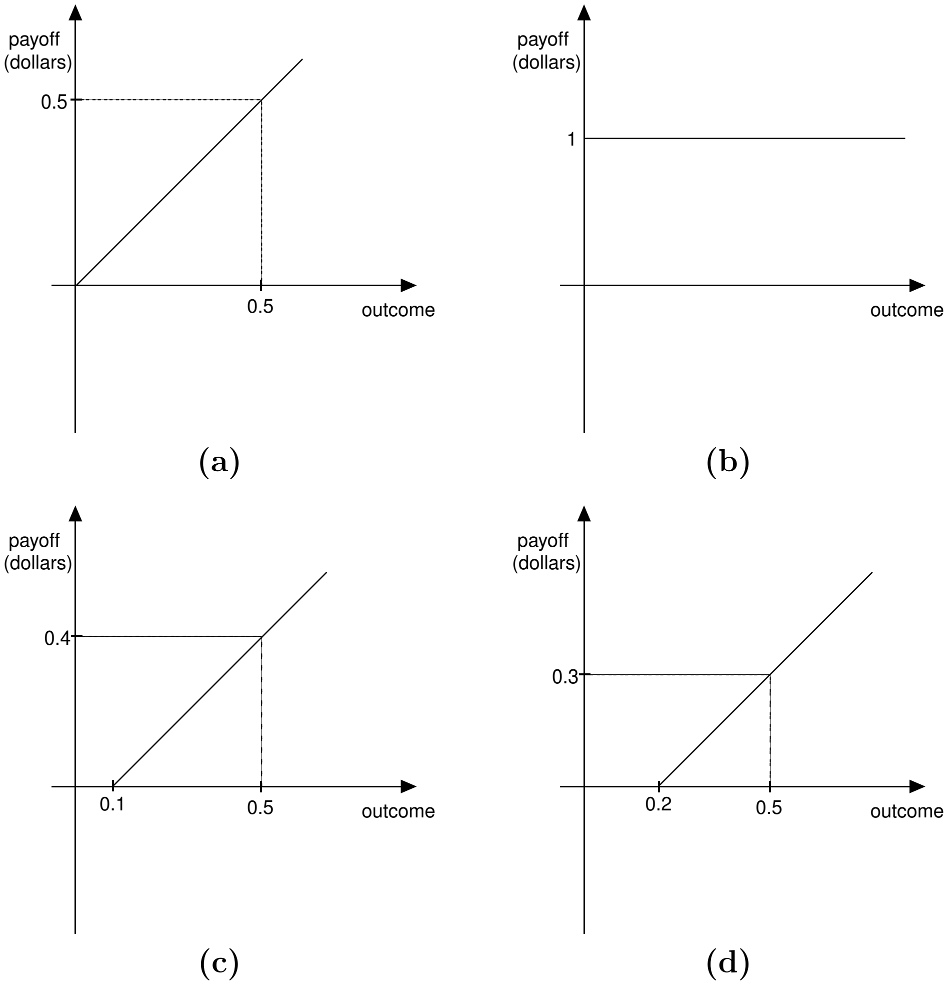

Figure 3.5: Payoff functions of the currency (a), a zero-coupon bond (b), a European call option with strike 0.1 (c), and a European call option with strike 0.2 (d).

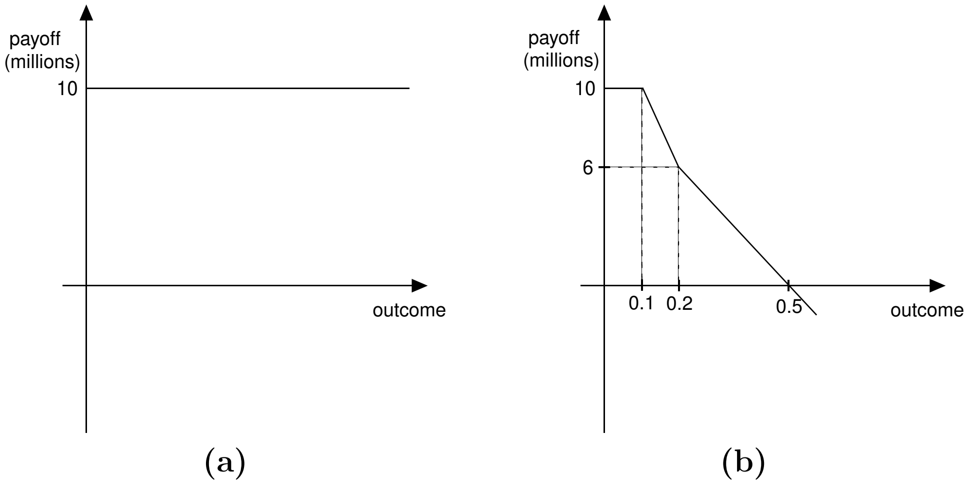

Figure 3.6: Payoff functions of two super-replicating portfolios: (a) one consisting only of bonds, and (b) one consisting of stocks and long and short positions in European call options.

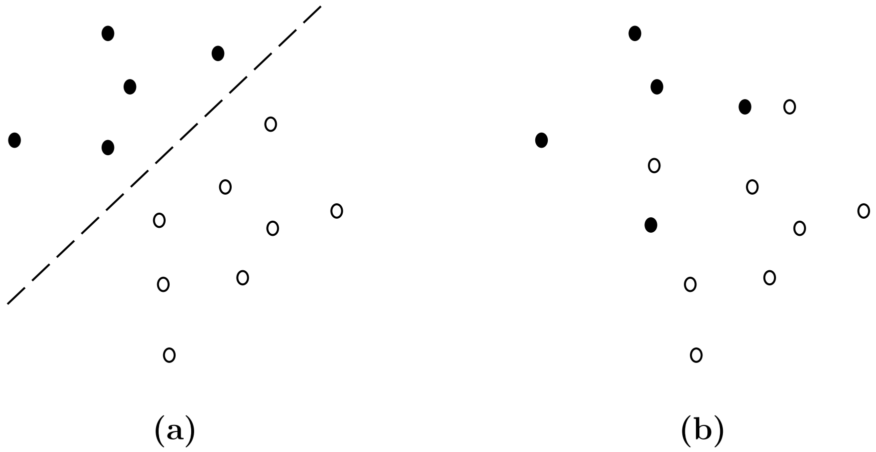

Figure 3.7: (a) Linearly separable data. The dashed line represents a separating hyperplane. (b) Data that are not linearly separable.

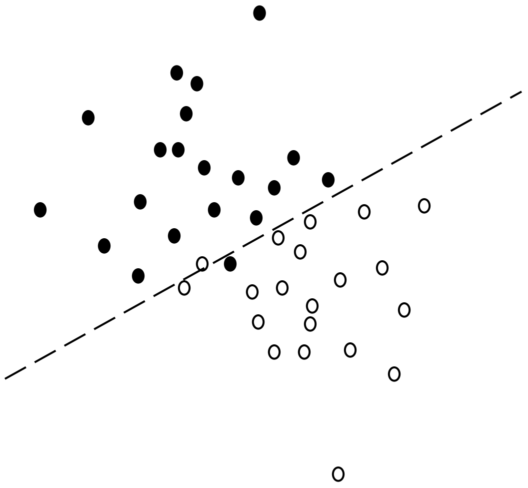

Figure 3.8: A hyperplane constructed by solving the linear program that minimizes violations. The resulting objective value was approximately 3.04.