Chapter 1
Introduction

1.1 Optimization

The concept of optimization is now well rooted as a principle underlying the analysis of many complex decision or allocation problems. It offers a certain degree of philosophical elegance that is hard to dispute, and it often offers an indispensable degree of operational simplicity. Using this optimization philosophy, one approaches a complex decision problem, involving the selection of values for a number of interrelated variables, by focusing attention on a single objective designed to quantify performance and measure the quality of the decision. This one objective is maximized (or minimized, depending on the formulation) subject to the constraints that may limit the selection of decision variable values. If a suitable single aspect of a problem can be isolated and characterized by an objective, be it profit or loss in a business setting, speed or distance in a physical problem, expected return in the environment of risky investments, or social welfare in the context of government planning, optimization may provide a suitable framework for analysis.

It is, of course, a rare situation in which it is possible to fully represent all the complexities of variable interactions, constraints, and appropriate objectives when faced with a complex decision problem. Thus, as with all quantitative techniques of analysis, a particular optimization formulation should be regarded only as an approximation. Skill in modeling, to capture the essential elements of a problem, and good judgment in the interpretation of results are required to obtain meaningful conclusions. Optimization, then, should be regarded as a tool of conceptualization and analysis rather than as a principle yielding the philosophically correct solution.

Skill and good judgment, with respect to problem formulation and interpretation of results, is enhanced through concrete practical experience and a thorough understanding of relevant theory. Problem formulation itself always involves a tradeoff between the conflicting objectives of building a mathematical model sufficiently complex to accurately capture the problem description and building a model that is

D.G. Luenberger, Y. Ye, *Linear and Nonlinear Programming*, International Series in Operations Research & Management Science 228,
DOI 10.1007/978-3-319-18842-3_1

tractable. The expert model builder is facile with both aspects of this tradeoff. One aspiring to become such an expert must learn to identify and capture the important issues of a problem mainly through example and experience; one must learn to distinguish tractable models from nontractable ones through a study of available technique and theory and by nurturing the capability to extend existing theory to new situations.

This book is centered around a certain optimization structure—that characteristic of linear and nonlinear programming. Examples of situations leading to this structure are sprinkled throughout the book, and these examples should help to indicate how practical problems can be often fruitfully structured in this form. The book mainly, however, is concerned with the development, analysis, and comparison of algorithms for solving general subclasses of optimization problems. This is valuable not only for the algorithms themselves, which enable one to solve given problems, but also because identification of the collection of structures they most effectively solve can enhance one's ability to formulate problems.

1.2 Types of Problems

The content of this book is divided into three major parts: Linear Programming, Unconstrained Problems, and Constrained Problems. The last two parts together comprise the subject of nonlinear programming.

Linear Programming

Linear programming is without doubt the most natural mechanism for formulating a vast array of problems with modest effort. A linear programming problem is characterized, as the name implies, by linear functions of the unknowns; the objective is linear in the unknowns, and the constraints are linear equalities or linear inequalities in the unknowns. One familiar with other branches of linear mathematics might suspect, initially, that linear programming formulations are popular because the mathematics is nicer, the theory is richer, and the computation simpler for linear problems than for nonlinear ones. But, in fact, these are *not* the primary reasons. In terms of mathematical and computational properties, there are much broader classes of optimization problems than linear programming problems that have elegant and potent theories and for which effective algorithms are available. It seems that the popularity of linear programming lies primarily with the formulation phase of analysis rather than the solution phase—and for good cause. For one thing, a great number of constraints and objectives that arise in practice *are* indisputably linear. Thus, for example, if one formulates a problem with a budget constraint restricting the total amount of money to be allocated among two different commodities, the budget constraint takes the form $x_1 + x_2 \leq B$, where x_j, $i = 1, 2$,

is the amount allocated to activity i, and B is the budget. Similarly, if the objective is, for example, maximum weight, then it can be expressed as $w_1x_1 + w_2x_2$, where $w_j,\ i = 1, 2$, is the unit weight of the commodity i. The overall problem would be expressed as

$$\begin{aligned} \text{maximize} \quad & w_1x_1 + w_2x_2 \\ \text{subject to} \quad & x_1 + x_2 \leq B, \\ & x_1 \geq 0,\ x_2 \geq 0, \end{aligned}$$

which is an elementary linear program. The linearity of the budget constraint is extremely natural in this case and does not represent simply an approximation to a more general functional form.

Another reason that linear forms for constraints and objectives are so popular in problem formulation is that they are often the least difficult to define. Thus, even if an objective function is not purely linear by virtue of its inherent definition (as in the above example), it is often far easier to define it as being linear than to decide on some other functional form and convince others that the more complex form is the best possible choice. Linearity, therefore, by virtue of its simplicity, often is selected as the easy way out or, when seeking generality, as the only functional form that will be equally applicable (or nonapplicable) in a class of similar problems.

Of course, the theoretical and computational aspects do take on a somewhat special character for linear programming problems—the most significant development being the simplex method. This algorithm is developed in Chaps. 2 and 3. More recent interior point methods are nonlinear in character and these are developed in Chap. 5.

Unconstrained Problems

It may seem that unconstrained optimization problems are so devoid of structural properties as to preclude their applicability as useful models of meaningful problems. Quite the contrary is true for two reasons. First, it can be argued, quite convincingly, that if the scope of a problem is broadened to the consideration of all relevant decision variables, there may then be no constraints—or put another way, constraints represent artificial delimitations of scope, and when the scope is broadened the constraints vanish. Thus, for example, it may be argued that a budget constraint is not characteristic of a meaningful problem formulation; since by borrowing at some interest rate it is always possible to obtain additional funds, and hence rather than introducing a budget constraint, a term reflecting the cost of funds should be incorporated into the objective. A similar argument applies to constraints describing the availability of other resources which at some cost (however great) could be supplemented.

The second reason that many important problems can be regarded as having no constraints is that constrained problems are sometimes easily converted to

unconstrained problems. For instance, the sole effect of equality constraints is simply to limit the degrees of freedom, by essentially making some variables functions of others. These dependencies can sometimes be explicitly characterized, and a new problem having its number of variables equal to the true degree of freedom can be determined. As a simple specific example, a constraint of the form $x_1 + x_2 = B$ can be eliminated by substituting $x_2 = B - x_1$ everywhere else that x_2 appears in the problem.

Aside from representing a significant class of practical problems, the study of unconstrained problems, of course, provides a stepping stone toward the more general case of constrained problems. Many aspects of both theory and algorithms are most naturally motivated and verified for the unconstrained case before progressing to the constrained case.

Constrained Problems

In spite of the arguments given above, many problems met in practice are formulated as constrained problems. This is because in most instances a complex problem such as, for example, the detailed production policy of a giant corporation, the planning of a large government agency, or even the design of a complex device cannot be directly treated in its entirety accounting for all possible choices, but instead must be decomposed into separate subproblems—each subproblem having constraints that are imposed to restrict its scope. Thus, in a planning problem, budget constraints are commonly imposed in order to decouple that one problem from a more global one. Therefore, one frequently encounters general nonlinear constrained mathematical programming problems.

The general mathematical programming problem can be stated as

$$\begin{aligned} &\text{minimize} && f(\mathbf{x}) \\ &\text{subject to} && h_j(\mathbf{x}) = 0,\ i = 1, 2, \ldots, m \\ &&& g_j(\mathbf{x}) \le 0,\ j = 1, 2,\ p \\ &&& \mathbf{x} \in S. \end{aligned}$$

In this formulation, $\mathbf{x}$ is an n-dimensional vector of unknowns, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and f, h_i, $i = 1, 2, \ldots, m$, and g_j, $j = 1, 2, \ldots, p$, are real-valued functions of the variables $x_1, x_2, \ldots, x_n$. The set S is a subset of n-dimensional space. The function f is the *objective function* of the problem and the equations, inequalities, and set restrictions are *constraints*.

Generally, in this book, additional assumptions are introduced in order to make the problem smooth in some suitable sense. For example, the functions in the problem are usually required to be continuous, or perhaps to have continuous derivatives. This ensures that small changes in $\mathbf{x}$ lead to small changes in other values associated with the problem. Also, the set S is not allowed to be arbitrary but usually is required to be a connected region of n-dimensional space, rather than, for example,

a set of distinct isolated points. This ensures that small changes in $\mathbf{x}$ can be made. Indeed, in a majority of problems treated, the set S is taken to be the entire space; there is no set restriction.

In view of these smoothness assumptions, one might characterize the problems treated in this book as *continuous variable programming*, since we generally discuss problems where all variables and function values can be varied continuously. In fact, this assumption forms the basis of many of the algorithms discussed, which operate essentially by making a series of small movements in the unknown $\mathbf{x}$ vector.

1.3 Size of Problems

One obvious measure of the complexity of a programming problem is its size, measured in terms of the number of unknown variables or the number of constraints. As might be expected, the size of problems that can be effectively solved has been increasing with advancing computing technology and with advancing theory. Today, with present computing capabilities, however, it is reasonable to distinguish three classes of problems: *small-scale problems* having about five or fewer unknowns and constraints; *intermediate-scale problems* having from about five to a hundred or a thousand variables; and *large-scale problems* having perhaps thousands or even millions of variables and constraints. This classification is not entirely rigid, but it reflects at least roughly not only size but the basic differences in approach that accompany different size problems. As a rough rule, small-scale problems can be solved by hand or by a small computer. Intermediate-scale problems can be solved on a personal computer with general purpose mathematical programming codes. Large-scale problems require sophisticated codes that exploit special structure and usually require large computers.

Much of the basic theory associated with optimization, particularly in nonlinear programming, is directed at obtaining necessary and sufficient conditions satisfied by a solution point, rather than at questions of computation. This theory involves mainly the study of Lagrange multipliers, including the Karush-Kuhn-Tucker Theorem and its extensions. It tremendously enhances insight into the philosophy of constrained optimization and provides satisfactory basic foundations for other important disciplines, such as the theory of the firm, consumer economics, and optimal control theory. The interpretation of Lagrange multipliers that accompanies this theory is valuable in virtually every optimization setting. As a basis for computing numerical solutions to optimization, however, this theory is far from adequate, since it does not consider the difficulties associated with solving the equations resulting from the necessary conditions.

If it is acknowledged from the outset that a given problem is too large and too complex to be efficiently solved by hand (and hence it is acknowledged that a computer solution is desirable), then one's theory should be directed toward development of procedures that exploit the efficiencies of computers. In most cases this

leads to the abandonment of the idea of solving the set of necessary conditions in favor of the more direct procedure of searching through the space (in an intelligent manner) for ever-improving points.

Today, search techniques can be effectively applied to more or less general nonlinear programming problems. Problems of great size, *large-scale programming* problems, can be solved if they possess special structural characteristics, especially sparsity, that can be exploited by a solution method. Today linear programming software packages are capable of automatically identifying sparse structure within the input data and taking advantage of this sparsity in numerical computation. It is now not uncommon to solve linear programs of up to a million variables and constraints, as long as the structure is sparse. Problem-dependent methods, where the structure is not automatically identified, are largely directed to transportation and network flow problems as discussed in the book.

This book focuses on the aspects of general theory that are most fruitful for computation in the widest class of problems. While necessary and sufficient conditions are examined and their application to small-scale problems is illustrated, our primary interest in such conditions is in their role as the core of a broader theory applicable to the solution of larger problems. At the other extreme, although some instances of structure exploitation are discussed, we focus primarily on the general continuous variable programming problem rather than on special techniques for special structures.

1.4 Iterative Algorithms and Convergence

The most important characteristic of a high-speed computer is its ability to perform repetitive operations efficiently, and in order to exploit this basic characteristic, most algorithms designed to solve large optimization problems are iterative in nature. Typically, in seeking a vector that solves the programming problem, an initial vector $\mathbf{x}_0$ is selected and the algorithm generates an improved vector $\mathbf{x}_1$. The process is repeated and a still better solution $\mathbf{x}_2$ is found. Continuing in this fashion, a sequence of ever-improving points $\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_k, \ldots$, is found that approaches a solution point $\mathbf{x}^*$. For linear programming problems solved by the simplex method, the generated sequence is of finite length, reaching the solution point exactly after a finite (although initially unspecified) number of steps. For nonlinear programming problems or interior-point methods, the sequence generally does not ever exactly reach the solution point, but converges toward it. In operation, the process is terminated when a point sufficiently close to the solution point, for practical purposes, is obtained.

The theory of iterative algorithms can be divided into three (somewhat overlapping) aspects. The first is concerned with the creation of the algorithms themselves. Algorithms are not conceived arbitrarily, but are based on a creative examination of the programming problem, its inherent structure, and the efficiencies of digital computers. The second aspect is the verification that a given algorithm will in fact

generate a sequence that converges to a solution point. This aspect is referred to as *global convergence analysis*, since it addresses the important question of whether the algorithm, when initiated far from the solution point, will eventually converge to it. The third aspect is referred to as *local convergence analysis* or *complexity analysis* and is concerned with the rate at which the generated sequence of points converges to the solution. One cannot regard a problem as solved simply because an algorithm is known which will converge to the solution, since it may require an exorbitant amount of time to reduce the error to an acceptable tolerance. It is essential when prescribing algorithms that some estimate of the time required be available. It is the convergence-rate aspect of the theory that allows some quantitative evaluation and comparison of different algorithms, and at least crudely, assigns a measure of tractability to a problem, as discussed in Sect. 1.1.

A modern-day technical version of Confucius' most famous saying, and one which represents an underlying philosophy of this book, might be, "One good theory is worth a thousand computer runs." Thus, the convergence properties of an iterative algorithm can be estimated with confidence either by performing numerous computer experiments on different problems or by a simple well-directed theoretical analysis. A simple theory, of course, provides invaluable insight as well as the desired estimate.

For linear programming using the simplex method, solid theoretical statements on the speed of convergence were elusive, because the method actually converges to an exact solution in a finite number of steps. The question is how many steps might be required. This question was finally resolved when it was shown that it was possible for the number of steps to be exponential in the size of the program. The situation is different for interior point algorithms, which essentially treat the problem by introducing nonlinear terms, and which therefore do not generally obtain a solution in a finite number of steps but instead converge toward a solution.

For nonlinear programs, including interior point methods applied to linear programs, it is meaningful to consider the speed of convergence. There are many different classes of nonlinear programming algorithms, each with its own convergence characteristics. However, in many cases the convergence properties can be deduced analytically by fairly simple means, and this analysis is substantiated by computational experience. Presentation of convergence analysis, which seems to be the natural focal point of a theory directed at obtaining specific answers, is a unique feature of this book.

There are in fact two aspects of convergence-rate theory. The first is generally known as *complexity analysis* and focuses on how fast the method converges overall, distinguishing between polynomial-time algorithms and non-polynomial-time algorithms. The second aspect provides more detailed analysis of how fast the method converges in the final stages, and can provide comparisons between different algorithms. Both of these are treated in this book.

The convergence-rate theory presented has two somewhat surprising but definitely pleasing aspects. First, the theory is, for the most part, extremely simple in nature. Although initially one might fear that a theory aimed at predicting the speed of convergence of a complex algorithm might itself be doubly complex, in fact the

associated convergence analysis often turns out to be exceedingly elementary, requiring only a line or two of calculation. Second, a large class of seemingly distinct algorithms turns out to have a common convergence rate. Indeed, as emphasized in the later chapters of the book, there is a *canonical rate* associated with a given programming problem that seems to govern the speed of convergence of many algorithms when applied to that problem. It is this fact that underlies the potency of the theory, allowing definitive comparisons among algorithms to be made even without detailed knowledge of the problems to which they will be applied. Together these two properties, simplicity and potency, assure convergence analysis a permanent position of major importance in mathematical programming theory.

Part I
Linear Programming

Chapter 2
Basic Properties of Linear Programs

2.1 Introduction

A linear program (LP) is an optimization problem in which the objective function is linear in the unknowns and the constraints consist of linear equalities and linear inequalities. The exact form of these constraints may differ from one problem to another, but as shown below, any linear program can be transformed into the following *standard form*:

$$\begin{array}{lll} \text{minimize} & c_1x_1 + c_2x_2 + \ldots + c_nx_n & \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & = b_2 \\ & \vdots & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = b_m \\ \text{and} & x_1 \geqslant 0,\ x_2 \geqslant 0,\ \ldots,\ x_n \geqslant 0, & \end{array} \tag{2.1}$$

where the b_i's, c_i's and a_{ij}'s are fixed real constants, and the x_i's are real numbers to be determined. We always assume that each equation has been multiplied by minus unity, if necessary, so that each $b_i \geqslant 0$.

In more compact vector notation,[1] this standard problem becomes

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{and} \quad \mathbf{x} \geqslant \mathbf{0}. \end{array} \tag{2.2}$$

Here $\mathbf{x}$ is an n-dimensional column vector, $\mathbf{c}^T$ is an n-dimensional row vector, $\mathbf{A}$ is an $m \times n$ matrix, and $\mathbf{b}$ is an m-dimensional column vector. The vector inequality $\mathbf{x} \geqslant \mathbf{0}$ means that each component of $\mathbf{x}$ is nonnegative.

[1] See Appendix A for a description of the vector notation used throughout this book.

D.G. Luenberger, Y. Ye, *Linear and Nonlinear Programming*, International Series in Operations Research & Management Science 228,
DOI 10.1007/978-3-319-18842-3_2

Before giving some examples of areas in which linear programming problems arise naturally, we indicate how various other forms of linear programs can be converted to the standard form.

Example 1 (Slack Variables). Consider the problem

$$\begin{array}{lll} \text{minimize} & c_1x_1 + c_2x_2 + \cdots + c_nx_n & \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & \leqslant b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & \leqslant b_2 \\ & \vdots & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & \leqslant b_m \\ \text{and} & x_1 \geqslant 0,\ x_2 \geqslant 0,\ \ldots,\ x_n \geqslant 0, & \end{array}$$

In this case the constraint set is determined entirely by linear inequalities. The problem may be alternatively expressed as

$$\begin{array}{lll} \text{minimize} & c_1x_1 + c_2x_2 + \cdots + c_nx_n & \\ \text{subject to} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + y_1 & = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + y_2 & = b_2 \\ & \vdots & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n + y_m & = b_m \\ \text{and} & x_1 \geqslant 0,\ x_2 \geqslant 0,\ \ldots,\ x_n \geqslant 0, & \\ \text{and} & y_1 \geqslant 0,\ y_2 \geqslant 0,\ \ldots,\ y_m \geqslant 0. & \end{array}$$

The new positive variables y_i introduced to convert the inequalities to equalities are called *slack variables* (or more loosely, *slacks*). By considering the problem as one having $n + m$ unknowns $x_1,\ x_2,\ \ldots,\ x_n,\ y_1,\ y_2,\ \ldots,\ y_m$, the problem takes the standard form. The $m \times (n + m)$ matrix that now describes the linear equality constraints is of the special form $[\mathbf{A},\ \mathbf{I}]$ (that is, its columns can be partitioned into two sets; the first n columns make up the original A matrix and the last m columns make up an $m \times m$ identity matrix).

Example 2 (Surplus Variables). If the linear inequalities of Example 1 are reversed so that a typical inequality is

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geqslant b_i,$$

it is clear that this is equivalent to

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - y_i = b_i$$

with $y_i \geqslant 0$. Variables, such as y_i, adjoined in this fashion to convert a "greater than or equal to" inequality to equality are called *surplus variables*.

It should be clear that by suitably multiplying by minus unity, and adjoining slack and surplus variables, any set of linear inequalities can be converted to standard form if the unknown variables are restricted to be nonnegative.

Example 3 (Free Variables—First Method). If a linear program is given in standard form except that one or more of the unknown variables is not required to be nonnegative, the problem can be transformed to standard form by either of two simple techniques.

To describe the first technique, suppose in (2.1), for example, that the restriction $x_1 \geqslant 0$ is not present and hence x_1 is free to take on either positive or negative values. We then write

$$x_1 = u_1 - v_1, \tag{2.3}$$

where we require $u_1 \geqslant 0$ and $v_1 \geqslant 0$. If we substitute $u_1 - v_1$ for x_1 everywhere in (2.1), the linearity of the constraints is preserved and all variables are now required to be nonnegative. The problem is then expressed in terms of the $n + 1$ variables $u_1, v_1, x_2, x_3, \ldots, x_n$.

There is obviously a certain degree of redundancy introduced by this technique, however, since a constant added to u_1 and v_1 does not change x_1 (that is, the representation of a given value x_1 is not unique). Nevertheless, this does not hinder the simplex method of solution.

Example 4 (Free Variables—Second Method). A second approach for converting to standard form when x_1 is unconstrained in sign is to eliminate x_1 together with one of the constraint equations. Take any one of the m equations in (2.1) which has a nonzero coefficient for x_1. Say, for example,

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i, \tag{2.4}$$

where $a_{i1} \neq 0$. Then x_1 can be expressed as a linear combination of the other variables plus a constant. If this expression is substituted for x_1 everywhere in (2.1), we are led to a new problem of exactly the same form but expressed in terms of the variables $x_2, x_3, \ldots, x_n$ only. Furthermore, the ith equation, used to determine x_1, is now identically zero and it too can be eliminated. This substitution scheme is valid since any combination of nonnegative variables $x_2, x_3, \ldots, x_n$ leads to a feasible x_1 from (2.4), since the sign of x_1 is unrestricted. As a result of this simplification, we obtain a standard linear program having $n - 1$ variables and $m - 1$ constraint equations. The value of the variable x_1 can be determined after solution through (2.4).

Example 5 (Specific Case). As a specific instance of the above technique consider the problem

$$\begin{aligned} \text{minimize} \quad & x_1 + 3x_2 + 4x_3 \\ \text{subject to} \quad & x_1 + 2x_2 + x_3 = 5 \\ & 2x_1 + 3x_2 + x_3 = 6 \\ & x_2 \geqslant 0, \qquad x_3 \geqslant 0. \end{aligned}$$

Since x_1 is free, we solve for it from the first constraint, obtaining

$$x_1 = 5 - 2x_2 - x_3. \tag{2.5}$$

Substituting this into the objective and the second constraint, we obtain the equivalent problem (subtracting five from the objective)

$$\begin{aligned} \text{minimize} \quad & x_2 + 3x_3 \\ \text{subject to} \quad & x_2 + x_3 = 4 \\ & x_2 \geqslant 0,\ x_3 \geqslant 0, \end{aligned}$$

which is a problem in standard form. After the smaller problem is solved (the answer is $x_2 = 4,\ x_3 = 0$) the value for $x_1 (x_1 = -3)$ can be found from (2.5).

2.2 Examples of Linear Programming Problems

Linear programming has long proved its merit as a significant model of numerous allocation problems and economic phenomena. The continuously expanding literature of applications repeatedly demonstrates the importance of linear programming as a general framework for problem formulation. In this section we present some classic examples of situations that have natural formulations.

Example 1 (The Diet Problem). How can we determine the most economical diet that satisfies the basic minimum nutritional requirements for good health? Such a problem might, for example, be faced by the dietitian of a large army. We assume that there are available at the market n different foods and that the jth food sells at a price c_j per unit. In addition there are m basic nutritional ingredients and, to achieve a balanced diet, each individual must receive at least b_i units of the ith nutrient per day. Finally, we assume that each unit of food j contains a_{ij} units of the ith nutrient.

If we denote by x_j the number of units of food j in the diet, the problem then is to select the x_j's to minimize the total cost

$$c_1 x_1 + c_2 x_2 + \cdots + c_n x_n$$

subject to the nutritional constraints

$$a_{i1} x_1 + a_{i2} x_2 + \cdots + a_{in} x_n \geqslant b_i,\ i = 1, \ldots, m,$$

and the nonnegativity constraints

$$x_1 \geqslant 0,\ x_2 \geqslant 0,\ \ldots,\ x_n \geqslant 0$$

on the food quantities.

This problem can be converted to standard form by subtracting a nonnegative surplus variable from the left side of each of the m linear inequalities. The diet problem is discussed further in Chap. 4.

Example 2 (Manufacturing Problem). Suppose we own a facility that is capable of manufacturing n different products, each of which may require various amounts

of m different resources. Each product can be produced at any level $x_j \geqslant 0$, $j = 1, 2, \ldots, n$, and each unit of the jth product can sell for p_j dollars and needs a_{ij} units of the ith resource, $i = 1, 2, \ldots, m$. Assuming linearity of the production facility, if we are given a set of m numbers $b_1, b_2, \ldots, b_m$ describing the available quantities of the m resources, and we wish to manufacture products at maximum revenue, ours decision problem is a linear program to maximize

$$p_1 x_1 + p_2 x_2 + \cdots + p_n x_n$$

subject to the resource constraints

$$a_{i1} x_1 + a_{i2} x_2 + \cdots + a_{in} x_n \leqslant b_i, \; i = 1, \ldots, m$$

and the nonnegativity constraints on all production variables.

Example 3 (The Transportation Problem). Quantities $a_1, a_2, \ldots, a_m$, respectively, of a certain product are to be shipped from each of m locations and received in amounts $b_1, b_2, \ldots, b_n$, respectively, at each of n destinations. Associated with the shipping of a unit of product from origin i to destination j is a shipping cost c_{ij}. It is desired to determine the amounts x_{ij} to be shipped between each origin–destination pair $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$; so as to satisfy the shipping requirements and minimize the total cost of transportation.

To formulate this problem as a linear programming problem, we set up the array shown below:

$$\begin{array}{|cccc|c} \hline x_{11} & x_{12} & \cdots & x_{1n} & a_1 \\ x_{21} & x_{22} & \cdots & x_{2n} & a_2 \\ \cdot & & & \cdot & \cdot \\ \cdot & & & \cdot & \cdot \\ \cdot & & & \cdot & \cdot \\ x_{m1} & x_{m2} & \cdots & x_{mn} & a_m \\ \hline \end{array}$$
$$b_1 \quad b_2 \quad \cdots \; b_n$$

The ith row in this array defines the variables associated with the ith origin, while the jth column in this array defines the variables associated with the jth destination. The problem is to place nonnegative variables x_{ij} in this array so that the sum across the ith row is a_j, the sum down the jth column is b_j, and the weighted sum $\sum_{j=1}^{n} \sum_{i=1}^{m} c_{ij} x_{ij}$, representing the transportation cost, is minimized.

Thus, we have the linear programming problem:

$$\text{minimize} \quad \sum_{ij} c_{ij} x_{ij}$$

$$\text{subject to} \quad \sum_{j=1}^{n} x_{ij} = a_j \quad \text{for} \quad i = 1, 2, \ldots, m \tag{2.6}$$

$$\sum_{i=1}^{m} x_{ij} = b_j \quad \text{for} \quad j = 1, 2, \ldots, n \tag{2.7}$$

$$x_{ij} \geqslant 0 \text{ for } i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n.$$

In order that the constraints (2.6) and (2.7) be consistent, we must, of course, assume that $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$ which corresponds to assuming that the total amount shipped is equal to the total amount received.

The transportation problem is now clearly seen to be a linear programming problem in mn variables. The equations (2.6) and (2.7) can be combined and expressed in matrix form in the usual manner and this results in an $(m + n) \times (mn)$ coefficient matrix consisting of zeros and ones only.

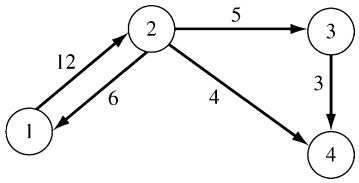

Fig. 2.1 A network with capacities

Example 4 (The Maximal Flow Problem). Consider a capacitated network (see Fig. 2.1, and Appendix D) in which two special nodes, called the *source* and the *sink*, are distinguished. Say they are nodes 1 and m, respectively. All other nodes must satisfy the strict conservation requirement; that is, the net flow into these nodes must be zero. However, the source may have a net outflow and the sink a net inflow. The outflow f of the source will equal the inflow of the sink as a consequence of the conservation at all other nodes. A set of arc flows satisfying these conditions is said to be a *flow* in the network of value f. The maximal flow problem is that of determining the maximal flow that can be established in such a network. When written out, it takes the form

$$\begin{aligned}
&\text{minimize} \quad f \\
&\text{subject to} \quad \sum_{j=1}^{n} x_{1j} - \sum_{j=1}^{n} x_{j1} - f = 0 \\
&\qquad\qquad \sum_{j=1}^{n} x_{ij} - \sum_{j=1}^{n} x_{ji} \qquad = 0, \qquad i \neq 1,\ m \qquad (2.8) \\
&\qquad\qquad \sum_{j=1}^{n} x_{mj} - \sum_{j=1}^{n} x_{jm} + f = 0 \\
&\qquad\qquad 0 \leq x_{ij} \leq k_{ij}, \qquad \text{forall } i,\ j,
\end{aligned}$$

where $k_{ij} = 0$ for those no-arc pairs (i, j).

Example 5 (A Warehousing Problem). Consider the problem of operating a warehouse, by buying and selling the stock of a certain commodity, in order to maximize profit over a certain length of time. The warehouse has a fixed capacity C, and there is a cost r per unit for holding stock for one period. The price, p_i, of the commodity is known to fluctuate over a number of time periods—say months, indexed by i. In any period the same price holds for both purchase or sale. The warehouse is originally empty and is required to be empty at the end of the last period.

To formulate this problem, variables are introduced for each time period. In particular, let x_i denote the level of stock in the warehouse at the beginning of period i. Let u_i denote the amount bought during period i, and let s_i denote the amount sold during period i. If there are n periods, the problem is

$$\begin{array}{lllll} \text{maximize} & \sum_{i=1}^{n}(p_i(s_i-u_i)-rx_i) & & & \\ \text{subject to} & x_{i+1} & = x_i+u_i-s_i & & i=1,2,\ \ldots,\ n-1 \\ & 0 & = x_n+u_n-s_n & & \\ & x_i+z_i & = C & & i=2,\ \ldots,\ n \\ & x_1=0,\ x_i\geqslant 0,\ u_i\geqslant 0, & & & s_i\geqslant 0,\ z_i\geqslant 0, \end{array}$$

where z_i is a slack variable. If the constraints are written out explicitly for the case $n = 3$, they take the form

$-u_1+s_1$	$+x_2$		$=0$
	$-x_2-u_2+s_2$	$+x_3$	$=0$
	$x_2 \quad +z_2$		$=C$
		$-x_3-u_3+s_3$	$=0$
		$x_3 \quad +z_3$	$=C$

Note that the coefficient matrix can be partitioned into blocks corresponding to the variables of the different time periods. The only blocks that have nonzero entries are the diagonal ones and the ones immediately above the diagonal. This structure is typical of problems involving time.

Example 6 (Linear Classifier and Support Vector Machine). Suppose several d-dimensional data points are classified into two distinct classes. For example, two-dimensional data points may be grade averages in science and humanities for different students. We also know the academic major of each student, as being in science or humanities, which serves as the classification. In general we have vectors $\mathbf{a}_i \in E^d$ for $i = 1, 2, \ldots, n_1$ and vectors $\mathbf{b}_j \in E^d$ for $j = 1, 2, \ldots, n_2$. We wish to find a hyperplane that separates the $\mathbf{a}_i$'s from the $\mathbf{b}_j$'s. Mathematically we wish to find $\mathbf{y} \in E^d$ and a number β such that

$$\begin{aligned} \mathbf{a}_i^T\mathbf{y}+\beta &\geqslant 1 \quad \text{for all } i \\ \mathbf{b}_j^T\mathbf{y}+\beta &\leqslant -1 \text{ for all } j, \end{aligned}$$

where $\{\mathbf{x} : \mathbf{x}^T\mathbf{y} + \beta = 0\}$ is the desired hyperplane, and the separation is defined by the $+1$ and -1. This is a linear program. See Fig. 2.2.

Example 7 (Combinatorial Auction). Suppose there are m mutually exclusive potential states and only one of them will be true at maturity. For example, the states may correspond to the winning horse in a race of m horses, or the value of a stock index, falling within m intervals. An auction organizer who establishes a *parimutuel* auction is prepared to issue contracts specifying subsets of the m possibilities that pay \$1 if the final state is one of those designated by the contract, and zero otherwise. There are n participants who may place orders with the organizer for the purchase of such contracts. An order by the jth participant consists of an m-vector $\mathbf{a}_j = (a_{1j}, a_{2j}, \ldots, a_{mj})^T$ where each component is either 0 or 1, a one indicating a desire to be paid if the corresponding state occurs.

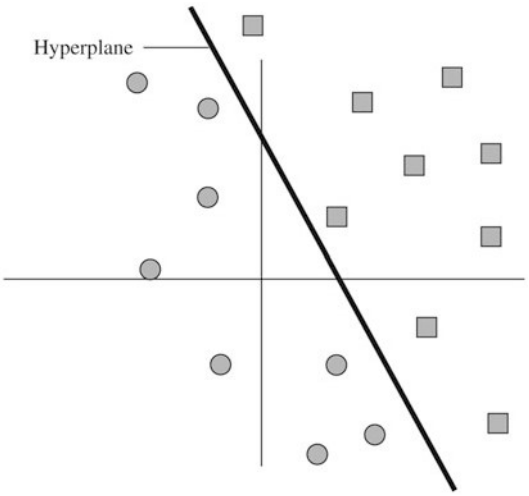

Fig. 2.2 Support vector for data classification

Accompanying the order is a number π_j which is the price limit the participant is willing to pay for one unit of the order. Finally, the participant also declares the maximum number q_j of units he or she is willing to accept under these terms.

The auction organizer, after receiving these various orders, must decide how many contracts to fill. Let x_j be the (real) number of units awarded to the jth order. Then the jth participant will pay $\boldsymbol{\pi}_j x_j$. The total amount paid by all participants is $\boldsymbol{\pi}^T \mathbf{x}$, where $\mathbf{x}$ is the vector of x_j's and π is the vector of prices.

If the outcome is the ith state, the auction organizer must pay out a total of $\sum_{j=1}^{n} a_{ij} x_j = (\mathbf{A}\mathbf{x})_j$. The organizer would like to maximize profit in the worst possible case, and does this by solving the problem

$$\begin{aligned} &\text{maximize} \quad \boldsymbol{\pi}^T \mathbf{x} - \max_i (\mathbf{A}\mathbf{x})_i \\ &\text{subject to} \quad \mathbf{0} \leqslant \mathbf{x} \leqslant \mathbf{q}. \end{aligned}$$

This problem can be expressed alternatively as selecting $\mathbf{x}$ and scalar s to

$$\begin{aligned} \text{maximize} \quad & \boldsymbol{\pi}^T\mathbf{x} - s \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} - \mathbf{1}s \leqslant \mathbf{0} \\ & \mathbf{0} \leqslant \mathbf{x} \leqslant \mathbf{q} \end{aligned}$$

where $\mathbf{1}$ is the vector of all 1's. Notice that the profit will always be nonnegative, since $\mathbf{x} = \mathbf{0}$ is feasible.

2.3 Basic Solutions

Consider the system of equalities

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{2.9}$$

where $\mathbf{x}$ is an n-vector, $\mathbf{b}$ an m-vector, and $\mathbf{A}$ is an $m \times n$ matrix. Suppose that from the n columns of $\mathbf{A}$ we select a set of m linearly independent columns (such a set exists if the rank of $\mathbf{A}$ is m). For notational simplicity assume that we select the first m columns of $\mathbf{A}$ and denote the $m \times m$ matrix determined by these columns by $\mathbf{B}$. The matrix $\mathbf{B}$ is then nonsingular and we may uniquely solve the equation.

$$\mathbf{B}\mathbf{x}_\mathbf{B} = \mathbf{b} \tag{2.10}$$

for the m-vector $\mathbf{x}_\mathbf{B}$. By putting $\mathbf{x} = (\mathbf{x}_\mathbf{B}, \mathbf{0})$ (that is, setting the first m components of $\mathbf{x}$ equal to those of $\mathbf{x}_\mathbf{B}$ and the remaining components equal to zero), we obtain a solution to $\mathbf{A}\mathbf{x} = \mathbf{b}$. This leads to the following definition.

Definition. Given the set of m simultaneous linear equations in n unknowns (2.9), let $\mathbf{B}$ be any nonsingular $m \times m$ submatrix made up of columns of $\mathbf{A}$. Then, if all $n-m$ components of $\mathbf{x}$ not associated with columns of $\mathbf{B}$ are set equal to zero, the solution to the resulting set of equations is said to be a *basic solution* to (2.9) with respect to the basis B. The components of $\mathbf{x}$ associated with columns of $\mathbf{B}$ are called *basic variables*.

In the above definition we refer to $\mathbf{B}$ as a basis, since $\mathbf{B}$ consists of m linearly independent columns that can be regarded as a basis for the space E^m. The basic solution corresponds to an expression for the vector $\mathbf{b}$ as a linear combination of these basis vectors. This interpretation is discussed further in the next section.

In general, of course, Eq. (2.9) may have no basic solutions. However, we may avoid trivialities and difficulties of a nonessential nature by making certain elementary assumptions regarding the structure of the matrix $\mathbf{A}$. First, we usually assume that $n > m$, that is, the number of variables x_j exceeds the number of equality constraints. Second, we usually assume that the rows of $\mathbf{A}$ are linearly independent, corresponding to linear independence of the m equations. $\mathbf{A}$ linear dependency among the rows of $\mathbf{A}$ would lead either to contradictory constraints and hence no solutions to (2.9), or to a redundancy that could be eliminated. Formally, we explicitly make the following assumption in our development, unless noted otherwise.

Full Rank Assumption. *The $m \times n$ matrix* $\mathbf{A}$ *has* $m < n$, *and the* m *rows of* $\mathbf{A}$ *are linearly independent.*

Under the above assumption, the system (2.9) will always have a solution and, in fact, it will always have at least one basic solution.

The basic variables in a basic solution are not necessarily all nonzero. This is noted by the following definition.

Definition. If one or more of the basic variables in a basic solution has value zero, that solution is said to be a *degenerate basic solution.*

We note that in a nondegenerate basic solution the basic variables, and hence the basis $\mathbf{B}$, can be immediately identified from the positive components of the solution. There is ambiguity associated with a degenerate basic solution, however, since the zero-valued basic and some of nonbasic variables can be interchanged.

So far in the discussion of basic solutions we have treated only the equality constraint (2.9) and have made no reference to positivity constraints on the variables. Similar definitions apply when these constraints are also considered. Thus, consider now the system of constraints

$$\mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}, \tag{2.11}$$

which represent the constraints of a linear program in standard form.

Definition. A vector $\mathbf{x}$ satisfying (2.11) is said to be *feasible* for these constraints. A feasible solution to the constraints (2.11) that is also basic is said to be a *basic feasible solution*; if this solution is also a degenerate basic solution, it is called a *degenerate basic feasible solution.*

2.4 The Fundamental Theorem of Linear Programming

In this section, through the fundamental theorem of linear programming, we establish the primary importance of basic feasible solutions in solving linear programs. The method of proof of the theorem is in many respects as important as the result itself, since it represents the beginning of the development of the simplex method. The theorem (due to Carathéodory) itself shows that it is necessary only to consider basic feasible solutions when seeking an optimal solution to a linear program because the optimal value is always achieved at such a solution.

Corresponding to a linear program in standard form

$$\begin{aligned} &\text{minimize} \quad \mathbf{c}^T\mathbf{x} \\ &\text{subject to} \quad \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0} \end{aligned} \tag{2.12}$$

a feasible solution to the constraints that achieves the minimum value of the objective function subject to those constraints is said to be an *optimal feasible solution*. If this solution is basic, it is an *optimal basic feasible solution*.

Fundamental Theorem of Linear Programming. *Given a linear program in standard form (2.12) where* $\mathbf{A}$ *is an* $m \times n$ *matrix of rank* m,

i) *if there is a feasible solution, there is a basic feasible solution*;
ii) *if there is an optimal feasible solution, there is an optimal basic feasible solution.*

Proof of (i). Denote the columns of $\mathbf{A}$ by $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$. Suppose $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is a feasible solution. Then, in terms of the columns of $\mathbf{A}$, this solution satisfies:

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_n\mathbf{a}_n = \mathbf{b}.$$

Assume that exactly p of the variables x_i are greater than zero, and for convenience, that they are the first p variables. Thus

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_p\mathbf{a}_p = \mathbf{b}. \tag{2.13}$$

There are now two cases, corresponding as to whether the set $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_p$ is linearly independent or linearly dependent.

Case 1: Assume $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_p$ are linearly independent. Then clearly, $p \leqslant m$. If $p = m$, the solution is basic and the proof is complete. If $p < m$, then, since A has rank m, $m - p$ vectors can be found from the remaining $n - p$ vectors so that the resulting set of m vectors is linearly independent. (See Exercise 12.) Assigning the value zero to the corresponding $m - p$ variables yields a (degenerate) basic feasible solution.

Case 2: Assume $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_p$ are linearly dependent. Then there is a nontrivial linear combination of these vectors that is zero. Thus there are constants $y_1, y_2, \ldots, y_p$, at least one of which can be assumed to be positive, such that

$$y_1\mathbf{a}_1 + y_2\mathbf{a}_2 + \cdots + y_p\mathbf{a}_p = \mathbf{0}. \tag{2.14}$$

Multiplying this equation by a scalar ε and subtracting it from (2.13), we obtain

$$(x_1 - \varepsilon y_1)\mathbf{a}_1 + (x_2 - \varepsilon y_2)\mathbf{a}_2 + \cdots + (x_p - \varepsilon y_p)\mathbf{a}_p = \mathbf{b}. \tag{2.15}$$

This equation holds for every ε, and for each ε the components $x_j - \varepsilon y_j$ correspond to a solution of the linear equalities—although they may violate $x_i - \varepsilon y_i \geqslant 0$. Denoting $\mathbf{y} = (y_1, y_2, \ldots, y_p, 0, 0, \ldots, 0)$, we see that for any ε

$$\mathbf{x} - \varepsilon\mathbf{y} \tag{2.16}$$

is a solution to the equalities. For $\varepsilon = 0$, this reduces to the original feasible solution. As ε is increased from zero, the various components increase, decrease, or remain constant, depending upon whether the corresponding y_i is negative, positive, or zero. Since we assume at least one y_i is positive, at least one component will decrease as ε is increased. We increase ε to the first point where one or more components become zero. Specifically, we set

$$\varepsilon = \min\{x_i/y_i : y_i > 0\}.$$

For this value of ε the solution given by (2.16) is feasible and has at most $p - 1$ positive variables. Repeating this process if necessary, we can eliminate positive variables until we have a feasible solution with corresponding columns that are linearly independent. At that point Case 1 applies. ∎

Proof of (ii). Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be an optimal feasible solution and, as in the proof of (i) above, suppose there are exactly p positive variables $x_1, x_2, \ldots, x_p$. Again there are two cases; and Case 1, corresponding to linear independence, is exactly the same as before.

Case 2 also goes exactly the same as before, but it must be shown that for any ε the solution (2.16) is optimal. To show this, note that the value of the solution $\mathbf{x} - \varepsilon\mathbf{y}$ is

$$\mathbf{c}^T\mathbf{x} - \varepsilon\mathbf{c}^T\mathbf{y}. \tag{2.17}$$

For ε sufficiently small in magnitude, $\mathbf{x} - \varepsilon\mathbf{y}$ is a feasible solution for positive or negative values of ε. Thus we conclude that $\mathbf{c}^T\mathbf{y} = 0$. For, if $\mathbf{c}^T\mathbf{y} \neq 0$, an ε of small magnitude and proper sign could be determined so as to render (2.17) smaller than $\mathbf{c}^T\mathbf{x}$ while maintaining feasibility. This would violate the assumption of optimality of $\mathbf{x}$ and hence we must have $\mathbf{c}^T\mathbf{y} = 0$.

Having established that the new feasible solution with fewer positive components is also optimal, the remainder of the proof may be completed exactly as in part (i). ∎

This theorem reduces the task of solving a linear program to that of searching over basic feasible solutions. Since for a problem having n variables and m constraints there are at most

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}$$

basic solutions (corresponding to the number of ways of selecting m of n columns), there are only a finite number of possibilities. Thus the fundamental theorem yields an obvious, but terribly inefficient, finite search technique. By expanding upon the technique of proof as well as the statement of the fundamental theorem, the efficient simplex procedure is derived.

It should be noted that the proof of the fundamental theorem given above is of a simple algebraic character. In the next section the geometric interpretation of this theorem is explored in terms of the general theory of convex sets. Although the geometric interpretation is aesthetically pleasing and theoretically important, the reader should bear in mind, lest one be diverted by the somewhat more advanced arguments employed, the underlying elementary level of the fundamental theorem.

2.5 Relations to Convexity

Our development to this point, including the above proof of the fundamental theorem, has been based only on elementary properties of systems of linear equations. These results, however, have interesting interpretations in terms of the theory of convex sets that can lead not only to an alternative derivation of the fundamental theorem, but also to a clearer geometric understanding of the result. The main link between the algebraic and geometric theories is the formal relation between basic feasible solutions of linear inequalities in standard form and extreme points of polytopes. We establish this correspondence as follows. The reader is referred to Appendix B for a more complete summary of concepts related to convexity, but the definition of an extreme point is stated here.

Definition. A point $\mathbf{x}$ in a convex set C is said to be an *extreme point* of C if there are no two distinct points $\mathbf{x}_1$ and $\mathbf{x}_2$ in C such that $\mathbf{x} = \alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2$ for some $\alpha,\ 0 < \alpha < 1$.

An extreme point is thus a point that does not lie strictly within a line segment connecting two other points of the set. The extreme points of a triangle, for example, are its three vertices.

***Theorem* (Equivalence of Extreme Points and Basic Solutions).** *Let* $\mathbf{A}$ *be an* $m \times n$ *matrix of rank* m *and* $\mathbf{b}$ *an* m*-vector. Let* K *be the convex polytope consisting of all* n*-vectors* $\mathbf{x}$ *satisfying*

$$\mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}. \tag{2.18}$$

A vector $\mathbf{x}$ *is an extreme point of* K *if and only if* $\mathbf{x}$ *is a basic feasible solution to (2.18).*

Proof. Suppose first that $\mathbf{x} = (x_1,\ x_2,\ \ldots,\ x_m,\ 0, 0,\ \ldots,\ 0)$ is a basic feasible solution to (2.18). Then

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_m\mathbf{a}_m = \mathbf{b},$$

where $\mathbf{a}_1,\ \mathbf{a}_2,\ \ldots,\ \mathbf{a}_m$, the first m columns of $\mathbf{A}$, are linearly independent. Suppose that $\mathbf{x}$ could be expressed as a convex combination of two other points in K; say, $\mathbf{x} = \alpha\mathbf{y} + (1-\alpha)\mathbf{z},\ 0 < \alpha < 1,\ \mathbf{y} \neq \mathbf{z}$. Since all components of $\mathbf{x},\ \mathbf{y},\ \mathbf{z}$ are nonnegative and since $0 < \alpha < 1$, it follows immediately that the last $n-m$ components of $\mathbf{y}$ and $\mathbf{z}$ are zero. Thus, in particular, we have

$$y_1\mathbf{a}_1 + y_2\mathbf{a}_2 + \cdots + y_m\mathbf{a}_m = \mathbf{b}$$

and

$$z_1\mathbf{a}_1 + z_2\mathbf{a}_2 + \cdots + z_m\mathbf{a}_m = \mathbf{b}.$$

Since the vectors $\mathbf{a}_1,\ \mathbf{a}_2,\ \ldots,\ \mathbf{a}_m$ are linearly independent, however, it follows that $\mathbf{x} = \mathbf{y} = \mathbf{z}$ and hence $\mathbf{x}$ is an extreme point of K.

Conversely, assume that $\mathbf{x}$ is an extreme point of K. Let us assume that the nonzero components of $\mathbf{x}$ are the first k components. Then

$$x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \cdots + x_k\mathbf{a}_k = \mathbf{b},$$

with $x_i > 0,\ i = 1, 2, \ldots, k$. To show that $\mathbf{x}$ is a basic feasible solution it must be shown that the vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k$ are linearly independent. We do this by contradiction. Suppose $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k$ are linearly dependent. Then there is a nontrivial linear combination that is zero:

$$y_1\mathbf{a}_1 + y_2\mathbf{a}_2 + \cdots + y_k\mathbf{a}_k = 0.$$

Define the n-vector $\mathbf{y} = (y_1, y_2, \ldots, y_k, 0, 0, \ldots, 0)$. Since $x_i > 0,\ 1 \leqslant i \leqslant k$, it is possible to select ε such that

$$\mathbf{x} + \varepsilon\mathbf{y} \geqslant 0, \quad \mathbf{x} - \varepsilon\mathbf{y} \geqslant 0.$$

We then have $\mathbf{x} = \frac{1}{2}(\mathbf{x}+\varepsilon\mathbf{y}) + \frac{1}{2}(\mathbf{x}-\varepsilon\mathbf{y})$ which expresses $\mathbf{x}$ as a convex combination of two distinct vectors in K. This cannot occur, since $\mathbf{x}$ is an extreme point of K. Thus $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_k$ are linearly independent and $\mathbf{x}$ is a basic feasible solution. (Although if $k < m$, it is a degenerate basic feasible solution.) ∎

This correspondence between extreme points and basic feasible solutions enables us to prove certain geometric properties of the convex polytope K defining the constraint set of a linear programming problem.

Corollary 1. *If the convex set K corresponding to (2.18) is nonempty, it has at least one extreme point.*

Proof. This follows from the first part of the Fundamental Theorem and the Equivalence Theorem above. ∎

Corollary 2. *If there is a finite optimal solution to a linear programming problem, there is a finite optimal solution which is an extreme point of the constraint set.*

Corollary 3. *The constraint set K corresponding to (2.18) possesses at most a finite number of extreme points.*

Proof. There are obviously only a finite number of basic solutions obtained by selecting m basis vectors from the n columns of $\mathbf{A}$. The extreme points of K are a subset of these basic solutions. ∎

Finally, we come to the special case which occurs most frequently in practice and which in some sense is characteristic of well-formulated linear programs—the case where the constraint set K is nonempty and bounded. In this case we combine the results of the Equivalence Theorem and Corollary 3 above to obtain the following corollary.

Corollary 4. *If the convex polytope K corresponding to (2.18) is bounded, then K is a convex polyhedron, that is, K consists of points that are convex combinations of a finite number of points.*

Some of these results are illustrated by the following examples:

Example 1. Consider the constraint set in E^3 defined by

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ x_1 \geqslant 0,\ x_2 &\geqslant 0,\ x_3 \geqslant 0. \end{aligned}$$

This set is illustrated in Fig. 2.3. It has three extreme points, corresponding to the three basic solutions to $x_1 + x_2 + x_3 = 1$.

Example 2. Consider the constraint set in E^3 defined by

$$\begin{aligned} x_1 + x_2 + x_3 &= 1 \\ 2x_1 + 3x_2 \quad &= 1 \\ x_1 \geqslant 0,\ x_2 &\geqslant 0,\ x_3 \geqslant 0. \end{aligned}$$

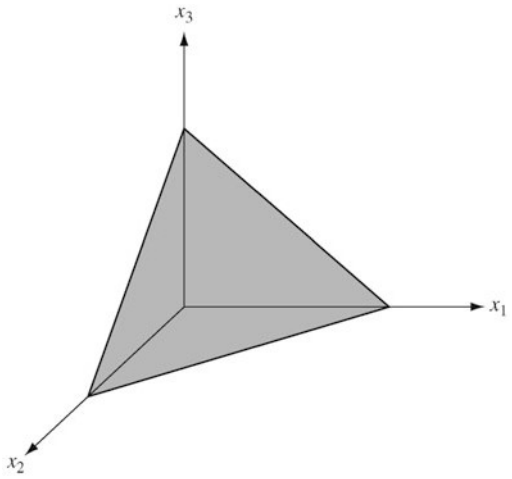

Fig. 2.3 Feasible set for Example 1

This set is illustrated in Fig. 2.4. It has two extreme points, corresponding to the two basic feasible solutions. Note that the system of equations itself has three basic solutions, $(2, -1, 0)$, $(1/2, 0, 1/2)$, $(0, 1/3, 2/3)$, the first of which is not feasible.

Example 3. Consider the constraint set in E^2 defined in terms of the inequalities

$$\begin{aligned} x_1 + \frac{8}{3}x_2 &\leqslant 4 \\ x_1 + \quad x_2 &\leqslant 2 \\ 2x_1 \qquad &\leqslant 3 \\ x_1 \geqslant 0, \quad x_2 &\geqslant 0. \end{aligned}$$

This set is illustrated in Fig. 2.5. We see by inspection that this set has five extreme points. In order to compare this example with our general results we must introduce slack variables to yield the equivalent set in E^5:

$$\begin{aligned} x_1 + \frac{8}{3}x_2 + x_3 \qquad\quad &= 4 \\ x_1 + x_2 \qquad + x_4 \qquad &= 2 \\ 2x_1 \qquad\quad + x_5 &= 3 \end{aligned}$$
$$x_1 \geqslant 0,\ x_2 \geqslant 0,\ x_3 \geqslant 0,\ x_4 \geqslant 0,\ x_5 \geqslant 0.$$

A basic solution for this system is obtained by setting any two variables to zero and solving for the remaining three. As indicated in Fig. 2.5, each edge of the figure corresponds to one variable being zero, and the extreme points are the points where two variables are zero.

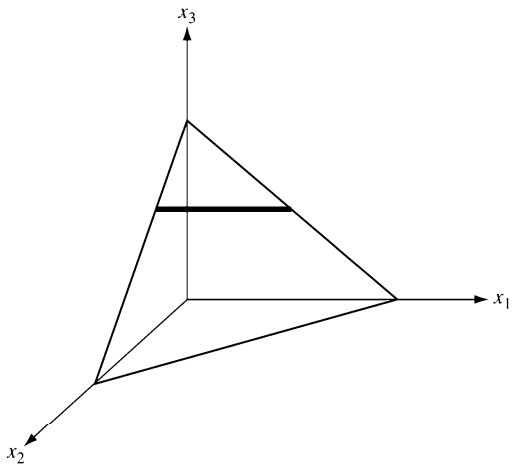

Fig. 2.4 Feasible set for Example 2

The last example illustrates that even when not expressed in standard form the extreme points of the set defined by the constraints of a linear program correspond to the possible solution points. This can be illustrated more directly by including the objective function in the figure as well. Suppose, for example, that in Example 3 the objective function to be minimized is $-2x_1 - x_2$. The set of points satisfying $-2x_1 - x_2 = z$ for fixed z is a line. As z varies, different parallel lines are obtained as shown in Fig. 2.6. The optimal value of the linear program is the smallest value of z for which the corresponding line has a point in common with the feasible set. It should be reasonably clear, at least in two dimensions, that the points of solution will always include an extreme point. In the figure this occurs at the point (3/2, 1/2) with $z = -7/2$.

2.6 Exercises

1. Convert the following problems to standard form:

$$\begin{aligned}
&\text{(a) minimize } x + 2y + 3z \\
&\quad \text{subject to } 2 \leqslant x + y \leqslant 3 \\
&\qquad\qquad 4 \leqslant x + z \leqslant 5 \\
&\qquad\qquad x \geqslant 0,\ y \geqslant 0,\ z \geqslant 0. \\
&\text{(b) minimize } x + y + z \\
&\quad \text{subject to } x + 2y + 3z = 10 \\
&\qquad\qquad x \geqslant 1,\ y \geqslant 2,\ z \geqslant 1.
\end{aligned}$$

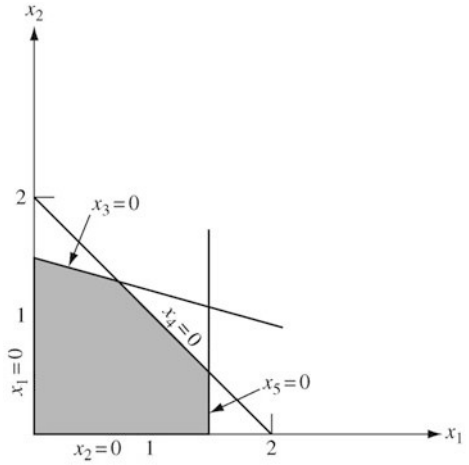

Fig. 2.5 Feasible set for Example 3

2. A manufacturer wishes to produce an alloy that is, by weight, 30 % metal A and 70 % metal B. Five alloys are available at various prices as indicated below:

Alloy	1	2	3	4	5
%A	10	25	50	75	95
% B	90	75	50	25	5
Price/lb	$ 5	$ 4	$ 3	$ 2	$ 1.50

The desired alloy will be produced by combining some of the other alloys. The manufacturer wishes to find the amounts of the various alloys needed and to determine the least expensive combination. Formulate this problem as a linear program.

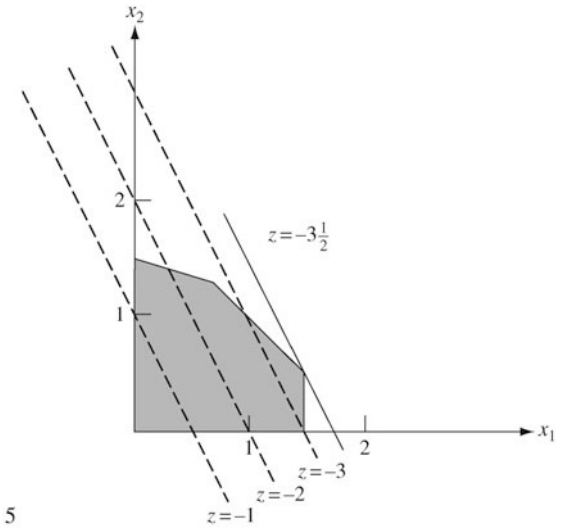

Fig. 2.6 Illustration of extreme point solution

3. An oil refinery has two sources of crude oil: a light crude that costs \$35/barrel and a heavy crude that costs \$30/barrel. The refinery produces gasoline, heating oil, and jet fuel from crude in the amounts per barrel indicated in the following table:

	Gasoline	Heating oil	Jet fuel
Light crude	0.3	0.2	0.3
Heavy crude	0.3	0.4	0.2

The refinery has contracted to supply 900,000 barrels of gasoline, 800,000 barrels of heating oil, and 500,000 barrels of jet fuel. The refinery wishes to find the amounts of light and heavy crude to purchase so as to be able to meet its obligations at minimum cost. Formulate this problem as a linear program.

4. A small firm specializes in making five types of spare automobile parts. Each part is first cast from iron in the casting shop and then sent to the finishing shop where holes are drilled, surfaces are turned, and edges are ground. The required worker-hours (per 100 units) for each of the parts of the two shops are shown below:

Part	1	2	3	4	5
Casting	2	1	3	3	1
Finishing	3	2	2	1	1

The profits from the parts are \$30, \$20, \$40, \$25, and \$10 (per 100 units), respectively. The capacities of the casting and finishing shops over the next month are 700 and 1,000 worker-hours, respectively. Formulate the problem of

determining the quantities of each spare part to be made during the month so as to maximize profit.

5. Convert the following problem to standard form and solve:

$$\begin{aligned} \text{maximize} \quad & x_1 + 4x_2 + x_3 \\ \text{subject to} \quad & 2x_1 - 2x_2 + x_3 = 4 \\ & x_1 \qquad\quad - x_3 = 1 \\ & x_2 \geqslant 0, \qquad x_3 \geqslant 0. \end{aligned}$$

6. A large textile firm has two manufacturing plants, two sources of raw material, and three market centers. The transportation costs between the sources and the plants and between the plants and the markets are as follows:

		Plant A	Plant B
Source	1	\$1/ton	\$1.50/ton
	2	\$2/ton	\$1.50/ton

		Market 1	Market 2	Market 3
Plant	A	\$4/ton	\$2/ton	\$1/ton
	B	\$3/ton	\$4/ton	\$2/ton

Ten tons are available from source 1 and 15 tons from source 2. The three market centers require 8 tons, 14 tons, and 3 tons. The plants have unlimited processing capacity.

(a) Formulate the problem of finding the shipping patterns from sources to plants to markets that minimizes the total transportation cost.
(b) Reduce the problem to a single standard transportation problem with two sources and three destinations. (*Hint*: Find minimum cost paths from sources to markets.)
(c) Suppose that plant A has a processing capacity of 8 tons, and plant B has a processing capacity of 7 tons. Show how to reduce the problem to two separate standard transportation problems.

7. A businessman is considering an investment project. The project has a lifetime of 4 years, with cash flows of −\$100,000, +\$50,000, +\$70,000, and +\$30,000 in each of the 4 years, respectively. At any time he may borrow funds at the rates of 12 %, 22 %, and 34 % (total) for 1, 2, or 3 periods, respectively. He may loan funds at 10 % per period. He calculates the *present value* of a project as the maximum amount of money he would pay now, to another party, for the project, assuming that he has no cash on hand and must borrow and lend to pay the other party and operate the project while maintaining a nonnegative cash

balance after all debts are paid. Formulate the project valuation problem in a linear programming framework.

8. Convert the following problem to a linear program in standard form:

$$\begin{aligned} &\text{minimize } |x| + |y| + |z| \\ &\text{subject to } x + y \leqslant 1 \\ &\qquad\qquad 2x + z = 3. \end{aligned}$$

9. A class of piecewise linear functions can be represented as $f(\mathbf{x}) = \text{Maximum } (\mathbf{c}_1^T\mathbf{x} + d_1,\ \mathbf{c}_2^T\mathbf{x} + d_2,\ \ldots,\ \mathbf{c}_p^T\mathbf{x} + d_p)$. For such a function f, consider the problem

$$\begin{aligned} &\text{minimize } f(\mathbf{x}) \\ &\text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geqslant \mathbf{0}. \end{aligned}$$

Show how to convert this problem to a linear programming problem.

10. A small computer manufacturing company forecasts the demand over the next n months to be $d_i,\ i = 1, 2,\ \ldots,\ n$. In any month it can produce r units, using *regular* production, at a cost of b dollars per unit. By using *overtime*, it can produce additional units at c dollars per unit, where $c > b$. The firm can store units from month to month at a cost of s dollars per unit per month. Formulate the problem of determining the production schedule that minimizes cost. (*Hint*: See Exercise 9.)
11. Discuss the situation of a linear program that has one or more columns of the $\mathbf{A}$ matrix equal to zero. Consider both the case where the corresponding variables are required to be nonnegative and the case where some are free.
12. Suppose that the matrix $\mathbf{A} = (\mathbf{a}_1,\ \mathbf{a}_2,\ \ldots,\ \mathbf{a}_n)$ has rank m, and that for some $p < m$, $\mathbf{a}_1,\ \mathbf{a}_2,\ \ldots,\ \mathbf{a}_p$ are linearly independent. Show that $m - p$ vectors from the remaining $n - p$ vectors can be adjoined to form a set of m linearly independent vectors.
13. Suppose that $\mathbf{x}$ is a feasible solution to the linear program (2.12), with $\mathbf{A}$ an $m \times n$ matrix of rank m. Show that there is a feasible solution $\mathbf{y}$ having the same value (that is, $\mathbf{c}^T\mathbf{y} = \mathbf{c}^T\mathbf{x}$) and having at most $m + 1$ positive components.
14. What are the basic solutions of Example 3, Sect. 2.5?
15. Let S be a convex set in E^n and S^* a convex set in E^m. Suppose $\mathbf{T}$ is an $m \times n$ matrix that establishes a one-to-one correspondence between S and S^*, i.e., for every $\mathbf{s} \in S$ there is $\mathbf{s}^* \in S^*$ such that $\mathbf{T}\mathbf{s} = \mathbf{s}^*$, and for every $\mathbf{s}^* \in S^*$ there is a single $\mathbf{s} \in S$ such that $\mathbf{T}\mathbf{s} = \mathbf{s}^*$. Show that there is a one-to-one correspondence between extreme points of S and S^*.
16. Consider the two linear programming problems in Example 1, Sect. 2.1, one in E^n and the other in E^{n+m}. Show that there is a one-to-one correspondence between extreme points of these two problems.

References

2.1–2.4 The approach taken in this chapter, which is continued in the next, is the more or less standard approach to linear programming as presented in, for example, Dantzig [D6], Hadley [H1], Gass [G4], Simonnard [S6], Murty [M11], and Gale [G2]. Also see Bazaraa, Jarvis, and H. F. Sherali [B6], Bertsimas and Tsitsiklis [B13], Cottle, [C6], Dantzig and Thapa [D9, D10], Nash and Sofer [N1], Saigal [S1], and Vanderbei [V3].

2.5 An excellent discussion of this type can be found in Simonnard [S6].