Introduction to LINEAR OPTIMIZATION

Dimitris Bertsimas
John N. Tsitsiklis

Introduction
to Linear Optimization

ATHENA SCIENTIFIC SERIES
IN OPTIMIZATION AND NEURAL COMPUTATION

1. Dynamic Programming and Optimal Control, Vols. I and II, by Dimitri P. Bertsekas, 1995.
2. Nonlinear Programming, by Dimitri P. Bertsekas, 1995.
3. Neuro-Dynamic Programming, by Dimitri P. Bertsekas and John N. Tsitsiklis, 1996.
4. Constrained Optimization and Lagrange Multiplier Methods, by Dimitri P. Bertsekas, 1996.
5. Stochastic Optimal Control: The Discrete-Time Case, by Dimitri P. Bertsekas and Steven E. Shreve, 1996.
6. Introduction to Linear Optimization, by Dimitris Bertsimas and John N. Tsitsiklis, 1997.

Introduction to Linear Optimization

Dimitris Bertsimas
John N. Tsitsiklis

Massachusetts Institute of Technology

Athena Scientific, Belmont, Massachusetts

Athena Scientific
Post Office Box 391
Belmont, Mass. 02178-9998
U.S.A.

Email: athenasc@world.std.com
WWW information and orders: http://world.std.com/~athenasc/

Cover Design: *Ann Gallager*

Publisher's Cataloging-in-Publication Data

Bertsimas, Dimitris, Tsitsiklis, John N.
Introduction to Linear Optimization
Includes bibliographical references and index
1. Linear programming. 2. Mathematical optimization.
3. Integer programming. I. Title.
T57.74.B465 1997 519.7 96-78786

ISBN 1-886529-19-1

To Georgia,
and to George Michael, who left us so early
To Alexandra and Melina

Contents

Preface

The purpose of this book is to provide a unified, insightful, and modern treatment of linear optimization, that is, linear programming, network flow problems, and discrete linear optimization. We discuss both classical topics, as well as the state of the art. We give special attention to theory, but also cover applications and present case studies. Our main objective is to help the reader become a sophisticated practitioner of (linear) optimization, or a researcher. More specifically, we wish to develop the ability to formulate fairly complex optimization problems, provide an appreciation of the main classes of problems that are practically solvable, describe the available solution methods, and build an understanding of the qualitative properties of the solutions they provide.

Our general philosophy is that insight matters most. For the subject matter of this book, this necessarily requires a geometric view. On the other hand, problems are solved by algorithms, and these can only be described algebraically. Hence, our focus is on the beautiful interplay between algebra and geometry. We build understanding using figures and geometric arguments, and then translate ideas into algebraic formulas and algorithms. Given enough time, we expect that the reader will develop the ability to pass from one domain to the other without much effort.

Another of our objectives is to be comprehensive, but economical. We have made an effort to cover and highlight all of the principal ideas in this field. However, we have not tried to be encyclopedic, or to discuss every possible detail relevant to a particular algorithm. Our premise is that once mature understanding of the basic principles is in place, further details can be acquired by the reader with little additional effort.

Our last objective is to bring the reader up to date with respect to the state of the art. This is especially true in our treatment of interior point methods, large scale optimization, and the presentation of case studies that stretch the limits of currently available algorithms and computers.

The success of any optimization methodology hinges on its ability to deal with large and important problems. In that sense, the last chapter, on the art of linear optimization, is a critical part of this book. It will, we hope, convince the reader that progress on challenging problems requires both problem specific insight, as well as a deeper understanding of the underlying theory.

In any book dealing with linear programming, there are some important choices to be made regarding the treatment of the simplex method. Traditionally, the simplex method is developed in terms of the full simplex tableau, which tends to become the central topic. We have found that the full simplex tableau is a useful device for working out numerical examples. But other than that, we have tried not to overemphasize its importance.

Let us also mention another departure from many other textbooks. Introductory treatments often focus on standard form problems, which is sufficient for the purposes of the simplex method. On the other hand, this approach often leaves the reader wondering whether certain properties are generally true, and can hinder the deeper understanding of the subject. We depart from this tradition: we consider the general form of linear programming problems and define key concepts (e.g., extreme points) within this context. (Of course, when it comes to algorithms, we often have to specialize to the standard form.) In the same spirit, we separate the structural understanding of linear programming from the particulars of the simplex method. For example, we include a derivation of duality theory that does not rely on the simplex method.

Finally, this book contains a treatment of several important topics that are not commonly covered. These include a discussion of the column geometry and of the insights it provides into the efficiency of the simplex method, the connection between duality and the pricing of financial assets, a unified view of delayed column generation and cutting plane methods, stochastic programming and Benders decomposition, the auction algorithm for the assignment problem, certain theoretical implications of the ellipsoid algorithm, a thorough treatment of interior point methods, and a whole chapter on the practice of linear optimization. There are also several noteworthy topics that are covered in the exercises, such as Leontief systems, strict complementarity, options pricing, von Neumann's algorithm, submodular function minimization, and bounds for a number of integer programming problems.

Here is a chapter by chapter description of the book.

Chapter 1: Introduces the linear programming problem, together with a number of examples, and provides some background material on linear algebra.

Chapter 2: Deals with the basic geometric properties of polyhedra, focusing on the definition and the existence of extreme points, and emphasizing the interplay betwen the geometric and the algebraic viewpoints.

Chapter 3: Contains more or less the classical material associated with the simplex method, as well as a discussion of the column geometry. It starts with a high-level and geometrically motivated derivation of the simplex method. It then introduces the revised simplex method, and concludes with the simplex tableau. The usual topics of Phase I and anticycling are

also covered.

Chapter 4: It is a comprehensive treatment of linear programming duality. The duality theorem is first obtained as a corollary of the simplex method. A more abstract derivation is also provided, based on the separating hyperplane theorem, which is developed from first principles. It ends with a deeper look into the geometry of polyhedra.

Chapter 5: Discusses sensitivity analysis, that is, the dependence of solutions and the optimal cost on the problem data, including parametric programming. It also develops a characterization of dual optimal solutions as subgradients of a suitably defined optimal cost function.

Chapter 6: Presents the complementary ideas of delayed column generation and cutting planes. These methods are first developed at a high level, and are then made concrete by discussing the cutting stock problem, Dantzig-Wolfe decomposition, stochastic programming, and Benders decomposition.

Chapter 7: Provides a comprehensive review of the principal results and methods for the different variants of the network flow problem. It contains representatives from all major types of algorithms: primal descent (the simplex method), dual ascent (the primal-dual method), and approximate dual ascent (the auction algorithm). The focus is on the major algorithmic ideas, rather than on the refinements that can lead to better complexity estimates.

Chapter 8: Includes a discussion of complexity, a development of the ellipsoid method, and a proof of the polynomiality of linear programming. It also discusses the equivalence of separation and optimization, and provides examples where the ellipsoid algorithm can be used to derive polynomial time results for problems involving an exponential number of constraints.

Chapter 9: Contains an overview of all major classes of interior point methods, including affine scaling, potential reduction, and path following (both primal and primal-dual) methods. It includes a discussion of the underlying geometric ideas and computational issues, as well as convergence proofs and complexity analysis.

Chapter 10: Introduces integer programming formulations of discrete optimization problems. It provides a number of examples, as well as some intuition as to what constitutes a "strong" formulation.

Chapter 11: Covers the major classes of integer programming algorithms, including exact methods (branch and bound, cutting planes, dynamic programming), approximation algorithms, and heuristic methods (local search and simulated annealing). It also introduces a duality theory for integer programming.

Chapter 12: Deals with the art in linear optimization, i.e., the process

of modeling, exploiting problem structure, and fine tuning of optimization algorithms. We discuss the relative performance of interior point methods and different variants of the simplex method, in a realistic large scale setting. We also give some indication of the size of problems that can be currently solved.

An important theme that runs through several chapters is the modeling, complexity, and algorithms for problems with an exponential number constraints. We discuss modeling in Section 10.3, complexity in Section 8.5, algorithmic approaches in Chapter 6 and 8.5, and we conclude with a case study in Section 12.5.

There is a fair number of exercises that are given at the end of each chapter. Most of them are intended to deepen the understanding of the subject, or to explore extensions of the theory in the text, as opposed to routine drills. However, several numerical exercises are also included. Starred exercises are supposed to be fairly hard. A solutions manual for qualified instructors can be obtained from the authors.

We have made a special effort to keep the text as modular as possible, allowing the reader to omit certain topics without loss of continuity. For example, much of the material in Chapters 5 and 6 is rarely used in the rest of the book. Furthermore, in Chapter 7 (on network flow problems), a reader who has gone through the problem formulation (Sections 7.1-7.2) can immediately move to any later section in that chapter. Also, the interior point algorithms of Chapter 9 are not used later, with the exception of some of the applications in Chapter 12. Even within the core chapters (Chapters 1-4), there are many sections that can be skipped during a first reading. Some sections have been marked with a star indicating that they contain somewhat more advanced material that is not usually covered in an introductory course.

The book was developed while we took turns teaching a first-year graduate course at M.I.T., for students in engineering and operations research. The only prerequisite is a working knowledge of linear algebra. In fact, it is only a small subset of linear algebra that is needed (e.g., the concepts of subspaces, linear independence, and the rank of a matrix). However, these elementary tools are sometimes used in subtle ways, and some mathematical maturity on the part of the reader can lead to a better appreciation of the subject.

The book can be used to teach several different types of courses. The first two suggestions below are one-semester variants that we have tried at M.I.T., but there are also other meaningful alternatives, depending on the students' background and the course's objectives.

(a) Cover most of Chapters 1-7, and if time permits, cover a small number of topics from Chapters 9-12.

(b) An alternative could be the same as above, except that interior point

algorithms (Chapter 9) are fully covered, replacing network flow problems (Chapter 7).

(c) A broad overview course can be constructed by concentrating on the easier material in most of the chapters. The core of such a course could consist of Chapter 1, Sections 2.1-2.4, 3.1-3.5, 4.1-4.3, 5.1, 7.1-7.3, 9.1, 10.1, some of the easier material in Chapter 11, and an application from Chapter 12.

(d) Finally, the book is also suitable for a half-course on integer programming, based on parts of Chapters 1 and 8, as well as Chapters 10-12.

There is a truly large literature on linear optimization, and we make no attempt to provide a comprehensive bibliography. To a great extent, the sources that we cite are either original references of historical interest, or recent texts where additional information can be found. For those topics, however, that touch upon current research, we also provide pointers to recent journal articles.

We would like to express our thanks to a number of individuals. We are grateful to our colleagues Dimitri Bertsekas and Rob Freund, for many discussions on the subjects in this book, as well as for reading parts of the manuscript. Several of our students, colleagues, and friends have contributed by reading parts of the manuscript, providing critical comments, and working on the exercises: Jim Christodouleas, Thalia Chryssikou, Austin Frakt, David Gamarnik, Leon Hsu, Spyros Kontogiorgis, Peter Marbach, Gina Mourtzinou, Yannis Paschalidis, Georgia Perakis, Lakis Polymenakos, Jay Sethuraman, Sarah Stock, Paul Tseng, and Ben Van Roy. But mostly, we are grateful to our families for their patience, love, and support in the course of this long project.

Dimitris Bertsimas
John N. Tsitsiklis
Cambridge, January 1997

Chapter 1

Introduction

Contents

In this chapter, we introduce *linear programming*, the problem of minimizing a linear cost function subject to linear equality and inequality constraints. We consider a few equivalent forms and then present a number of examples to illustrate the applicability of linear programming to a wide variety of contexts. We also solve a few simple examples and obtain some basic geometric intuition on the nature of the problem. The chapter ends with a review of linear algebra and of the conventions used in describing the computational requirements (operation count) of algorithms.

1.1 Variants of the linear programming problem

In this section, we pose the linear programming problem, discuss a few special forms that it takes, and establish some standard notation that we will be using. Rather than starting abstractly, we first state a concrete example, which is meant to facilitate understanding of the formal definition that will follow. The example we give is devoid of any interpretation. Later on, in Section 1.2, we will have ample opportunity to develop examples that arise in practical settings.

Example 1.1 The following is a linear programming problem:

$$\begin{array}{lrcrcrcl} \text{minimize} & 2x_1 & - & x_2 & + & 4x_3 & & \\ \text{subject to} & x_1 & + & x_2 & & & + \ x_4 & \leq 2 \\ & & & 3x_2 & - & x_3 & & = 5 \\ & & & & & x_3 & + \ x_4 & \geq 3 \\ & x_1 & & & & & & \geq 0 \\ & & & & & x_3 & & \leq 0. \end{array}$$

Here x_1, x_2, x_3, and x_4 are variables whose values are to be chosen to minimize the linear cost function $2x_1 - x_2 + 4x_3$, subject to a set of linear equality and inequality constraints. Some of these constraints, such as $x_1 \geq 0$ and $x_3 \leq 0$, amount to simple restrictions on the sign of certain variables. The remaining constraints are of the form $\mathbf{a}'\mathbf{x} \leq b$, $\mathbf{a}'\mathbf{x} = b$, or $\mathbf{a}'\mathbf{x} \geq b$, where $\mathbf{a} = (a_1, a_2, a_3, a_4)$ is a given vector[1], $\mathbf{x} = (x_1, x_2, x_3, x_4)$ is the vector of decision variables, $\mathbf{a}'\mathbf{x}$ is their inner product $\sum_{i=1}^{4} a_i x_i$, and b is a given scalar. For example, in the first constraint, we have $\mathbf{a} = (1, 1, 0, 1)$ and $b = 2$.

We now generalize. In a *general* linear programming problem, we are given a cost vector $\mathbf{c} = (c_1, \ldots, c_n)$ and we seek to minimize a linear cost function $\mathbf{c}'\mathbf{x} = \sum_{i=1}^{n} c_i x_i$ over all n-dimensional vectors $\mathbf{x} = (x_1, \ldots, x_n)$,

[1] As discussed further in Section 1.5, all vectors are assumed to be column vectors, and are treated as such in matrix-vector products. Row vectors are indicated as transposes of (column) vectors. However, whenever we refer to a vector $\mathbf{x}$ inside the text, we use the more economical notation $\mathbf{x} = (x_1, \ldots, x_n)$, even though $\mathbf{x}$ is a column vector. The reader who is unfamiliar with our notation may wish to consult Section 1.5 before continuing.

subject to a set of linear equality and inequality constraints. In particular, let M_1, M_2, M_3 be some finite index sets, and suppose that for every i in any one of these sets, we are given an n-dimensional vector $\mathbf{a}_i$ and a scalar b_i, that will be used to form the ith constraint. Let also N_1 and N_2 be subsets of $\{1, \ldots, n\}$ that indicate which variables x_j are constrained to be nonnegative or nonpositive, respectively. We then consider the problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{a}_i'\mathbf{x} \geq b_i, \qquad i \in M_1, \\ & \mathbf{a}_i'\mathbf{x} \leq b_i, \qquad i \in M_2, \\ & \mathbf{a}_i'\mathbf{x} = b_i, \qquad i \in M_3, \\ & x_j \geq 0, \qquad j \in N_1, \\ & x_j \leq 0, \qquad j \in N_2. \end{aligned} \tag{1.1}$$

The variables $x_1, \ldots, x_n$ are called *decision variables*, and a vector $\mathbf{x}$ satisfying all of the constraints is called a *feasible solution* or *feasible vector*. The set of all feasible solutions is called the *feasible set* or *feasible region*. If j is in neither N_1 nor N_2, there are no restrictions on the sign of x_j, in which case we say that x_j is a *free* or *unrestricted* variable. The function $\mathbf{c}'\mathbf{x}$ is called the *objective function* or *cost function*. A feasible solution $\mathbf{x}^*$ that minimizes the objective function (that is, $\mathbf{c}'\mathbf{x}^* \leq \mathbf{c}'\mathbf{x}$, for all feasible $\mathbf{x}$) is called an *optimal feasible solution* or, simply, an *optimal solution*. The value of $\mathbf{c}'\mathbf{x}^*$ is then called the *optimal cost*. On the other hand, if for every real number K we can find a feasible solution $\mathbf{x}$ whose cost is less than K, we say that the optimal cost is $-\infty$ or that the cost is *unbounded below*. (Sometimes, we will abuse terminology and say that the problem is *unbounded*.) We finally note that there is no need to study maximization problems separately, because maximizing $\mathbf{c}'\mathbf{x}$ is equivalent to minimizing the linear cost function $-\mathbf{c}'\mathbf{x}$.

An equality constraint $\mathbf{a}_i'\mathbf{x} = b_i$ is equivalent to the two constraints $\mathbf{a}_i'\mathbf{x} \leq b_i$ and $\mathbf{a}_i'\mathbf{x} \geq b_i$. In addition, any constraint of the form $\mathbf{a}_i'\mathbf{x} \leq b_i$ can be rewritten as $(-\mathbf{a}_i)'\mathbf{x} \geq -b_i$. Finally, constraints of the form $x_j \geq 0$ or $x_j \leq 0$ are special cases of constraints of the form $\mathbf{a}_i'\mathbf{x} \geq b_i$, where $\mathbf{a}_i$ is a unit vector and $b_i = 0$. We conclude that the feasible set in a general linear programming problem can be expressed exclusively in terms of inequality constraints of the form $\mathbf{a}_i'\mathbf{x} \geq b_i$. Suppose that there is a total of m such constraints, indexed by $i = 1, \ldots, m$, let $\mathbf{b} = (b_1, \ldots, b_m)$, and let $\mathbf{A}$ be the $m \times n$ matrix whose rows are the row vectors $\mathbf{a}_1', \ldots, \mathbf{a}_m'$, that is,

$$\mathbf{A} = \begin{bmatrix} — & \mathbf{a}_1' & — \\ & \vdots & \\ — & \mathbf{a}_m' & — \end{bmatrix}.$$

Then, the constraints $\mathbf{a}_i'\mathbf{x} \geq b_i$, $i = 1, \ldots, m$, can be expressed compactly in the form $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, and the linear programming problem can be written

as

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b}. \end{array} \tag{1.2}$$

Inequalities such as $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ will always be interpreted componentwise; that is, for every i, the ith component of the vector $\mathbf{A}\mathbf{x}$, which is $\mathbf{a}_i'\mathbf{x}$, is greater than or equal to the ith component b_i of the vector $\mathbf{b}$.

Example 1.2 The linear programming problem in Example 1.1 can be rewritten as

$$\begin{array}{lrcrcrcrcl} \text{minimize} & 2x_1 & - & x_2 & + & 4x_3 & & & & \\ \text{subject to} & -x_1 & - & x_2 & & & - & x_4 & \geq & -2 \\ & & & 3x_2 & - & x_3 & & & \geq & 5 \\ & & - & 3x_2 & + & x_3 & & & \geq & -5 \\ & & & & & x_3 & + & x_4 & \geq & 3 \\ & x_1 & & & & & & & \geq & 0 \\ & & & & - & x_3 & & & \geq & 0, \end{array}$$

which is of the same form as the problem (1.2), with $\mathbf{c} = (2, -1, 4, 0)$,

$$\mathbf{A} = \begin{bmatrix} -1 & -1 & 0 & -1 \\ 0 & 3 & -1 & 0 \\ 0 & -3 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix},$$

and $\mathbf{b} = (-2, 5, -5, 3, 0, 0)$.

Standard form problems

A linear programming problem of the form

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \tag{1.3}$$

is said to be in *standard form*. We provide an interpretation of problems in standard form. Suppose that $\mathbf{x}$ has dimension n and let $\mathbf{A}_1, \ldots, \mathbf{A}_n$ be the columns of $\mathbf{A}$. Then, the constraint $\mathbf{A}\mathbf{x} = \mathbf{b}$ can be written in the form

$$\sum_{i=1}^{n} \mathbf{A}_i x_i = \mathbf{b}.$$

Intuitively, there are n available resource vectors $\mathbf{A}_1, \ldots, \mathbf{A}_n$, and a target vector $\mathbf{b}$. We wish to "synthesize" the target vector $\mathbf{b}$ by using a nonnegative amount x_i of each resource vector $\mathbf{A}_i$, while minimizing the cost $\sum_{i=1}^{n} c_i x_i$, where c_i is the unit cost of the ith resource. The following is a more concrete example.

Example 1.3 (The diet problem) Suppose that there are n different foods and m different nutrients, and that we are given the following table with the nutritional content of a unit of each food:

	food 1	$\cdots$	food n
nutrient 1	a_{11}	$\cdots$	a_{1n}
$\vdots$	$\vdots$		$\vdots$
nutrient m	a_{m1}	$\cdots$	a_{mn}

Let $\mathbf{A}$ be the $m \times n$ matrix with entries a_{ij}. Note that the jth column $\mathbf{A}_j$ of this matrix represents the nutritional content of the jth food. Let $\mathbf{b}$ be a vector with the requirements of an ideal diet or, equivalently, a specification of the nutritional contents of an "ideal food." We then interpret the standard form problem as the problem of mixing nonnegative quantities x_i of the available foods, to synthesize the ideal food at minimal cost. In a variant of this problem, the vector $\mathbf{b}$ specifies the *minimal* requirements of an adequate diet; in that case, the constraints $\mathbf{Ax} = \mathbf{b}$ are replaced by $\mathbf{Ax} \geq \mathbf{b}$, and the problem is not in standard form.

Reduction to standard form

As argued earlier, any linear programming problem, including the standard form problem (1.3), is a special case of the general form (1.1). We now argue that the converse is also true and that a general linear programming problem can be transformed into an equivalent problem in standard form. Here, when we say that the two problems are equivalent, we mean that given a feasible solution to one problem, we can construct a feasible solution to the other, with the same cost. In particular, the two problems have the same optimal cost and given an optimal solution to one problem, we can construct an optimal solution to the other. The problem transformation we have in mind involves two steps:

(a) *Elimination of free variables:* Given an unrestricted variable x_j in a problem in general form, we replace it by $x_j^+ - x_j^-$, where x_j^+ and x_j^- are new variables on which we impose the sign constraints $x_j^+ \geq 0$ and $x_j^- \geq 0$. The underlying idea is that any real number can be written as the difference of two nonnegative numbers.

(b) *Elimination of inequality constraints:* Given an inequality constraint of the form

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i,$$

we introduce a new variable s_i and the standard form constraints

$$\sum_{j=1}^{n} a_{ij}x_j + s_i = b_i,$$
$$s_i \geq 0.$$

Such a variable s_i is called a *slack* variable. Similarly, an inequality constraint $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ can be put in standard form by introducing a *surplus* variable s_i and the constraints $\sum_{j=1}^{n} a_{ij}x_j - s_i = b_i,\ s_i \geq 0$.

We conclude that a general problem can be brought into standard form and, therefore, we only need to develop methods that are capable of solving standard form problems.

Example 1.4 The problem

$$\begin{array}{lrcl} \text{minimize} & 2x_1 + 4x_2 & & \\ \text{subject to} & x_1 + x_2 & \geq & 3 \\ & 3x_1 + 2x_2 & = & 14 \\ & x_1 & \geq & 0, \end{array}$$

is equivalent to the standard form problem

$$\begin{array}{lrcl} \text{minimize} & 2x_1 + 4x_2^+ - 4x_2^- & & \\ \text{subject to} & x_1 + x_2^+ - x_2^- - x_3 & = & 3 \\ & 3x_1 + 2x_2^+ - 2x_2^- & = & 14 \\ & x_1, x_2^+, x_2^-, x_3 \geq 0. & & \end{array}$$

For example, given the feasible solution $(x_1, x_2) = (6, -2)$ to the original problem, we obtain the feasible solution $(x_1, x_2^+, x_2^-, x_3) = (6, 0, 2, 1)$ to the standard form problem, which has the same cost. Conversely, given the feasible solution $(x_1, x_2^+, x_2^-, x_3) = (8, 1, 6, 0)$ to the standard form problem, we obtain the feasible solution $(x_1, x_2) = (8, -5)$ to the original problem with the same cost.

In the sequel, we will often use the general form $\mathbf{Ax} \geq \mathbf{b}$ to develop the theory of linear programming. However, when it comes to algorithms, and especially the simplex and interior point methods, we will be focusing on the standard form $\mathbf{Ax} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, which is computationally more convenient.

1.2 Examples of linear programming problems

In this section, we discuss a number of examples of linear programming problems. One of our purposes is to indicate the vast range of situations to which linear programming can be applied. Another purpose is to develop some familiarity with the art of constructing mathematical formulations of loosely defined optimization problems.

A production problem

A firm produces n different goods using m different raw materials. Let b_i, $i = 1, \ldots, m$, be the available amount of the ith raw material. The jth good, $j = 1, \ldots, n$, requires a_{ij} units of the ith material and results in a revenue of c_j per unit produced. The firm faces the problem of deciding how much of each good to produce in order to maximize its total revenue.

In this example, the choice of the decision variables is simple. Let x_j, $j = 1, \ldots, n$, be the amount of the jth good. Then, the problem facing the firm can be formulated as follows:

$$\begin{array}{lll} \text{maximize} & c_1x_1 + \cdots + c_nx_n & \\ \text{subject to} & a_{i1}x_1 + \cdots + a_{in}x_n \leq b_i, & i = 1, \ldots, m, \\ & x_j \geq 0, & j = 1, \ldots, n. \end{array}$$

Production planning by a computer manufacturer

The example that we consider here is a problem that Digital Equipment Corporation (DEC) had faced in the fourth quarter of 1988. It illustrates the complexities and uncertainties of real world applications, as well as the usefulness of mathematical modeling for making important strategic decisions.

In the second quarter of 1988, DEC introduced a new family of (single CPU) computer systems and workstations: GP-1, GP-2, and GP-3, which are general purpose computer systems with different memory, disk storage, and expansion capabilities, as well as WS-1 and WS-2, which are workstations. In Table 1.1, we list the models, the list prices, the average disk usage per system, and the memory usage. For example, GP-1 uses four 256K memory boards, and 3 out of every 10 units are produced with a disk drive.

System	Price	# disk drives	# 256K boards
GP-1	\$60,000	0.3	4
GP-2	\$40,000	1.7	2
GP-3	\$30,000	0	2
WS-1	\$30,000	1.4	2
WS-2	\$15,000	0	1

Table 1.1: Features of the five different DEC systems.

Shipments of this new family of products started in the third quarter and ramped slowly during the fourth quarter. The following difficulties were anticipated for the next quarter:

(a) The in-house supplier of CPUs could provide at most 7,000 units, due to debugging problems.

(b) The supply of disk drives was uncertain and was estimated by the manufacturer to be in the range of 3,000 to 7,000 units.

(c) The supply of 256K memory boards was also limited in the range of 8,000 to 16,000 units.

On the demand side, the marketing department established that the maximum demand for the first quarter of 1989 would be 1,800 for GP-1 systems, 300 for GP-3 systems, 3,800 systems for the whole GP family, and 3,200 systems for the WS family. Included in these projections were 500 orders for GP-2, 500 orders for WS-1, and 400 orders for WS-2 that had already been received and had to be fulfilled in the next quarter.

In the previous quarters, in order to address the disk drive shortage, DEC had produced GP-1, GP-3, and WS-2 with no disk drive (although 3 out of 10 customers for GP-1 systems wanted a disk drive), and GP-2, WS-1 with one disk drive. We refer to this way of configuring the systems as the constrained mode of production.

In addition, DEC could address the shortage of 256K memory boards by using two alternative boards, instead of four 256K memory boards, in the GP-1 system. DEC could provide 4,000 alternative boards for the next quarter.

It was clear to the manufacturing staff that the problem had become complex, as revenue, profitability, and customer satisfaction were at risk. The following decisions needed to be made:

(a) The production plan for the first quarter of 1989.

(b) Concerning disk drive usage, should DEC continue to manufacture products in the constrained mode, or should it plan to satisfy customer preferences?

(c) Concerning memory boards, should DEC use alternative memory boards for its GP-1 systems?

(d) A final decision that had to be made was related to tradeoffs between shortages of disk drives and of 256K memory boards. The manufacturing staff would like to concentrate their efforts on either decreasing the shortage of disks or decreasing the shortage of 256K memory boards. Hence, they would like to know which alternative would have a larger effect on revenue.

In order to model the problem that DEC faced, we introduce variables x_1, x_2, x_3, x_4, x_5, that represent the number (in thousands) of GP-1, GP-2, GP-3, WS-1, and WS-2 systems, respectively, to be produced in the next quarter. Strictly speaking, since $1000x_i$ stands for number of units, it must be an integer. This can be accomplished by truncating each x_i after the third decimal point; given the size of the demand and the size of the

variables x_i, this has a negligible effect and the integrality constraint on $1000x_i$ can be ignored.

DEC had to make two distinct decisions: whether to use the constrained mode of production regarding disk drive usage, and whether to use alternative memory boards for the GP-1 system. As a result, there are four different combinations of possible choices.

We first develop a model for the case where alternative memory boards are not used and the constrained mode of production of disk drives is selected. The problem can be formulated as follows:

$$\text{maximize} \quad 60x_1 + 40x_2 + 30x_3 + 30x_4 + 15x_5 \qquad \text{(total revenue)}$$

subject to the following constraints:

$$\begin{aligned}
x_1 + x_2 + x_3 + x_4 + x_5 &\leq 7 && \text{(CPU availability)}\\
4x_1 + 2x_2 + 2x_3 + 2x_4 + x_5 &\leq 8 && \text{(256K availability)}\\
x_2 + x_4 &\leq 3 && \text{(disk drive availability)}\\
x_1 &\leq 1.8 && \text{(max demand for GP-1)}\\
x_3 &\leq 0.3 && \text{(max demand for GP-3)}\\
x_1 + x_2 + x_3 &\leq 3.8 && \text{(max demand for GP)}\\
x_4 + x_5 &\leq 3.2 && \text{(max demand for WS)}\\
x_2 &\geq 0.5 && \text{(min demand for GP-2)}\\
x_4 &\geq 0.5 && \text{(min demand for WS-1)}\\
x_5 &\geq 0.4 && \text{(min demand for WS-2)}\\
x_1, x_2, x_3, x_4, x_5 &\geq 0. &&
\end{aligned}$$

Notice that the objective function is in millions of dollars. In some respects, this is a pessimistic formulation, because the 256K memory and disk drive availability were set to 8 and 3, respectively, which is the lowest value in the range that was estimated. It is actually of interest to determine the solution to this problem as the 256K memory availability ranges from 8 to 16, and the disk drive availability ranges from 3 to 7, because this provides valuable information on the sensitivity of the optimal solution on availability. In another respect, the formulation is optimistic because, for example, it assumes that the revenue from GP-1 systems is $60x_1$ for any $x_1 \leq 1.8$, even though a demand for 1,800 GP-1 systems is not guaranteed.

In order to accommodate the other three choices that DEC had, some of the problem constraints have to be modified, as follows. If we use the unconstrained mode of production for disk drives, the constraint $x_2 + x_4 \leq 3$ is replaced by

$$0.3x_1 + 1.7x_2 + 1.4x_4 \leq 3.$$

Furthermore, if we wish to use alternative memory boards in GP-1 systems, we replace the constraint $4x_1 + 2x_2 + 2x_3 + 2x_4 + x_5 \leq 8$ by the two

constraints

$$\begin{aligned} 2x_1 &\leq 4, \\ 2x_2 + 2x_3 + 2x_4 + x_5 &\leq 8. \end{aligned}$$

The four combinations of choices lead to four different linear programming problems, each of which needs to be solved for a variety of parameter values because, as discussed earlier, the right-hand side of some of the constraints is only known to lie within a certain range. Methods for solving linear programming problems, when certain parameters are allowed to vary, will be studied in Chapter 5, where this case study is revisited.

Multiperiod planning of electric power capacity

A state wants to plan its electricity capacity for the next T years. The state has a forecast of d_t megawatts, presumed accurate, of the demand for electricity during year $t = 1, \ldots, T$. The existing capacity, which is in oil-fired plants, that will not be retired and will be available during year t, is e_t. There are two alternatives for expanding electric capacity: coal-fired or nuclear power plants. There is a capital cost of c_t per megawatt of coal-fired capacity that becomes operational at the beginning of year t. The corresponding capital cost for nuclear power plants is n_t. For various political and safety reasons, it has been decided that no more than 20% of the total capacity should ever be nuclear. Coal plants last for 20 years, while nuclear plants last for 15 years. A least cost capacity expansion plan is desired.

The first step in formulating this problem as a linear programming problem is to define the decision variables. Let x_t and y_t be the amount of coal (respectively, nuclear) capacity brought on line at the beginning of year t. Let w_t and z_t be the total coal (respectively, nuclear) capacity available in year t. The cost of a capacity expansion plan is therefore,

$$\sum_{t=1}^{T} (c_t x_t + n_t y_t).$$

Since coal-fired plants last for 20 years, we have

$$w_t = \sum_{s=\max\{1,t-19\}}^{t} x_s, \qquad t = 1, \ldots, T.$$

Similarly, for nuclear power plants,

$$z_t = \sum_{s=\max\{1,t-14\}}^{t} y_s, \qquad t = 1, \ldots, T.$$

Since the available capacity must meet the forecasted demand, we require

$$w_t + z_t + e_t \geq d_t, \qquad t = 1, \ldots, T.$$

Finally, since no more than 20% of the total capacity should ever be nuclear, we have

$$\frac{z_t}{w_t + z_t + e_t} \leq 0.2,$$

which can be written as

$$0.8z_t - 0.2w_t \leq 0.2e_t.$$

Summarizing, the capacity expansion problem is as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{t=1}^{T} (c_t x_t + n_t y_t) \\
\text{subject to} \quad & w_t - \sum_{s=\max\{1,t-19\}}^{t} x_s = 0, && t = 1, \ldots, T, \\
& z_t - \sum_{s=\max\{1,t-14\}}^{t} y_s = 0, && t = 1, \ldots, T, \\
& w_t + z_t \geq d_t - e_t, && t = 1, \ldots, T, \\
& 0.8z_t - 0.2w_t \leq 0.2e_t, && t = 1, \ldots, T, \\
& x_t, y_t, w_t, z_t \geq 0, && t = 1, \ldots, T.
\end{aligned}$$

We note that this formulation is not entirely realistic, because it disregards certain economies of scale that may favor larger plants. However, it can provide a ballpark estimate of the true cost.

A scheduling problem

In the previous examples, the choice of the decision variables was fairly straightforward. We now discuss an example where this choice is less obvious.

A hospital wants to make a weekly night shift (12pm-8am) schedule for its nurses. The demand for nurses for the night shift on day j is an integer d_j, $j = 1, \ldots, 7$. Every nurse works 5 days in a row on the night shift. The problem is to find the minimal number of nurses the hospital needs to hire.

One could try using a decision variable y_j equal to the number of nurses that work on day j. With this definition, however, we would not be able to capture the constraint that every nurse works 5 days in a row. For this reason, we choose the decision variables differently, and define x_j as

the number of nurses starting their week on day j. (For example, a nurse whose week starts on day 5 will work days $5, 6, 7, 1, 2$.) We then have the following problem formulation:

$$
\begin{array}{lrcl}
\text{minimize} & x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 & & \\
\text{subject to} & x_1 + x_4 + x_5 + x_6 + x_7 & \geq & d_1 \\
& x_1 + x_2 + x_5 + x_6 + x_7 & \geq & d_2 \\
& x_1 + x_2 + x_3 + x_6 + x_7 & \geq & d_3 \\
& x_1 + x_2 + x_3 + x_4 + x_7 & \geq & d_4 \\
& x_1 + x_2 + x_3 + x_4 + x_5 & \geq & d_5 \\
& x_2 + x_3 + x_4 + x_5 + x_6 & \geq & d_6 \\
& x_3 + x_4 + x_5 + x_6 + x_7 & \geq & d_7 \\
& x_j \geq 0, \quad x_j \text{ integer.} & &
\end{array}
$$

This would be a linear programming problem, except for the constraint that each x_j must be an integer, and we actually have a linear *integer programming* problem. One way of dealing with this issue is to ignore ("relax") the integrality constraints and obtain the so-called *linear programming relaxation* of the original problem. Because the linear programming problem has fewer constraints, and therefore more options, the optimal cost will be less than or equal to the optimal cost of the original problem. If the optimal solution to the linear programming relaxation happens to be integer, then it is also an optimal solution to the original problem. If it is not integer, we can round each x_j upwards, thus obtaining a feasible, but not necessarily optimal, solution to the original problem. It turns out that for this particular problem, an optimal solution can be found without too much effort. However, this is the exception rather than the rule: finding optimal solutions to general integer programming problems is typically difficult; some methods will be discussed in Chapter 11.

Choosing paths in a communication network

Consider a communication network consisting of n nodes. Nodes are connected by communication links. A link allowing one-way transmission from node i to node j is described by an ordered pair (i, j). Let $\mathcal{A}$ be the set of all links. We assume that each link $(i, j) \in \mathcal{A}$ can carry up to u_{ij} bits per second. There is a positive charge c_{ij} per bit transmitted along that link. Each node k generates data, at the rate of $b^{k\ell}$ bits per second, that have to be transmitted to node ℓ, either through a direct link (k, ℓ) or by tracing a sequence of links. The problem is to choose paths along which all data reach their intended destinations, while minimizing the total cost. We allow the data with the same origin and destination to be split and be transmitted along different paths.

In order to formulate this problem as a linear programming problem, we introduce variables $x_{ij}^{k\ell}$ indicating the amount of data with origin k and

destination ℓ that traverse link (i, j). Let

$$b_i^{k\ell} = \begin{cases} b^{k\ell}, & \text{if } i = k, \\ -b^{k\ell}, & \text{if } i = \ell, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $b_i^{k\ell}$ is the net inflow at node i, from outside the network, of data with origin k and destination ℓ. We then have the following formulation:

$$\text{minimize} \quad \sum_{(i,j)\in\mathcal{A}} \sum_{k=1}^{n} \sum_{\ell=1}^{n} c_{ij} x_{ij}^{k\ell}$$

$$\text{subject to} \quad \sum_{\{j|(i,j)\in\mathcal{A}\}} x_{ij}^{k\ell} - \sum_{\{j|(j,i)\in\mathcal{A}\}} x_{ji}^{k\ell} = b_i^{k\ell}, \qquad i, k, \ell = 1, \ldots, n,$$

$$\sum_{k=1}^{n} \sum_{\ell=1}^{n} x_{ij}^{k\ell} \leq u_{ij}, \qquad (i, j) \in \mathcal{A},$$

$$x_{ij}^{k\ell} \geq 0, \qquad (i, j) \in \mathcal{A}, \quad k, \ell = 1, \ldots, n.$$

The first constraint is a flow conservation constraint at node i for data with origin k and destination ℓ. The expression

$$\sum_{\{j|(i,j)\in\mathcal{A}\}} x_{ij}^{k\ell}$$

represents the amount of data with origin and destination k and ℓ, respectively, that leave node i along some link. The expression

$$\sum_{\{j|(j,i)\in\mathcal{A}\}} x_{ji}^{k\ell}$$

represents the amount of data with the same origin and destination that enter node i through some link. Finally, $b_i^{k\ell}$ is the net amount of such data that enter node i from outside the network. The second constraint expresses the requirement that the total traffic through a link (i, j) cannot exceed the link's capacity.

This problem is known as the *multicommodity flow* problem, with the traffic corresponding to each origin-destination pair viewed as a different commodity. A mathematically similar problem arises when we consider a transportation company that wishes to transport several commodities from their origins to their destinations through a network. There is a version of this problem, known as the minimum cost *network flow* problem, in which we do not distinguish between different commodities. Instead, we are given the amount b_i of external supply or demand at each node i, and the objective is to transport material from the supply nodes to the demand nodes, at minimum cost. The network flow problem, which is the subject of Chapter 7, contains as special cases some important problems such as the shortest path problem, the maximum flow problem, and the assignment problem.

Pattern classification

We are given m examples of objects and for each one, say the ith one, a description of its features in terms of an n-dimensional vector $\mathbf{a}_i$. Objects belong to one of two classes, and for each example we are told the class that it belongs to.

More concretely, suppose that each object is an image of an apple or an orange (these are our two classes). In this context, we can use a three-dimensional feature vector $\mathbf{a}_i$ to summarize the contents of the ith image. The three components of $\mathbf{a}_i$ (the features) could be the ellipticity of the object, the length of its stem, and its color, as measured in some scale. We are interested in designing a *classifier* which, given a new object (other than the originally available examples), will figure out whether it is an image of an apple or of an orange.

A *linear classifier* is defined in terms of an n-dimensional vector $\mathbf{x}$ and a scalar x_{n+1}, and operates as follows. Given a new object with feature vector $\mathbf{a}$, the classifier declares it to be an object of the first class if

$$\mathbf{a}'\mathbf{x} \geq x_{n+1},$$

and of the second class if

$$\mathbf{a}'\mathbf{x} < x_{n+1}.$$

In words, a linear classifier makes decisions on the basis of a linear combination of the different features. Our objective is to use the available examples in order to design a "good" linear classifier.

There are many ways of approaching this problem, but a reasonable starting point could be the requirement that the classifier must give the correct answer for each one of the available examples. Let S be the set of examples of the first class. We are then looking for some $\mathbf{x}$ and x_{n+1} that satisfy the constraints

$$\begin{aligned} \mathbf{a}_i'\mathbf{x} &\geq x_{n+1}, \qquad && i \in S, \\ \mathbf{a}_i'\mathbf{x} &< x_{n+1}, \qquad && i \notin S. \end{aligned}$$

Note that the second set of constraints involves a strict inequality and is not quite of the form arising in linear programming. This issue can be bypassed by observing that if some choice of $\mathbf{x}$ and x_{n+1} satisfies all of the above constraints, then there exists some other choice (obtained by multiplying $\mathbf{x}$ and x_{n+1} by a suitably large positive scalar) that satisfies

$$\begin{aligned} \mathbf{a}_i'\mathbf{x} &\geq x_{n+1}, \qquad && i \in S, \\ \mathbf{a}_i'\mathbf{x} &\leq x_{n+1} - 1, \qquad && i \notin S. \end{aligned}$$

We conclude that the search for a linear classifier consistent with all available examples is a problem of finding a feasible solution to a linear programming problem.

1.3 Piecewise linear convex objective functions

All of the examples in the preceding section involved a *linear* objective function. However, there is an important class of optimization problems with a nonlinear objective function that can be cast as linear programming problems; these are examined next.

We first need some definitions:

Definition 1.1

(a) *A function* $f : \Re^n \mapsto \Re$ *is called* **convex** *if for every* $\mathbf{x}, \mathbf{y} \in \Re^n$, *and every* $\lambda \in [0, 1]$, *we have*

$$f\big(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}\big) \leq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y}).$$

(b) *A function* $f : \Re^n \mapsto \Re$ *is called* **concave** *if for every* $\mathbf{x}, \mathbf{y} \in \Re^n$, *and every* $\lambda \in [0, 1]$, *we have*

$$f\big(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}\big) \geq \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y}).$$

Note that if $\mathbf{x}$ and $\mathbf{y}$ are vectors in $\Re^n$ and if λ ranges in $[0, 1]$, then points of the form $\lambda\mathbf{x} + (1-\lambda)\mathbf{y}$ belong to the line segment joining $\mathbf{x}$ and $\mathbf{y}$. The definition of a convex function refers to the values of f, as its argument traces this segment. If f were linear, the inequality in part (a) of the definition would hold with equality. The inequality therefore means that when we restrict attention to such a segment, the graph of the function lies no higher than the graph of a corresponding linear function; see Figure 1.1(a).

It is easily seen that a function f is convex if and only if the function $-f$ is concave. Note that a function of the form $f(\mathbf{x}) = a_0 + \sum_{i=1}^{n} a_i x_i$, where $a_0, \ldots, a_n$ are scalars, called an *affine* function, is both convex and concave. (It turns out that affine functions are the only functions that are both convex and concave.) Convex (as well as concave) functions play a central role in optimization.

We say that a vector $\mathbf{x}$ is a *local minimum* of f if $f(\mathbf{x}) \leq f(\mathbf{y})$ for all $\mathbf{y}$ in the vicinity of $\mathbf{x}$. We also say that $\mathbf{x}$ is a *global minimum* if $f(\mathbf{x}) \leq f(\mathbf{y})$ for all $\mathbf{y}$. A convex function cannot have local minima that fail to be global minima (see Figure 1.1), and this property is of great help in designing efficient optimization algorithms.

Let $\mathbf{c}_1, \ldots, \mathbf{c}_m$ be vectors in $\Re^n$, let $d_1, \ldots, d_m$ be scalars, and consider the function $f : \Re^n \mapsto \Re$ defined by

$$f(\mathbf{x}) = \max_{i=1,\ldots,m} (\mathbf{c}_i'\mathbf{x} + d_i)$$

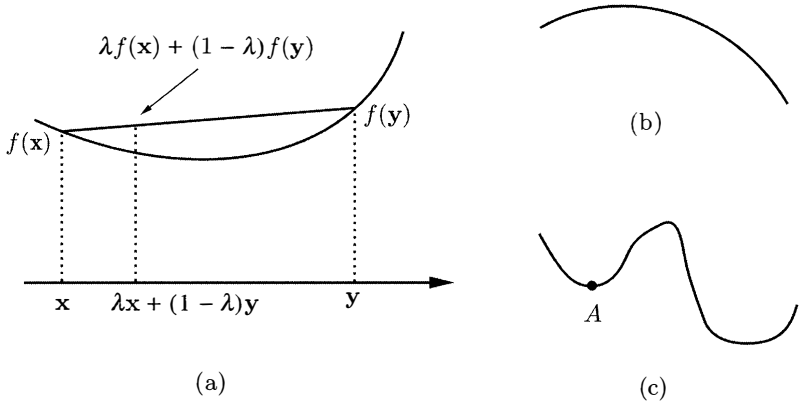


Figure 1.1: (a) Illustration of the definition of a convex function. (b) A concave function. (c) A function that is neither convex nor concave; note that A is a local, but not global, minimum.

[see Figure 1.2(a)]. Such a function is convex, as a consequence of the following result.

Theorem 1.1 *Let $f_1, \ldots, f_m : \Re^n \mapsto \Re$ be convex functions. Then, the function f defined by $f(\mathbf{x}) = \max_{i=1,\ldots,m} f_i(\mathbf{x})$ is also convex.*

Proof. Let $\mathbf{x}, \mathbf{y} \in \Re^n$ and let $\lambda \in [0,1]$. We have

$$\begin{aligned} f\big(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}\big) &= \max_{i=1,\ldots,m} f_i\big(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}\big) \\ &\leq \max_{i=1,\ldots,m} \big(\lambda f_i(\mathbf{x}) + (1-\lambda) f_i(\mathbf{y})\big) \\ &\leq \max_{i=1,\ldots,m} \lambda f_i(\mathbf{x}) + \max_{i=1,\ldots,m} (1-\lambda) f_i(\mathbf{y}) \\ &= \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y}). \end{aligned}$$

□

A function of the form $\max_{i=1,\ldots,m}(\mathbf{c}_i'\mathbf{x}+d_i)$ is called a *piecewise linear convex* function. A simple example is the absolute value function defined by $f(x) = |x| = \max\{x, -x\}$. As illustrated in Figure 1.2(b), a piecewise linear convex function can be used to approximate a general convex function.

We now consider a generalization of linear programming, where the objective function is piecewise linear and convex rather than linear:

$$\begin{aligned} &\text{minimize} && \max_{i=1,\ldots,m} (\mathbf{c}_i'\mathbf{x} + d_i) \\ &\text{subject to} && \mathbf{A}\mathbf{x} \geq \mathbf{b}. \end{aligned}$$

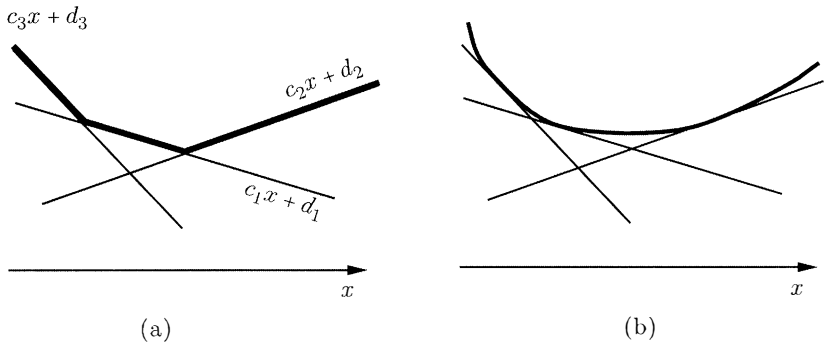


Figure 1.2: (a) A piecewise linear convex function of a single variable. (b) An approximation of a convex function by a piecewise linear convex function.

Note that $\max_{i=1,\ldots,m}(\mathbf{c}_i'\mathbf{x} + d_i)$ is equal to the smallest number z that satisfies $z \geq \mathbf{c}_i'\mathbf{x} + d_i$ for all i. For this reason, the optimization problem under consideration is equivalent to the linear programming problem

$$\begin{array}{lll} \text{minimize} & z & \\ \text{subject to} & z \geq \mathbf{c}_i'\mathbf{x} + d_i, & i = 1, \ldots, m, \\ & \mathbf{A}\mathbf{x} \geq \mathbf{b}, & \end{array}$$

where the decision variables are z and $\mathbf{x}$.

To summarize, linear programming can be used to solve problems with piecewise linear convex cost functions, and the latter class of functions can be used as an approximation of more general convex cost functions. On the other hand, such a piecewise linear approximation is not always a good idea because it can turn a smooth function into a nonsmooth one (piecewise linear functions have discontinuous derivatives).

We finally note that if we are given a constraint of the form $f(\mathbf{x}) \leq h$, where f is the piecewise linear convex function $f(\mathbf{x}) = \max_{i=1,\ldots,m}(\mathbf{f}_i'\mathbf{x}+g_i)$, such a constraint can be rewritten as

$$\mathbf{f}_i'\mathbf{x} + g_i \leq h, \qquad i = 1, \ldots, m,$$

and linear programming is again applicable.

Problems involving absolute values

Consider a problem of the form

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{n} c_i|x_i| \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b}, \end{array}$$

where $\mathbf{x} = (x_1, \ldots, x_n)$, and where the cost coefficients c_i are assumed to be nonnegative. The cost criterion, being the sum of the piecewise linear convex functions $c_i|x_i|$ is easily shown to be piecewise linear and convex (Exercise 1.2). However, expressing this cost criterion in the form $\max_j(\mathbf{c}_j'\mathbf{x} + d_j)$ is a bit involved, and a more direct route is preferable. We observe that $|x_i|$ is the smallest number z_i that satisfies $x_i \leq z_i$ and $-x_i \leq z_i$, and we obtain the linear programming formulation

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} c_i z_i \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & x_i \leq z_i, \qquad i = 1, \ldots, n, \\ & -x_i \leq z_i, \qquad i = 1, \ldots, n. \end{aligned}$$

An alternative method for dealing with absolute values is to introduce new variables x_i^+, x_i^-, constrained to be nonnegative, and let $x_i = x_i^+ - x_i^-$. (Our intention is to have $x_i = x_i^+$ or $x_i = -x_i^-$, depending on whether x_i is positive or negative.) We then replace every occurrence of $|x_i|$ with $x_i^+ + x_i^-$ and obtain the alternative formulation

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} c_i(x_i^+ + x_i^-) \\ \text{subject to} \quad & \mathbf{A}\mathbf{x}^+ - \mathbf{A}\mathbf{x}^- \geq \mathbf{b} \\ & \mathbf{x}^+, \mathbf{x}^- \geq \mathbf{0}, \end{aligned}$$

where $\mathbf{x}^+ = (x_1^+, \ldots, x_n^+)$ and $\mathbf{x}^- = (x_1^-, \ldots, x_n^-)$.

The relations $x_i = x_i^+ - x_i^-$, $x_i^+ \geq 0$, $x_i^- \geq 0$, are not enough to guarantee that $|x_i| = x_i^+ + x_i^-$, and the validity of this reformulation may not be entirely obvious. Let us assume for simplicity that $c_i > 0$ for all i. At an optimal solution to the reformulated problem, and for each i, we must have either $x_i^+ = 0$ or $x_i^- = 0$, because otherwise we could reduce both x_i^+ and x_i^- by the same amount and preserve feasibility, while reducing the cost, in contradiction of optimality. Having guaranteed that either $x_i^+ = 0$ or $x_i^- = 0$, the desired relation $|x_i| = x_i^+ + x_i^-$ now follows.

The formal correctness of the two reformulations that have been presented here, and in a somewhat more general setting, is the subject of Exercise 1.5. We also note that the nonnegativity assumption on the cost coefficients c_i is crucial because, otherwise, the cost criterion is nonconvex.

Example 1.5 Consider the problem

$$\begin{aligned} \text{minimize} \quad & 2|x_1| + x_2 \\ \text{subject to} \quad & x_1 + x_2 \geq 4. \end{aligned}$$

Our first reformulation yields

$$\begin{array}{rrcl} \text{minimize} & 2z_1 + x_2 & & \\ \text{subject to} & x_1 + x_2 & \geq & 4 \\ & x_1 & \leq & z_1 \\ & -x_1 & \leq & z_1, \end{array}$$

while the second yields

$$\begin{array}{rrcl} \text{minimize} & 2x_1^+ + 2x_1^- + x_2 & & \\ \text{subject to} & x_1^+ - x_1^- + x_2 & \geq & 4 \\ & x_1^+ & \geq & 0 \\ & x_1^- & \geq & 0. \end{array}$$

We now continue with some applications involving piecewise linear convex objective functions.

Data fitting

We are given m data points of the form $(\mathbf{a}_i, b_i)$, $i = 1, \ldots, m$, where $\mathbf{a}_i \in \Re^n$ and $b_i \in \Re$, and wish to build a model that predicts the value of the variable b from knowledge of the vector $\mathbf{a}$. In such a situation, one often uses a linear model of the form $b = \mathbf{a}'\mathbf{x}$, where $\mathbf{x}$ is a parameter vector to be determined. Given a particular parameter vector $\mathbf{x}$, the *residual*, or prediction error, at the ith data point is defined as $|b_i - \mathbf{a}_i'\mathbf{x}|$. Given a choice between alternative models, one should choose a model that "explains" the available data as best as possible, i.e., a model that results in small residuals.

One possibility is to minimize the largest residual. This is the problem of minimizing

$$\max_i |b_i - \mathbf{a}_i'\mathbf{x}|,$$

with respect to $\mathbf{x}$, subject to no constraints. Note that we are dealing here with a piecewise linear convex cost criterion. The following is an equivalent linear programming formulation:

$$\begin{array}{rll} \text{minimize} & z & \\ \text{subject to} & b_i - \mathbf{a}_i'\mathbf{x} \leq z, & i = 1, \ldots, m, \\ & -b_i + \mathbf{a}_i'\mathbf{x} \leq z, & i = 1, \ldots, m, \end{array}$$

the decision variables being z and $\mathbf{x}$.

In an alternative formulation, we could adopt the cost criterion

$$\sum_{i=1}^{m} |b_i - \mathbf{a}_i'\mathbf{x}|.$$

Since $|b_i - \mathbf{a}_i'\mathbf{x}|$ is the smallest number z_i that satisfies $b_i - \mathbf{a}_i'\mathbf{x} \leq z_i$ and $-b_i + \mathbf{a}_i'\mathbf{x} \leq z_i$, we obtain the formulation

$$\begin{aligned} \text{minimize} \quad & z_1 + \cdots + z_m \\ \text{subject to} \quad & b_i - \mathbf{a}_i'\mathbf{x} \leq z_i, \qquad i = 1, \ldots, m, \\ & -b_i + \mathbf{a}_i'\mathbf{x} \leq z_i, \qquad i = 1, \ldots, m. \end{aligned}$$

In practice, one may wish to use the quadratic cost criterion $\sum_{i=1}^{m}(b_i - \mathbf{a}_i'\mathbf{x})^2$, in order to obtain a "least squares fit." This is a problem which is easier than linear programming; it can be solved using calculus methods, but its discussion is outside the scope of this book.

Optimal control of linear systems

Consider a dynamical system that evolves according to a model of the form

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \\ y(t) &= \mathbf{c}'\mathbf{x}(t). \end{aligned}$$

Here $\mathbf{x}(t)$ is the state of the system at time t, $y(t)$ is the system output, assumed scalar, and $\mathbf{u}(t)$ is a control vector that we are free to choose subject to linear constraints of the form $\mathbf{D}\mathbf{u}(t) \leq \mathbf{d}$ [these might include saturation constraints, i.e., hard bounds on the magnitude of each component of $\mathbf{u}(t)$]. To mention some possible applications, this could be a model of an airplane, an engine, an electrical circuit, a mechanical system, a manufacturing system, or even a model of economic growth. We are also given the initial state $\mathbf{x}(0)$. In one possible problem, we are to choose the values of the control variables $\mathbf{u}(0), \ldots, \mathbf{u}(T-1)$ to drive the state $\mathbf{x}(T)$ to a target state, assumed for simplicity to be zero. In addition to zeroing the state, it is often desirable to keep the magnitude of the output small at all intermediate times, and we may wish to minimize

$$\max_{t=1,\ldots,T-1} |y(t)|.$$

We then obtain the following linear programming problem:

$$\begin{aligned} \text{minimize} \quad & z \\ \text{subject to} \quad & -z \leq y(t) \leq z, && t = 1, \ldots, T-1, \\ & \mathbf{x}(t+1) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), && t = 0, \ldots, T-1, \\ & y(t) = \mathbf{c}'\mathbf{x}(t), && t = 1, \ldots, T-1, \\ & \mathbf{D}\mathbf{u}(t) \leq \mathbf{d}, && t = 0, \ldots, T-1, \\ & \mathbf{x}(T) = \mathbf{0}, \\ & \mathbf{x}(0) = \text{given}. \end{aligned}$$

Additional linear constraints on the state vectors $\mathbf{x}(t)$, or a more general piecewise linear convex cost function of the state and the control, can also be incorporated.

Rocket control

Consider a rocket that travels along a straight path. Let x_t, v_t, and a_t be the position, velocity, and acceleration, respectively, of the rocket at time t. By discretizing time and by taking the time increment to be unity, we obtain an approximate discrete-time model of the form

$$\begin{aligned} x_{t+1} &= x_t + v_t, \\ v_{t+1} &= v_t + a_t. \end{aligned}$$

We assume that the acceleration a_t is under our control, as it is determined by the rocket thrust. In a rough model, the magnitude $|a_t|$ of the acceleration can be assumed to be proportional to the rate of fuel consumption at time t.

Suppose that the rocket is initially at rest at the origin, that is, $x_0 = 0$ and $v_0 = 0$. We wish the rocket to take off and "land softly" at unit distance from the origin after T time units, that is, $x_T = 1$ and $v_T = 0$. Furthermore, we wish to accomplish this in an economical fashion. One possibility is to minimize the total fuel $\sum_{t=0}^{T-1} |a_t|$ spent subject to the preceding constraints. Alternatively, we may wish to minimize the maximum thrust required, which is $\max_t |a_t|$. Under either alternative, the problem can be formulated as a linear programming problem (Exercise 1.6).

1.4 Graphical representation and solution

In this section, we consider a few simple examples that provide useful geometric insights into the nature of linear programming problems. Our first example involves the graphical solution of a linear programming problem with two variables.

Example 1.6 Consider the problem

$$\begin{aligned} \text{minimize} \quad & -x_1 - x_2 \\ \text{subject to} \quad & x_1 + 2x_2 \leq 3 \\ & 2x_1 + x_2 \leq 3 \\ & x_1, x_2 \geq 0. \end{aligned}$$

The feasible set is the shaded region in Figure 1.3. In order to find an optimal solution, we proceed as follows. For any given scalar z, we consider the set of all points whose cost $\mathbf{c}'\mathbf{x}$ is equal to z; this is the line described by the equation $-x_1 - x_2 = z$. Note that this line is perpendicular to the vector $\mathbf{c} = (-1, -1)$. Different values of z lead to different lines, all of them parallel to each other. In

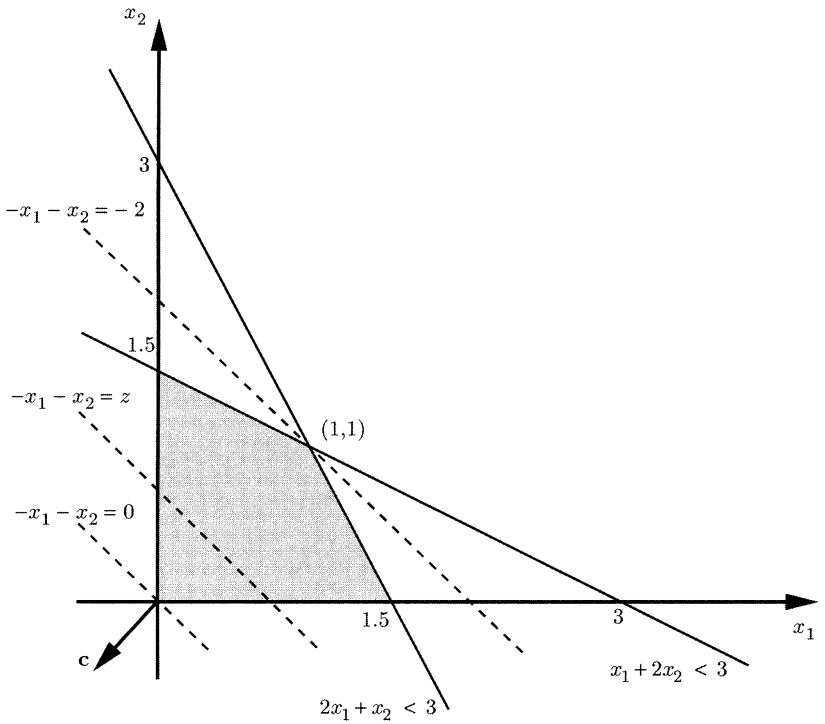


Figure 1.3: Graphical solution of the problem in Example 1.6.

particular, increasing z corresponds to moving the line $z = -x_1 - x_2$ along the direction of the vector $\mathbf{c}$. Since we are interested in minimizing z, we would like to move the line as much as possible in the direction of $-\mathbf{c}$, as long as we do not leave the feasible region. The best we can do is $z = -2$ (see Figure 1.3), and the vector $\mathbf{x} = (1, 1)$ is an optimal solution. Note that this is a corner of the feasible set. (The concept of a "corner" will be defined formally in Chapter 2.)

For a problem in three dimensions, the same approach can be used except that the set of points with the same value of $\mathbf{c}'\mathbf{x}$ is a plane, instead of a line. This plane is again perpendicular to the vector $\mathbf{c}$, and the objective is to slide this plane as much as possible in the direction of $-\mathbf{c}$, as long as we do not leave the feasible set.

Example 1.7 Suppose that the feasible set is the unit cube, described by the constraints $0 \leq x_i \leq 1$, $i = 1, 2, 3$, and that $\mathbf{c} = (-1, -1, -1)$. Then, the vector $\mathbf{x} = (1, 1, 1)$ is an optimal solution. Once more, the optimal solution happens to be a corner of the feasible set (Figure 1.4).

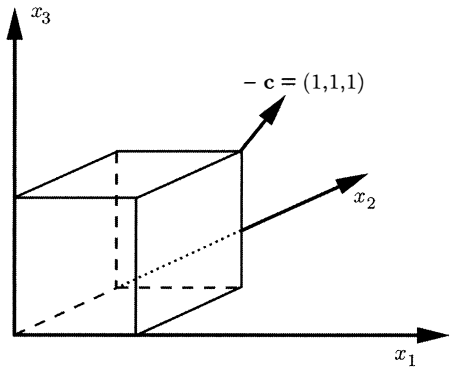


Figure 1.4: The three-dimensional linear programming problem in Example 1.7.

In both of the preceding examples, the feasible set is bounded (does not extend to infinity), and the problem has a unique optimal solution. This is not always the case and we have some additional possibilities that are illustrated by the example that follows.

Example 1.8 Consider the feasible set in $\Re^2$ defined by the constraints

$$\begin{aligned} -x_1 + x_2 &\leq 1 \\ x_1 &\geq 0 \\ x_2 &\geq 0, \end{aligned}$$

which is shown in Figure 1.5.

(a) For the cost vector $\mathbf{c} = (1,1)$, it is clear that $\mathbf{x} = (0,0)$ is the unique optimal solution.

(b) For the cost vector $\mathbf{c} = (1,0)$, there are multiple optimal solutions, namely, every vector $\mathbf{x}$ of the form $\mathbf{x} = (0, x_2)$, with $0 \leq x_2 \leq 1$, is optimal. Note that the set of optimal solutions is bounded.

(c) For the cost vector $\mathbf{c} = (0,1)$, there are multiple optimal solutions, namely, every vector $\mathbf{x}$ of the form $\mathbf{x} = (x_1, 0)$, with $x_1 \geq 0$, is optimal. In this case, the set of optimal solutions is unbounded (contains vectors of arbitrarily large magnitude).

(d) Consider the cost vector $\mathbf{c} = (-1,-1)$. For any feasible solution (x_1, x_2), we can always produce another feasible solution with less cost, by increasing the value of x_1. Therefore, no feasible solution is optimal. Furthermore, by considering vectors (x_1, x_2) with ever increasing values of x_1 and x_2, we can obtain a sequence of feasible solutions whose cost converges to $-\infty$. We therefore say that the optimal cost is $-\infty$.

(e) If we impose an additional constraint of the form $x_1 + x_2 \leq -2$, it is evident that no feasible solution exists.

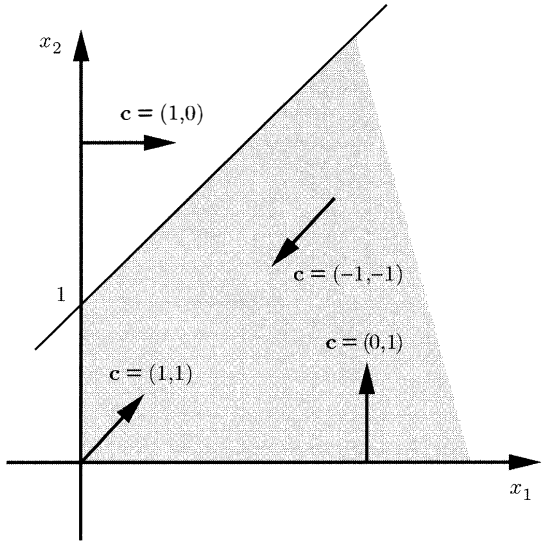


Figure 1.5: The feasible set in Example 1.8. For each choice of $\mathbf{c}$, an optimal solution is obtained by moving as much as possible in the direction of $-\mathbf{c}$.

To summarize the insights obtained from Example 1.8, we have the following possibilities:

(a) There exists a unique optimal solution.

(b) There exist multiple optimal solutions; in this case, the set of optimal solutions can be either bounded or unbounded.

(c) The optimal cost is $-\infty$, and no feasible solution is optimal.

(d) The feasible set is empty.

In principle, there is an additional possibility: an optimal solution does not exist even though the problem is feasible and the optimal cost is not $-\infty$; this is the case, for example, in the problem of minimizing $1/x$ subject to $x > 0$ (for every feasible solution, there exists another with less cost, but the optimal cost is not $-\infty$). We will see later in this book that this possibility never arises in linear programming.

In the examples that we have considered, if the problem has at least one optimal solution, then an optimal solution can be found among the corners of the feasible set. In Chapter 2, we will show that this is a general feature of linear programming problems, as long as the feasible set has at least one corner.

Visualizing standard form problems

We now discuss a method that allows us to visualize standard form problems even if the dimension n of the vector $\mathbf{x}$ is greater than three. The reason for wishing to do so is that when $n \leq 3$, the feasible set of a standard form problem does not have much variety and does not provide enough insight into the general case. (In contrast, if the feasible set is described by constraints of the form $\mathbf{Ax} \geq \mathbf{b}$, enough variety is obtained even if $\mathbf{x}$ has dimension three.)

Suppose that we have a standard form problem, and that the matrix $\mathbf{A}$ has dimensions $m \times n$. In particular, the decision vector $\mathbf{x}$ is of dimension n and we have m equality constraints. We assume that $m \leq n$ and that the constraints $\mathbf{Ax} = \mathbf{b}$ force $\mathbf{x}$ to lie on an $(n - m)$-dimensional set. (Intuitively, each constraint removes one of the "degrees of freedom" of $\mathbf{x}$.) If we "stand" on that $(n-m)$-dimensional set and ignore the m dimensions orthogonal to it, the feasible set is only constrained by the linear inequality constraints $x_i \geq 0$, $i = 1, \ldots, n$. In particular, if $n - m = 2$, the feasible set can be drawn as a two-dimensional set defined by n linear inequality constraints.

To illustrate this approach, consider the feasible set in $\Re^3$ defined by the constraints $x_1 + x_2 + x_3 = 1$ and $x_1, x_2, x_3 \geq 0$ [Figure 1.6(a)], and note that $n = 3$ and $m = 1$. If we stand on the plane defined by the constraint $x_1 + x_2 + x_3 = 1$, then the feasible set has the appearance of a triangle in two-dimensional space. Furthermore, each edge of the triangle corresponds to one of the constraints $x_1, x_2, x_3 \geq 0$; see Figure 1.6(b).

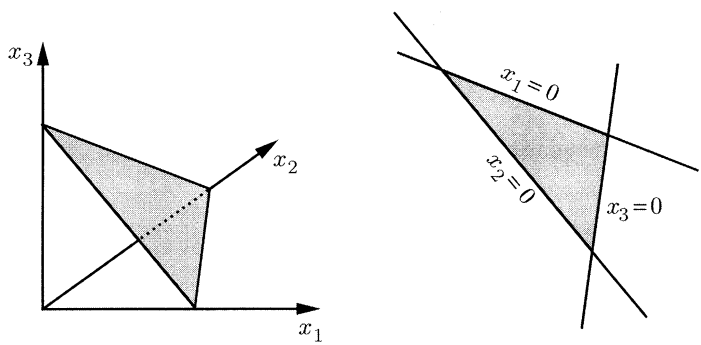


Figure 1.6: (a) An n-dimensional view of the feasible set. (b) An $(n - m)$-dimensional view of the same set.

1.5 Linear algebra background and notation

This section provides a summary of the main notational conventions that we will be employing. It also contains a brief review of those results from linear algebra that are used in the sequel.

Set theoretic notation

If S is a set and x is an element of S, we write $x \in S$. A set can be specified in the form $S = \{x \mid x \text{ satisfies } P\}$, as the set of all elements having property P. The cardinality of a finite set S is denoted by $|S|$. The union of two sets S and T is denoted by $S \cup T$, and their intersection by $S \cap T$. We use $S \backslash T$ to denote the set of all elements of S that do not belong to T. The notation $S \subset T$ means that S is a subset of T, i.e., every element of S is also an element of T; in particular, S could be equal to T. If in addition $S \neq T$, we say that S is a *proper* subset of T. We use $\emptyset$ to denote the empty set. The symbols $\exists$ and $\forall$ have the meanings "there exists" and "for all," respectively.

We use $\Re$ to denote the set of real numbers. For any real numbers a and b, we define the closed and open intervals $[a, b]$ and (a, b), respectively, by

$$[a, b] = \{x \in \Re \mid a \leq x \leq b\},$$

and

$$(a, b) = \{x \in \Re \mid a < x < b\}.$$

Vectors and matrices

A *matrix* of dimensions $m \times n$ is an array of real numbers a_{ij}:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Matrices will be always denoted by upper case boldface characters. If $\mathbf{A}$ is a matrix, we use the notation a_{ij} or $[\mathbf{A}]_{ij}$ to refer to its (i, j)th entry. A *row vector* is a matrix with $m = 1$ and a *column vector* is a matrix with $n = 1$. The word *vector* will always mean *column vector* unless the contrary is explicitly stated. Vectors will be usually denoted by lower case boldface characters. We use the notation $\Re^n$ to indicate the set of all n-dimensional vectors. For any vector $\mathbf{x} \in \Re^n$, we use $x_1, x_2, \ldots, x_n$ to

indicate its components. Thus,

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

The more economical notation $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ will also be used even if we are referring to column vectors. We use $\mathbf{0}$ to denote the vector with all components equal to zero. The ith *unit vector* $\mathbf{e}_i$ is the vector with all components equal to zero except for the ith component which is equal to one.

The *transpose* $\mathbf{A}'$ of an $m \times n$ matrix $\mathbf{A}$ is the $n \times m$ matrix

$$\mathbf{A}' = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{bmatrix};$$

that is, $[\mathbf{A}']_{ij} = [\mathbf{A}]_{ji}$. Similarly, if $\mathbf{x}$ is a vector in $\Re^n$, its transpose $\mathbf{x}'$ is the row vector with the same entries.

If $\mathbf{x}$ and $\mathbf{y}$ are two vectors in $\Re^n$, then

$$\mathbf{x}'\mathbf{y} = \mathbf{y}'\mathbf{x} = \sum_{i=1}^{n} x_i y_i.$$

This quantity is called the *inner product* of $\mathbf{x}$ and $\mathbf{y}$. Two vectors are called *orthogonal* if their inner product is zero. Note that $\mathbf{x}'\mathbf{x} \geq 0$ for every vector $\mathbf{x}$, with equality holding if and only if $\mathbf{x} = \mathbf{0}$. The expression $\sqrt{\mathbf{x}'\mathbf{x}}$ is the *Euclidean norm* of $\mathbf{x}$ and is denoted by $\|\mathbf{x}\|$. The *Schwartz inequality* asserts that for any two vectors of the same dimension, we have

$$|\mathbf{x}'\mathbf{y}| \leq \|\mathbf{x}\| \cdot \|\mathbf{y}\|,$$

with equality holding if and only if one of the two vectors is a scalar multiple of the other.

If $\mathbf{A}$ is an $m \times n$ matrix, we use $\mathbf{A}_j$ to denote its jth column, that is, $\mathbf{A}_j = (a_{1j}, a_{2j}, \ldots, a_{mj})$. (This is our only exception to the rule of using lower case characters to represent vectors.) We also use $\mathbf{a}_i$ to denote the vector formed by the entries of the ith row, that is, $\mathbf{a}_i = (a_{i1}, a_{i2}, \ldots, a_{in})$. Thus,

$$\mathbf{A} = \begin{bmatrix} | & | & & | \\ \mathbf{A}_1 & \mathbf{A}_2 & \cdots & \mathbf{A}_n \\ | & | & & | \end{bmatrix} = \begin{bmatrix} — & \mathbf{a}_1' & — \\ & \vdots & \\ — & \mathbf{a}_m' & — \end{bmatrix}.$$

Given two matrices $\mathbf{A}$, $\mathbf{B}$ of dimensions $m \times n$ and $n \times k$, respectively, their *product* $\mathbf{AB}$ is a matrix of dimensions $m \times k$ whose entries are given by

$$[\mathbf{AB}]_{ij} = \sum_{\ell=1}^{n} [\mathbf{A}]_{i\ell}[\mathbf{B}]_{\ell j} = \mathbf{a}_i' \mathbf{B}_j,$$

where $\mathbf{a}_i'$ is the ith row of $\mathbf{A}$, and $\mathbf{B}_j$ is the jth column of $\mathbf{B}$. Matrix multiplication is associative, i.e., $(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$, but, in general, it is not commutative, that is, the equality $\mathbf{AB} = \mathbf{BA}$ is not always true. We also have $(\mathbf{AB})' = \mathbf{B}'\mathbf{A}'$.

Let $\mathbf{A}$ be an $m \times n$ matrix with columns $\mathbf{A}_i$. We then have $\mathbf{A}\mathbf{e}_i = \mathbf{A}_i$. Any vector $\mathbf{x} \in \Re^n$ can be written in the form $\mathbf{x} = \sum_{i=1}^n x_i \mathbf{e}_i$, which leads to

$$\mathbf{Ax} = \mathbf{A} \sum_{i=1}^{n} \mathbf{e}_i x_i = \sum_{i=1}^{n} \mathbf{A}\mathbf{e}_i x_i = \sum_{i=1}^{n} \mathbf{A}_i x_i.$$

A different representation of the matrix-vector product $\mathbf{Ax}$ is provided by the formula

$$\mathbf{Ax} = \begin{bmatrix} \mathbf{a}_1'\mathbf{x} \\ \mathbf{a}_2'\mathbf{x} \\ \vdots \\ \mathbf{a}_m'\mathbf{x} \end{bmatrix},$$

where $\mathbf{a}_1', \ldots, \mathbf{a}_m'$ are the rows of $\mathbf{A}$.

A matrix is called *square* if the number m of its rows is equal to the number n of its columns. We use $\mathbf{I}$ to denote the *identity* matrix, which is a square matrix whose diagonal entries are equal to one and its off-diagonal entries are equal to zero. The identity matrix satisfies $\mathbf{IA} = \mathbf{A}$ and $\mathbf{BI} = \mathbf{B}$ for any matrices $\mathbf{A}$, $\mathbf{B}$ of dimensions compatible with those of $\mathbf{I}$.

If $\mathbf{x}$ is a vector, the notation $\mathbf{x} \geq \mathbf{0}$ and $\mathbf{x} > \mathbf{0}$ means that every component of $\mathbf{x}$ is nonnegative (respectively, positive). If $\mathbf{A}$ is a matrix, the inequalities $\mathbf{A} \geq \mathbf{0}$ and $\mathbf{A} > \mathbf{0}$ have a similar meaning.

Matrix inversion

Let $\mathbf{A}$ be a square matrix. If there exists a square matrix $\mathbf{B}$ of the same dimensions satisfying $\mathbf{AB} = \mathbf{BA} = \mathbf{I}$, we say that $\mathbf{A}$ is *invertible* or *nonsingular*. Such a matrix $\mathbf{B}$, called the *inverse* of $\mathbf{A}$, is unique and is denoted by $\mathbf{A}^{-1}$. We note that $(\mathbf{A}')^{-1} = (\mathbf{A}^{-1})'$. Also, if $\mathbf{A}$ and $\mathbf{B}$ are invertible matrices of the same dimensions, then $\mathbf{AB}$ is also invertible and $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$.

Given a finite collection of vectors $\mathbf{x}^1, \ldots, \mathbf{x}^K \in \Re^n$, we say that they are *linearly dependent* if there exist real numbers $a_1, \ldots, a_K$, not all of them zero, such that $\sum_{k=1}^K a_k \mathbf{x}^k = \mathbf{0}$; otherwise, they are called *linearly independent*. An equivalent definition of linear independence requires that

none of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^K$ is a linear combination of the remaining vectors (Exercise 1.18). We have the following result.

Theorem 1.2 *Let* $\mathbf{A}$ *be a square matrix. Then, the following statements are equivalent:*

(a) *The matrix* $\mathbf{A}$ *is invertible.*

(b) *The matrix* $\mathbf{A}'$ *is invertible.*

(c) *The determinant of* $\mathbf{A}$ *is nonzero.*

(d) *The rows of* $\mathbf{A}$ *are linearly independent.*

(e) *The columns of* $\mathbf{A}$ *are linearly independent.*

(f) *For every vector* $\mathbf{b}$, *the linear system* $\mathbf{Ax} = \mathbf{b}$ *has a unique solution.*

(g) *There exists some vector* $\mathbf{b}$ *such that the linear system* $\mathbf{Ax} = \mathbf{b}$ *has a unique solution.*

Assuming that $\mathbf{A}$ is an invertible square matrix, an explicit formula for the solution $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$ of the system $\mathbf{Ax} = \mathbf{b}$, is given by *Cramer's rule*. Specifically, the jth component of $\mathbf{x}$ is given by

$$x_j = \frac{\det(\mathbf{A}^j)}{\det(\mathbf{A})},$$

where $\mathbf{A}^j$ is the same matrix as $\mathbf{A}$, except that its jth column is replaced by $\mathbf{b}$. Here, as well as later, the notation $\det(\mathbf{A})$ is used to denote the *determinant* of a square matrix $\mathbf{A}$.

Subspaces and bases

A nonempty subset S of $\Re^n$ is called a *subspace* of $\Re^n$ if $a\mathbf{x} + b\mathbf{y} \in S$ for every $\mathbf{x}, \mathbf{y} \in S$ and every $a, b \in \Re$. If, in addition, $S \neq \Re^n$, we say that S is a *proper* subspace. Note that every subspace must contain the zero vector.

The *span* of a finite number of vectors $\mathbf{x}^1, \ldots, \mathbf{x}^K$ in $\Re^n$ is the subspace of $\Re^n$ defined as the set of all vectors $\mathbf{y}$ of the form $\mathbf{y} = \sum_{k=1}^{K} a_k \mathbf{x}^k$, where each a_k is a real number. Any such vector $\mathbf{y}$ is called a *linear combination* of $\mathbf{x}^1, \ldots, \mathbf{x}^K$.

Given a subspace S of $\Re^n$, with $S \neq \{\mathbf{0}\}$, a *basis* of S is a collection of vectors that are linearly independent and whose span is equal to S. Every basis of a given subspace has the same number of vectors and this number is called the *dimension* of the subspace. In particular, the dimension of $\Re^n$ is equal to n and every proper subspace of $\Re^n$ has dimension smaller than n. Note that one-dimensional subspaces are lines through the origin; two-dimensional subspaces are planes through the origin. Finally, the set $\{\mathbf{0}\}$ is a subspace and its dimension is defined to be zero.

If S is a proper subspace of $\Re^n$, then there exists a nonzero vector $\mathbf{a}$ which is orthogonal to S, that is, $\mathbf{a}'\mathbf{x} = 0$ for every $\mathbf{x} \in S$. More generally, if S has dimension $m < n$, there exist $n - m$ linearly independent vectors that are orthogonal to S.

The result that follows provides some important facts regarding bases and linear independence.

Theorem 1.3 *Suppose that the span S of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^K$ has dimension m. Then:*

(a) *There exists a basis of S consisting of m of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^K$.*

(b) *If $k \leq m$ and $\mathbf{x}^1, \ldots, \mathbf{x}^k$ are linearly independent, we can form a basis of S by starting with $\mathbf{x}^1, \ldots, \mathbf{x}^k$, and choosing $m - k$ of the vectors $\mathbf{x}^{k+1}, \ldots, \mathbf{x}^K$.*

Proof. We only prove part (b), because (a) is the special case of part (b) with $k = 0$. If every vector $\mathbf{x}^{k+1}, \ldots, \mathbf{x}^K$ can be expressed as a linear combination of $\mathbf{x}^1, \ldots, \mathbf{x}^k$, then every vector in the span of $\mathbf{x}^1, \ldots, \mathbf{x}^K$ is also a linear combination of $\mathbf{x}^1, \ldots, \mathbf{x}^k$, and the latter vectors form a basis. (In particular, $m = k$.) Otherwise, at least one of the vectors $\mathbf{x}^{k+1}, \ldots, \mathbf{x}^K$ is linearly independent from $\mathbf{x}^1, \ldots, \mathbf{x}^k$. By picking one such vector, we now have $k+1$ of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^K$ that are linearly independent. By repeating this process $m - k$ times, we end up with the desired basis of S. □

Let $\mathbf{A}$ be a matrix of dimensions $m \times n$. The *column space* of $\mathbf{A}$ is the subspace of $\Re^m$ spanned by the columns of $\mathbf{A}$. The *row space* of $\mathbf{A}$ is the subspace of $\Re^n$ spanned by the rows of $\mathbf{A}$. The dimension of the column space is always equal to the dimension of the row space, and this number is called the *rank* of $\mathbf{A}$. Clearly, $\text{rank}(\mathbf{A}) \leq \min\{m, n\}$. The matrix $\mathbf{A}$ is said to have *full rank* if $\text{rank}(\mathbf{A}) = \min\{m, n\}$. Finally, the set $\{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} = \mathbf{0}\}$ is called the *nullspace* of $\mathbf{A}$; it is a subspace of $\Re^n$ and its dimension is equal to $n - \text{rank}(\mathbf{A})$.

Affine subspaces

Let S_0 be a subspace of $\Re^n$ and let $\mathbf{x}^0$ be some vector. If we add $\mathbf{x}^0$ to every element of S_0, this amounts to translating S_0 by $\mathbf{x}^0$. The resulting set S can be defined formally by

$$S = S_0 + \mathbf{x}^0 = \{\mathbf{x} + \mathbf{x}^0 \mid \mathbf{x} \in S_0\}.$$

In general, S is not a subspace, because it does not necessarily contain the zero vector, and it is called an *affine subspace*. The *dimension* of S is defined to be equal to the dimension of the underlying subspace S_0.

As an example, let $\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^k$ be some vectors in $\Re^n$, and consider the set S of all vectors of the form

$$\mathbf{x}^0 + \lambda_1 \mathbf{x}^1 + \cdots + \lambda_k \mathbf{x}^k,$$

where $\lambda_1, \ldots, \lambda_k$ are arbitrary scalars. For this case, S_0 can be identified with the span of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$, and S is an affine subspace. If the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$ are linearly independent, their span has dimension k, and the affine subspace S also has dimension k.

For a second example, we are given an $m \times n$ matrix $\mathbf{A}$ and a vector $\mathbf{b} \in \Re^m$, and we consider the set

$$S = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} = \mathbf{b}\},$$

which we assume to be nonempty. Let us fix some $\mathbf{x}^0$ such that $\mathbf{A}\mathbf{x}^0 = \mathbf{b}$. An arbitrary vector $\mathbf{x}$ belongs to S if and only if $\mathbf{A}\mathbf{x} = \mathbf{b} = \mathbf{A}\mathbf{x}^0$, or $\mathbf{A}(\mathbf{x} - \mathbf{x}^0) = \mathbf{0}$. Hence, $\mathbf{x} \in S$ if and only if $\mathbf{x} - \mathbf{x}^0$ belongs to the subspace $S_0 = \{\mathbf{y} \mid \mathbf{A}\mathbf{y} = \mathbf{0}\}$. We conclude that $S = \{\mathbf{y} + \mathbf{x}^0 \mid \mathbf{y} \in S_0\}$, and S is an affine subspace of $\Re^n$. If $\mathbf{A}$ has m linearly independent rows, its nullspace S_0 has dimension $n - m$. Hence, the affine subspace S also has dimension $n - m$. Intuitively, if $\mathbf{a}_i'$ are the rows of $\mathbf{A}$, each one of the constraints $\mathbf{a}_i'\mathbf{x} = b_i$ removes one degree of freedom from $\mathbf{x}$, thus reducing the dimension from n to $n - m$; see Figure 1.7 for an illustration.

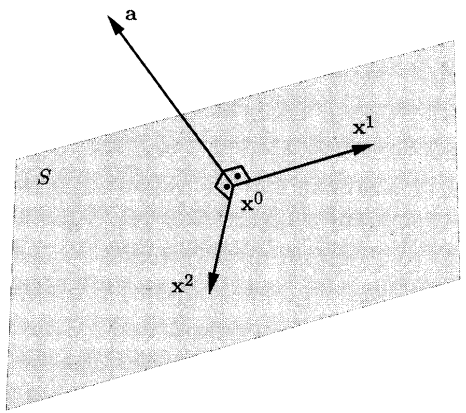


Figure 1.7: Consider a set S in $\Re^3$ defined by a single equality constraint $\mathbf{a}'\mathbf{x} = b$. Let $\mathbf{x}^0$ be an element of S. The vector $\mathbf{a}$ is perpendicular to S. If $\mathbf{x}^1$ and $\mathbf{x}^2$ are linearly independent vectors that are orthogonal to $\mathbf{a}$, then every $\mathbf{x} \in S$ is of the form $\mathbf{x} = \mathbf{x}^0 + \lambda_1 \mathbf{x}^1 + \lambda_2 \mathbf{x}^2$. In particular, S is a two-dimensional affine subspace.

1.6 Algorithms and operation counts

Optimization problems such as linear programming and, more generally, all computational problems are solved by *algorithms*. Loosely speaking, an algorithm is a finite set of instructions of the type used in common programming languages (arithmetic operations, conditional statements, read and write statements, etc.). Although the running time of an algorithm may depend substantially on clever programming or on the computer hardware available, we are interested in comparing algorithms without having to examine the details of a particular implementation. As a first approximation, this can be accomplished by counting the number of arithmetic operations (additions, multiplications, divisions, comparisons) required by an algorithm. This approach is often adequate even though it ignores the fact that adding or multiplying large integers or high-precision floating point numbers is more demanding than adding or multiplying single-digit integers. A more refined approach will be discussed briefly in Chapter 8.

Example 1.9

(a) Let $\mathbf{a}$ and $\mathbf{b}$ be vectors in $\Re^n$. The natural algorithm for computing $\mathbf{a}'\mathbf{b}$ requires n multiplications and $n-1$ additions, for a total of $2n-1$ arithmetic operations.

(b) Let $\mathbf{A}$ and $\mathbf{B}$ be matrices of dimensions $n \times n$. The traditional way of computing $\mathbf{AB}$ forms the inner product of a row of $\mathbf{A}$ and a column of $\mathbf{B}$ to obtain an entry of $\mathbf{AB}$. Since there are n^2 entries to be evaluated, a total of $(2n-1)n^2$ arithmetic operations are involved.

In Example 1.9, an exact operation count was possible. However, for more complicated problems and algorithms, an exact count is usually very difficult. For this reason, we will settle for an estimate of the rate of growth of the number of arithmetic operations, as a function of the problem parameters. Thus, in Example 1.9, we might be content to say that the number of operations in the computation of an inner product increases linearly with n, and the number of operations in matrix multiplication increases cubically with n. This leads us to the order of magnitude notation that we define next.

Definition 1.2 *Let f and g be functions that map positive numbers to positive numbers.*

(a) *We write $f(n) = O\big(g(n)\big)$ if there exist positive numbers n_0 and c such that $f(n) \leq cg(n)$ for all $n \geq n_0$.*

(b) *We write $f(n) = \Omega\big(g(n)\big)$ if there exist positive numbers n_0 and c such that $f(n) \geq cg(n)$ for all $n \geq n_0$.*

(c) *We write $f(n) = \Theta\big(g(n)\big)$ if both $f(n) = O\big(g(n)\big)$ and $f(n) = \Omega\big(g(n)\big)$ hold.*

For example, we have $3n^3 + n^2 + 10 = \Theta(n^3)$, $n \log n = O(n^2)$, and $n \log n = \Omega(n)$.

While the running time of the algorithms considered in Example 1.9 is predictable, the running time of more complicated algorithms often depends on the numerical values of the input data. In such cases, instead of trying to estimate the running time for each possible choice of the input, it is customary to estimate the running time for the *worst possible input data* of a given "size." For example, if we have an algorithm for linear programming, we might be interested in estimating its worst-case running time over all problems with a given number of variables and constraints. This emphasis on the worst case is somewhat conservative and, in practice, the "average" running time of an algorithm might be more relevant. However, the average running time is much more difficult to estimate, or even to define, and for this reason, the worst-case approach is widely used.

Example 1.10 (Operation count of linear system solvers and matrix inversion) Consider the problem of solving a system of n linear equations in n unknowns. The classical method that eliminates one variable at a time (Gaussian elimination) is known to require $O(n^3)$ arithmetic operations in order to either compute a solution or to decide that no solution exists. Practical methods for matrix inversion also require $O(n^3)$ arithmetic operations. These facts will be of use later on.

Is the $O(n^3)$ running time of Gaussian elimination good or bad? Some perspective into this question is provided by the following observation: each time that technological advances lead to computer hardware that is faster by a factor of 8 (presumably every few years), we can solve problems of twice the size than earlier possible. A similar argument applies to algorithms whose running time is $O(n^k)$ for some positive integer k. Such algorithms are said to run in *polynomial time*.

Algorithms also exist whose running time is $\Omega(2^{cn})$, where n is a parameter representing problem size and c is a constant; these are said to take at least *exponential time*. For such algorithms and if $c = 1$, each time that computer hardware becomes faster by a factor of 2, we can increase the value of n that we can handle only by 1. It is then reasonable to expect that no matter how much technology improves, problems with truly large values of n will always be difficult to handle.

Example 1.11 Suppose that we have a choice of two algorithms. The running time of the first is $10^n/100$ (exponential) and the running time of the second is $10n^3$ (polynomial). For very small n, e.g., for $n = 3$, the exponential time algorithm is preferable. To gain some perspective as to what happens for larger n, suppose that we have access to a workstation that can execute 10^7 arithmetic operations per second and that we are willing to let it run for 1000 seconds. Let us figure out what size problems can each algorithm handle within this time frame. The equation $10^n/100 = 10^7 \times 1000$ yields $n = 12$, whereas the equation

$10n^3 = 10^7 \times 1000$ yields $n = 1000$, indicating that the polynomial time algorithm allows us to solve much larger problems.

The point of view emerging from the above discussion is that, as a first cut, it is useful to juxtapose polynomial and exponential time algorithms, the former being viewed as relatively fast and efficient, and the latter as relatively slow. This point of view is justified in many – but not all – contexts and we will be returning to it later in this book.

1.7 Exercises

Exercise 1.1* Suppose that a function $f : \Re^n \mapsto \Re$ is both concave and convex. Prove that f is an affine function.

Exercise 1.2 Suppose that $f_1, \ldots, f_m$ are convex functions from $\Re^n$ into $\Re$ and let $f(\mathbf{x}) = \sum_{i=1}^m f_i(\mathbf{x})$.

(a) Show that if each f_i is convex, so is f.

(b) Show that if each f_i is piecewise linear and convex, so is f.

Exercise 1.3 Consider the problem of minimizing a cost function of the form $\mathbf{c}'\mathbf{x} + f(\mathbf{d}'\mathbf{x})$, subject to the linear constraints $\mathbf{A}\mathbf{x} \geq \mathbf{b}$. Here, $\mathbf{d}$ is a given vector and the function $f : \Re \mapsto \Re$ is as specified in Figure 1.8. Provide a linear programming formulation of this problem.

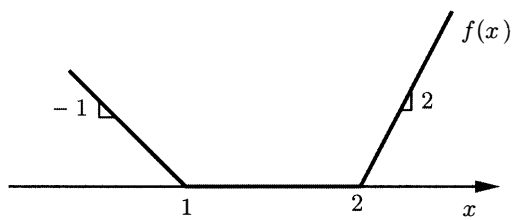


Figure 1.8: The function f of Exercise 1.3.

Exercise 1.4 Consider the problem

$$\begin{aligned} \text{minimize} \quad & 2x_1 + 3|x_2 - 10| \\ \text{subject to} \quad & |x_1 + 2| + |x_2| \leq 5, \end{aligned}$$

and reformulate it as a linear programming problem.

Exercise 1.5 Consider a linear optimization problem, with absolute values, of the following form:

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + \mathbf{d}'\mathbf{y} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} \leq \mathbf{b} \\ & y_i = |x_i|, \qquad \forall\, i. \end{aligned}$$

Assume that all entries of $\mathbf{B}$ and $\mathbf{d}$ are nonnegative.

(a) Provide two different linear programming formulations, along the lines discussed in Section 1.3.

(b) Show that the original problem and the two reformulations are equivalent in the sense that either all three are infeasible, or all three have the same optimal cost.

(c) Provide an example to show that if $\mathbf{B}$ has negative entries, the problem may have a local minimum that is not a global minimum. (It will be seen in Chapter 2 that this is never the case in linear programming problems. Hence, in the presence of such negative entries, a linear programming reformulation is implausible.)

Exercise 1.6 Provide linear programming formulations of the two variants of the rocket control problem discussed at the end of Section 1.3.

Exercise 1.7 (The moment problem) Suppose that Z is a random variable taking values in the set $0, 1, \ldots, K$, with probabilities $p_0, p_1, \ldots, p_K$, respectively. We are given the values of the first two moments $E[Z] = \sum_{k=0}^{K} k p_k$ and $E[Z^2] = \sum_{k=0}^{K} k^2 p_k$ of Z and we would like to obtain upper and lower bounds on the value of the fourth moment $E[Z^4] = \sum_{k=0}^{K} k^4 p_k$ of Z. Show how linear programming can be used to approach this problem.

Exercise 1.8 (Road lighting) Consider a road divided into n segments that is illuminated by m lamps. Let p_j be the power of the jth lamp. The illumination I_i of the ith segment is assumed to be $\sum_{j=1}^{m} a_{ij} p_j$, where a_{ij} are known coefficients. Let I_i^* be the desired illumination of road i.

We are interested in choosing the lamp powers p_j so that the illuminations I_i are close to the desired illuminations I_i^*. Provide a reasonable linear programming formulation of this problem. Note that the wording of the problem is loose and there is more than one possible formulation.

Exercise 1.9 Consider a school district with I neighborhoods, J schools, and G grades at each school. Each school j has a capacity of C_{jg} for grade g. In each neighborhood i, the student population of grade i is S_{ig}. Finally, the distance of school j from neighborhood i is d_{ij}. Formulate a linear programming problem whose objective is to assign all students to schools, while minimizing the total distance traveled by all students. (You may ignore the fact that numbers of students must be integer.)

Exercise 1.10 (Production and inventory planning) A company must deliver d_i units of its product at the end of the ith month. Material produced during

a month can be delivered either at the end of the same month or can be stored as inventory and delivered at the end of a subsequent month; however, there is a storage cost of c_1 dollars per month for each unit of product held in inventory. The year begins with zero inventory. If the company produces x_i units in month i and x_{i+1} units in month $i+1$, it incurs a cost of $c_2|x_{i+1} - x_i|$ dollars, reflecting the cost of switching to a new production level. Formulate a linear programming problem whose objective is to minimize the total cost of the production and inventory schedule over a period of twelve months. Assume that inventory left at the end of the year has no value and does not incur any storage costs.

Exercise 1.11 (Optimal currency conversion) Suppose that there are N available currencies, and assume that one unit of currency i can be exchanged for r_{ij} units of currency j. (Naturally, we assume that $r_{ij} > 0$.) There also certain regulations that impose a limit u_i on the total amount of currency i that can be exchanged on any given day. Suppose that we start with B units of currency 1 and that we would like to maximize the number of units of currency N that we end up with at the end of the day, through a sequence of currency transactions. Provide a linear programming formulation of this problem. Assume that for any sequence $i_1, \ldots, i_k$ of currencies, we have $r_{i_1 i_2} r_{i_2 i_3} \cdots r_{i_{k-1} i_k} r_{i_k i_1} \leq 1$, which means that wealth cannot be multiplied by going through a cycle of currencies.

Exercise 1.12 (Chebychev center) Consider a set P described by linear inequality constraints, that is, $P = \{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \leq b_i,\ i = 1, \ldots, m\}$. A ball with center $\mathbf{y}$ and radius r is defined as the set of all points within (Euclidean) distance r from $\mathbf{y}$. We are interested in finding a ball with the largest possible radius, which is entirely contained within the set P. (The center of such a ball is called the *Chebychev center* of P.) Provide a linear programming formulation of this problem.

Exercise 1.13 (Linear fractional programming) Consider the problem

$$\begin{aligned} \text{minimize} \quad & \frac{\mathbf{c}'\mathbf{x} + d}{\mathbf{f}'\mathbf{x} + g} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{f}'\mathbf{x} + g > 0. \end{aligned}$$

Suppose that we have some prior knowledge that the optimal cost belongs to an interval $[K, L]$. Provide a procedure, that uses linear programming as a subroutine, and that allows us to compute the optimal cost within any desired accuracy. *Hint:* Consider the problem of deciding whether the optimal cost is less than or equal to a certain number.

Exercise 1.14 A company produces and sells two different products. The demand for each product is unlimited, but the company is constrained by cash availability and machine capacity.

Each unit of the first and second product requires 3 and 4 machine hours, respectively. There are 20,000 machine hours available in the current production period. The production costs are \$3 and \$2 per unit of the first and second product, respectively. The selling prices of the first and second product are \$6 and \$5.40 per unit, respectively. The available cash is \$4,000; furthermore, 45%

of the sales revenues from the first product and 30% of the sales revenues from the second product will be made available to finance operations during the current period.

(a) Formulate a linear programming problem that aims at maximizing net income subject to the cash availability and machine capacity limitations.

(b) Solve the problem graphically to obtain an optimal solution.

(c) Suppose that the company could increase its available machine hours by 2,000, after spending \$400 for certain repairs. Should the investment be made?

Exercise 1.15 A company produces two kinds of products. A product of the first type requires 1/4 hours of assembly labor, 1/8 hours of testing, and \$1.2 worth of raw materials. A product of the second type requires 1/3 hours of assembly, 1/3 hours of testing, and \$0.9 worth of raw materials. Given the current personnel of the company, there can be at most 90 hours of assembly labor and 80 hours of testing, each day. Products of the first and second type have a market value of \$9 and \$8, respectively.

(a) Formulate a linear programming problem that can be used to maximize the daily profit of the company.

(b) Consider the following two modifications to the original problem:

(i) Suppose that up to 50 hours of overtime assembly labor can be scheduled, at a cost of \$7 per hour.

(ii) Suppose that the raw material supplier provides a 10% discount if the daily bill is above \$300.

Which of the above two elements can be easily incorporated into the linear programming formulation and how? If one or both are not easy to incorporate, indicate how you might nevertheless solve the problem.

Exercise 1.16 A manager of an oil refinery has 8 million barrels of crude oil A and 5 million barrels of crude oil B allocated for production during the coming month. These resources can be used to make either gasoline, which sells for \$38 per barrel, or home heating oil, which sells for \$33 per barrel. There are three production processes with the following characteristics:

	Process 1	Process 2	Process 3
Input crude A	3	1	5
Input crude B	5	1	3
Output gasoline	4	1	3
Output heating oil	3	1	4
Cost	\$51	\$11	\$40

All quantities are in barrels. For example, with the first process, 3 barrels of crude A and 5 barrels of crude B are used to produce 4 barrels of gasoline and

3 barrels of heating oil. The costs in this table refer to variable and allocated overhead costs, and there are no separate cost items for the cost of the crudes. Formulate a linear programming problem that would help the manager maximize net revenue over the next month.

Exercise 1.17 (Investment under taxation) An investor has a portfolio of n different stocks. He has bought s_i shares of stock i at price p_i, $i = 1, \ldots, n$. The current price of one share of stock i is q_i. The investor expects that the price of one share of stock i in one year will be r_i. If he sells shares, the investor pays transaction costs at the rate of 1% of the amount transacted. In addition, the investor pays taxes at the rate of 30% on capital gains. For example, suppose that the investor sells 1,000 shares of a stock at \$50 per share. He has bought these shares at \$30 per share. He receives \$50,000. However, he owes $0.30 \times (50{,}000 - 30{,}000) = \$6{,}000$ on capital gain taxes and $0.01 \times (50{,}000) = \$500$ on transaction costs. So, by selling 1,000 shares of this stock he nets $50{,}000 - 6{,}000 - 500 = \$43{,}500$. Formulate the problem of selecting how many shares the investor needs to sell in order to raise an amount of money K, net of capital gains and transaction costs, while maximizing the expected value of his portfolio next year.

Exercise 1.18 Show that the vectors in a given finite collection are linearly independent if and only if none of the vectors can be expressed as a linear combination of the others.

Exercise 1.19 Suppose that we are given a set of vectors in $\Re^n$ that form a basis, and let $\mathbf{y}$ be an arbitrary vector in $\Re^n$. We wish to express $\mathbf{y}$ as a linear combination of the basis vectors. How can this be accomplished?

Exercise 1.20

(a) Let $S = \{\mathbf{A}\mathbf{x} \mid \mathbf{x} \in \Re^n\}$, where $\mathbf{A}$ is a given matrix. Show that S is a subspace of $\Re^n$.

(b) Assume that S is a proper subspace of $\Re^n$. Show that there exists a matrix $\mathbf{B}$ such that $S = \{\mathbf{y} \in \Re^n \mid \mathbf{B}\mathbf{y} = \mathbf{0}\}$. *Hint:* Use vectors that are orthogonal to S to form the matrix $\mathbf{B}$.

(c) Suppose that V is an m-dimensional affine subspace of $\Re^n$, with $m < n$. Show that there exist linearly independent vectors $\mathbf{a}_1, \ldots, \mathbf{a}_{n-m}$, and scalars $b_1, \ldots, b_{n-m}$, such that

$$V = \left\{\mathbf{y} \mid \mathbf{a}_i'\mathbf{y} = b_i,\ i = 1, \ldots, n-m\right\}.$$

1.8 History, notes, and sources

The word "programming" has been used traditionally by planners to describe the process of operations planning and resource allocation. In the 1940s, it was realized that this process could often be aided by solving optimization problems involving linear constraints and linear objectives. The term "linear programming" then emerged. The initial impetus came in the aftermath of World War II, within the context of military planning problems. In 1947, Dantzig proposed an algorithm, the *simplex method*, which

made the solution of linear programming problems practical. There followed a period of intense activity during which many important problems in transportation, economics, military operations, scheduling, etc., were cast in this framework. Since then, computer technology has advanced rapidly, the range of applications has expanded, new powerful methods have been discovered, and the underlying mathematical understanding has become deeper and more comprehensive. Today, linear programming is a routinely used tool that can be found in some spreadsheet software packages.

Dantzig's development of the simplex method has been a defining moment in the history of the field, because it came at a time of growing practical needs and of advances in computing technology. But, as is the case with most "scientific revolutions," the history of the field is much richer. Early work goes back to Fourier, who in 1824 developed an algorithm for solving systems of linear inequalities. Fourier's method is far less efficient than the simplex method, but this issue was not relevant at the time. In 1910, de la Vallée Poussin developed a method, similar to the simplex method, for minimizing $\max_i |b_i - \mathbf{a}_i'\mathbf{x}|$, a problem that we discussed in Section 1.3.

In the late 1930s, the Soviet mathematician Kantorovich became interested in problems of optimal resource allocation in a centrally planned economy, for which he gave linear programming formulations. He also provided a solution method, but his work did not become widely known at the time. Around the same time, several models arising in classical, Walrasian, economics were studied and refined, and led to formulations closely related to linear programming. Koopmans, an economist, played an important role and eventually (in 1975) shared the Nobel Prize in economic science with Kantorovich.

On the theoretical front, the mathematical structures that underlie linear programming were independently studied, in the period 1870-1930, by many prominent mathematicians, such as Farkas, Minkowski, Carathéodory, and others. Also, in 1928, von Neumann developed an important result in game theory that would later prove to have strong connections with the deeper structure of linear programming.

Subsequent to Dantzig's work, there has been much and important research in areas such as large scale optimization, network optimization, interior point methods, integer programming, and complexity theory. We defer the discussion of this research to the notes and sources sections of later chapters. For a more detailed account of the history of linear programming, the reader is referred to Schrijver (1986), Orden (1993), and the volume edited by Lenstra, Rinnooy Kan, and Schrijver (1991) (see especially the article by Dantzig in that volume).

There are several texts that cover the general subject of linear programming, starting with a comprehensive one by Dantzig (1963). Some more recent texts are Papadimitriou and Steiglitz (1982), Chvátal (1983), Murty (1983), Luenberger (1984), Bazaraa, Jarvis, and Sherali (1990). Fi-

nally, Schrijver (1986) is a comprehensive, but more advanced reference on the subject.

1.1. The formulation of the diet problem is due to Stigler (1945).

1.2. The case study on DEC's production planning was developed by Freund and Shannahan (1992). Methods for dealing with the nurse scheduling and other cyclic problems are studied by Bartholdi, Orlin, and Ratliff (1980). More information on pattern classification can be found in Duda and Hart (1973), or Haykin (1994).

1.3. A deep and comprehensive treatment of convex functions and their properties is provided by Rockafellar (1970). Linear programming arises in control problems, in ways that are more sophisticated than what is described here; see, e.g., Dahleh and Diaz-Bobillo (1995).

1.5. For an introduction to linear algebra, see Strang (1988).

1.6. For a more detailed treatment of algorithms and their computational requirements, see Lewis and Papadimitriou (1981), Papadimitriou and Steiglitz (1982), or Cormen, Leiserson, and Rivest (1990).

1.7. Exercise 1.8 is adapted from Boyd and Vandenberghe (1995). Exercises 1.9 and 1.14 are adapted from Bradley, Hax, and Magnanti (1977). Exercise 1.11 is adapted from Ahuja, Magnanti, and Orlin (1993).

Chapter 2

The geometry of linear programming

Contents

In this chapter, we define a polyhedron as a set described by a finite number of linear equality and inequality constraints. In particular, the feasible set in a linear programming problem is a polyhedron. We study the basic geometric properties of polyhedra in some detail, with emphasis on their "corner points" (vertices). As it turns out, common geometric intuition derived from the familiar three-dimensional polyhedra is essentially correct when applied to higher-dimensional polyhedra. Another interesting aspect of the development in this chapter is that certain concepts (e.g., the concept of a vertex) can be defined either geometrically or algebraically. While the geometric view may be more natural, the algebraic approach is essential for carrying out computations. Much of the richness of the subject lies in the interplay between the geometric and the algebraic points of view.

Our development starts with a characterization of the corner points of feasible sets in the general form $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$. Later on, we focus on the case where the feasible set is in the standard form $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$, and we derive a simple algebraic characterization of the corner points. The latter characterization will play a central role in the development of the simplex method in Chapter 3.

The main results of this chapter state that a nonempty polyhedron has at least one corner point if and only if it does not contain a line, and if this is the case, the search for optimal solutions to linear programming problems can be restricted to corner points. These results are proved for the most general case of linear programming problems using geometric arguments. The same results will also be proved in the next chapter, for the case of standard form problems, as a corollary of our development of the simplex method. Thus, the reader who wishes to focus on standard form problems may skip the proofs in Sections 2.5 and 2.6. Finally, Sections 2.7 and 2.8 can also be skipped during a first reading; any results in these sections that are needed later on will be rederived in Chapter 4, using different techniques.

2.1 Polyhedra and convex sets

In this section, we introduce some important concepts that will be used to study the geometry of linear programming, including a discussion of convexity.

Hyperplanes, halfspaces, and polyhedra

We start with the formal definition of a polyhedron.

Definition 2.1 *A* **polyhedron** *is a set that can be described in the form* $\{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$, *where* $\mathbf{A}$ *is an* $m \times n$ *matrix and* $\mathbf{b}$ *is a vector in* $\Re^m$.

As discussed in Section 1.1, the feasible set of any linear programming problem can be described by inequality constraints of the form $\mathbf{Ax} \geq \mathbf{b}$, and is therefore a polyhedron. In particular, a set of the form $\{\mathbf{x} \in \Re^n \mid \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ is also a polyhedron and will be referred to as a *polyhedron in standard form.*

A polyhedron can either "extend to infinity," or can be confined in a finite region. The definition that follows refers to this distinction.

Definition 2.2 *A set $S \subset \Re^n$ is* **bounded** *if there exists a constant K such that the absolute value of every component of every element of S is less than or equal to K.*

The next definition deals with polyhedra determined by a single linear constraint.

Definition 2.3 *Let $\mathbf{a}$ be a nonzero vector in $\Re^n$ and let b be a scalar.*

(a) *The set $\{\mathbf{x} \in \Re^n \mid \mathbf{a}'\mathbf{x} = b\}$ is called a* **hyperplane.**

(b) *The set $\{\mathbf{x} \in \Re^n \mid \mathbf{a}'\mathbf{x} \geq b\}$ is called a* **halfspace.**

Note that a hyperplane is the boundary of a corresponding halfspace. In addition, the vector $\mathbf{a}$ in the definition of the hyperplane is perpendicular to the hyperplane itself. [To see this, note that if $\mathbf{x}$ and $\mathbf{y}$ belong to the same hyperplane, then $\mathbf{a}'\mathbf{x} = \mathbf{a}'\mathbf{y}$. Hence, $\mathbf{a}'(\mathbf{x} - \mathbf{y}) = 0$ and therefore $\mathbf{a}$ is orthogonal to any direction vector confined to the hyperplane.] Finally, note that a polyhedron is equal to the intersection of a finite number of halfspaces; see Figure 2.1.

Convex Sets

We now define the important notion of a convex set.

Definition 2.4 *A set $S \subset \Re^n$ is* **convex** *if for any $\mathbf{x}, \mathbf{y} \in S$, and any $\lambda \in [0,1]$, we have $\lambda\mathbf{x} + (1-\lambda)\mathbf{y} \in S$.*

Note that if $\lambda \in [0,1]$, then $\lambda\mathbf{x} + (1-\lambda)\mathbf{y}$ is a weighted average of the vectors $\mathbf{x}$, $\mathbf{y}$, and therefore belongs to the line segment joining $\mathbf{x}$ and $\mathbf{y}$. Thus, a set is convex if the segment joining any two of its elements is contained in the set; see Figure 2.2.

Our next definition refers to weighted averages of a finite number of vectors; see Figure 2.3.

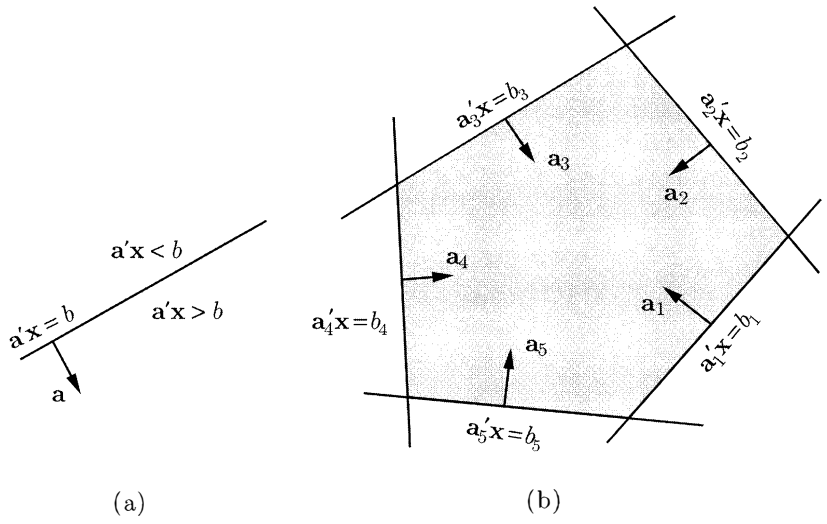


Figure 2.1: (a) A hyperplane and two halfspaces. (b) The polyhedron $\{\mathbf{x} \mid \mathbf{a}_i'\mathbf{x} \geq b_i,\ i = 1, \ldots, 5\}$ is the intersection of five halfspaces. Note that each vector $\mathbf{a}_i$ is perpendicular to the hyperplane $\{\mathbf{x} \mid \mathbf{a}_i'\mathbf{x} = b_i\}$.

Definition 2.5 *Let $\mathbf{x}^1, \ldots, \mathbf{x}^k$ be vectors in $\Re^n$ and let $\lambda_1, \ldots, \lambda_k$ be nonnegative scalars whose sum is unity.*

(a) *The vector $\sum_{i=1}^{k} \lambda_i \mathbf{x}^i$ is said to be a* **convex combination** *of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$.*

(b) *The* **convex hull** *of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$ is the set of all convex combinations of these vectors.*

The result that follows establishes some important facts related to convexity.

Theorem 2.1

(a) *The intersection of convex sets is convex.*

(b) *Every polyhedron is a convex set.*

(c) *A convex combination of a finite number of elements of a convex set also belongs to that set.*

(d) *The convex hull of a finite number of vectors is a convex set.*

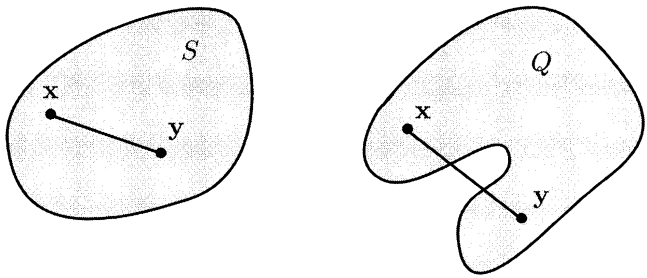


Figure 2.2: The set S is convex, but the set Q is not, because the segment joining $\mathbf{x}$ and $\mathbf{y}$ is not contained in Q.

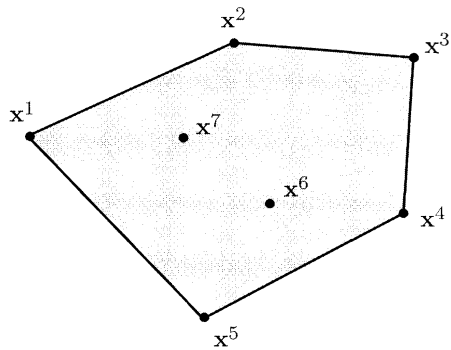


Figure 2.3: The convex hull of seven points in $\Re^2$.

Proof.

(a) Let S_i, $i \in I$, be convex sets where I is some index set, and suppose that $\mathbf{x}$ and $\mathbf{y}$ belong to the intersection $\cap_{i \in I} S_i$. Let $\lambda \in [0, 1]$. Since each S_i is convex and contains $\mathbf{x}$, $\mathbf{y}$, we have $\lambda\mathbf{x} + (1-\lambda)\mathbf{y} \in S_i$, which proves that $\lambda\mathbf{x} + (1-\lambda)\mathbf{y}$ also belongs to the intersection of the sets S_i. Therefore, $\cap_{i \in I} S_i$ is convex.

(b) Let $\mathbf{a}$ be a vector and let b a scalar. Suppose that $\mathbf{x}$ and $\mathbf{y}$ satisfy $\mathbf{a}'\mathbf{x} \geq b$ and $\mathbf{a}'\mathbf{y} \geq b$, respectively, and therefore belong to the same halfspace. Let $\lambda \in [0, 1]$. Then, $\mathbf{a}'(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \geq \lambda b + (1-\lambda)b = b$, which proves that $\lambda\mathbf{x} + (1-\lambda)\mathbf{y}$ also belongs to the same halfspace. Therefore a halfspace is convex. Since a polyhedron is the intersection of a finite number of halfspaces, the result follows from part (a).

(c) A convex combination of two elements of a convex set lies in that

set, by the definition of convexity. Let us assume, as an induction hypothesis, that a convex combination of k elements of a convex set belongs to that set. Consider $k+1$ elements $\mathbf{x}^1, \ldots, \mathbf{x}^{k+1}$ of a convex set S and let $\lambda_1, \ldots, \lambda_{k+1}$ be nonnegative scalars that sum to 1. We assume, without loss of generality, that $\lambda_{k+1} \neq 1$. We then have

$$\sum_{i=1}^{k+1} \lambda_i \mathbf{x}^i = \lambda_{k+1}\mathbf{x}^{k+1} + (1-\lambda_{k+1}) \sum_{i=1}^{k} \frac{\lambda_i}{1-\lambda_{k+1}} \mathbf{x}^i. \tag{2.1}$$

The coefficients $\lambda_i/(1-\lambda_{k+1})$, $i = 1, \ldots, k$, are nonnegative and sum to unity; using the induction hypothesis, $\sum_{i=1}^{k} \lambda_i \mathbf{x}^i/(1-\lambda_{k+1}) \in S$. Then, the fact that S is convex and Eq. (2.1) imply that $\sum_{i=1}^{k+1} \lambda_i \mathbf{x}^i \in S$, and the induction step is complete.

(d) Let S be the convex hull of the vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$ and let $\mathbf{y} = \sum_{i=1}^{k} \zeta_i \mathbf{x}^i$, $\mathbf{z} = \sum_{i=1}^{k} \theta_i \mathbf{x}^i$ be two elements of S, where $\zeta_i \geq 0$, $\theta_i \geq 0$, and $\sum_{i=1}^{k} \zeta_i = \sum_{i=1}^{k} \theta_i = 1$. Let $\lambda \in [0,1]$. Then,

$$\lambda\mathbf{y} + (1-\lambda)\mathbf{z} = \lambda \sum_{i=1}^{k} \zeta_i \mathbf{x}^i + (1-\lambda) \sum_{i=1}^{k} \theta_i \mathbf{x}^i = \sum_{i=1}^{k} \big(\lambda\zeta_i + (1-\lambda)\theta_i\big)\mathbf{x}^i.$$

We note that the coefficients $\lambda\zeta_i + (1-\lambda)\theta_i$, $i = 1, \ldots, k$, are nonnegative and sum to unity. This shows that $\lambda\mathbf{y} + (1-\lambda)\mathbf{z}$ is a convex combination of $\mathbf{x}^1, \ldots, \mathbf{x}^k$ and, therefore, belongs to S. This establishes the convexity of S. □

2.2 Extreme points, vertices, and basic feasible solutions

We observed in Section 1.4 that an optimal solution to a linear programming problem tends to occur at a "corner" of the polyhedron over which we are optimizing. In this section, we suggest three different ways of defining the concept of a "corner" and then show that all three definitions are equivalent.

Our first definition defines an *extreme point* of a polyhedron as a point that cannot be expressed as a convex combination of two other elements of the polyhedron, and is illustrated in Figure 2.4. Notice that this definition is entirely geometric and does not refer to a specific representation of a polyhedron in terms of linear constraints.

Definition 2.6 *Let P be a polyhedron. A vector $\mathbf{x} \in P$ is an* **extreme point** *of P if we cannot find two vectors $\mathbf{y}, \mathbf{z} \in P$, both different from $\mathbf{x}$, and a scalar $\lambda \in [0,1]$, such that $\mathbf{x} = \lambda\mathbf{y} + (1-\lambda)\mathbf{z}$.*

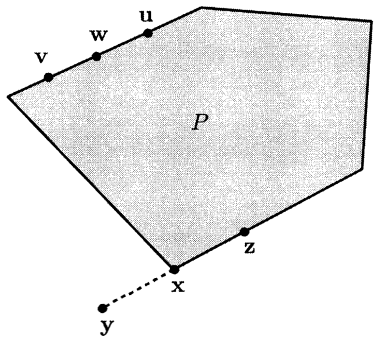


Figure 2.4: The vector $\mathbf{w}$ is not an extreme point because it is a convex combination of $\mathbf{v}$ and $\mathbf{u}$. The vector $\mathbf{x}$ is an extreme point: if $\mathbf{x} = \lambda\mathbf{y} + (1-\lambda)\mathbf{z}$ and $\lambda \in [0,1]$, then either $\mathbf{y} \notin P$, or $\mathbf{z} \notin P$, or $\mathbf{x} = \mathbf{y}$, or $\mathbf{x} = \mathbf{z}$.

An alternative geometric definition defines a *vertex* of a polyhedron P as the unique optimal solution to some linear programming problem with feasible set P.

Definition 2.7 *Let P be a polyhedron. A vector $\mathbf{x} \in P$ is a* **vertex** *of P if there exists some $\mathbf{c}$ such that $\mathbf{c}'\mathbf{x} < \mathbf{c}'\mathbf{y}$ for all $\mathbf{y}$ satisfying $\mathbf{y} \in P$ and $\mathbf{y} \neq \mathbf{x}$.*

In other words, $\mathbf{x}$ is a vertex of P if and only if P is on one side of a hyperplane (the hyperplane $\{\mathbf{y} \mid \mathbf{c}'\mathbf{y} = \mathbf{c}'\mathbf{x}\}$) which meets P only at the point $\mathbf{x}$; see Figure 2.5.

The two geometric definitions that we have given so far are not easy to work with from an algorithmic point of view. We would like to have a definition that relies on a representation of a polyhedron in terms of linear constraints and which reduces to an algebraic test. In order to provide such a definition, we need some more terminology.

Consider a polyhedron $P \subset \Re^n$ defined in terms of the linear equality and inequality constraints

$$\begin{aligned} \mathbf{a}_i'\mathbf{x} &\geq b_i, \qquad i \in M_1, \\ \mathbf{a}_i'\mathbf{x} &\leq b_i, \qquad i \in M_2, \\ \mathbf{a}_i'\mathbf{x} &= b_i, \qquad i \in M_3, \end{aligned}$$

where M_1, M_2, and M_3 are finite index sets, each $\mathbf{a}_i$ is a vector in $\Re^n$, and

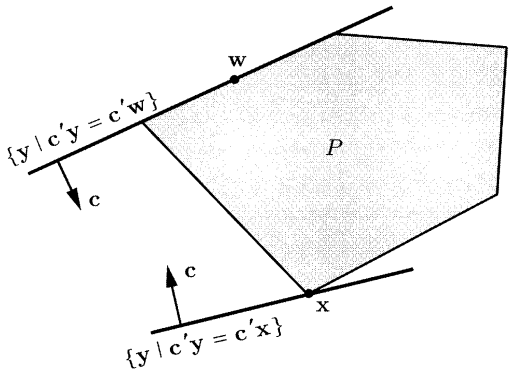


Figure 2.5: The line at the bottom touches P at a single point and $\mathbf{x}$ is a vertex. On the other hand, $\mathbf{w}$ is not a vertex because there is no hyperplane that meets P only at $\mathbf{w}$.

each b_i is a scalar. The definition that follows is illustrated in Figure 2.6.

Definition 2.8 *If a vector* $\mathbf{x}^*$ *satisfies* $\mathbf{a}_i'\mathbf{x}^* = b_i$ *for some* i *in* M_1, M_2, *or* M_3, *we say that the corresponding constraint is* **active** *or* **binding** *at* $\mathbf{x}^*$.

If there are n constraints that are active at a vector $\mathbf{x}^*$, then $\mathbf{x}^*$ satisfies a certain system of n linear equations in n unknowns. This system has a unique solution if and only if these n equations are "linearly independent." The result that follows gives a precise meaning to this statement, together with a slight generalization.

Theorem 2.2 *Let* $\mathbf{x}^*$ *be an element of* $\Re^n$ *and let* $I = \{i \mid \mathbf{a}_i'\mathbf{x}^* = b_i\}$ *be the set of indices of constraints that are active at* $\mathbf{x}^*$. *Then, the following are equivalent:*

(a) *There exist* n *vectors in the set* $\{\mathbf{a}_i \mid i \in I\}$, *which are linearly independent.*

(b) *The span of the vectors* $\mathbf{a}_i$, $i \in I$, *is all of* $\Re^n$, *that is, every element of* $\Re^n$ *can be expressed as a linear combination of the vectors* $\mathbf{a}_i$, $i \in I$.

(c) *The system of equations* $\mathbf{a}_i'\mathbf{x} = b_i$, $i \in I$, *has a unique solution.*

Proof. Suppose that the vectors $\mathbf{a}_i$, $i \in I$, span $\Re^n$. Then, the span of these vectors has dimension n. By Theorem 1.3(a) in Section 1.5, n of

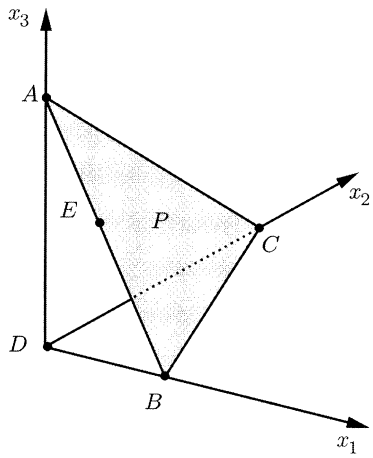


Figure 2.6: Let $P = \{(x_1, x_2, x_3) \mid x_1 + x_2 + x_3 = 1,\ x_1, x_2, x_3 \geq 0\}$. There are three constraints that are active at each one of the points A, B, C and D. There are only two constraints that are active at point E, namely $x_1 + x_2 + x_3 = 1$ and $x_2 = 0$.

these vectors form a basis of $\Re^n$, and are therefore linearly independent. Conversely, suppose that n of the vectors $\mathbf{a}_i$, $i \in I$, are linearly independent. Then, the subspace spanned by these n vectors is n-dimensional and must be equal to $\Re^n$. Hence, every element of $\Re^n$ is a linear combination of the vectors $\mathbf{a}_i$, $i \in I$. This establishes the equivalence of (a) and (b).

If the system of equations $\mathbf{a}_i'\mathbf{x} = b_i$, $i \in I$, has multiple solutions, say $\mathbf{x}^1$ and $\mathbf{x}^2$, then the nonzero vector $\mathbf{d} = \mathbf{x}^1 - \mathbf{x}^2$ satisfies $\mathbf{a}_i'\mathbf{d} = 0$ for all $i \in I$. Since $\mathbf{d}$ is orthogonal to every vector $\mathbf{a}_i$, $i \in I$, $\mathbf{d}$ is not a linear combination of these vectors and it follows that the vectors $\mathbf{a}_i$, $i \in I$, do not span $\Re^n$. Conversely, if the vectors $\mathbf{a}_i$, $i \in I$, do not span $\Re^n$, choose a nonzero vector $\mathbf{d}$ which is orthogonal to the subspace spanned by these vectors. If $\mathbf{x}$ satisfies $\mathbf{a}_i'\mathbf{x} = b_i$ for all $i \in I$, we also have $\mathbf{a}_i'(\mathbf{x} + \mathbf{d}) = b_i$ for all $i \in I$, thus obtaining multiple solutions. We have therefore established that (b) and (c) are equivalent. □

With a slight abuse of language, we will often say that certain *constraints* are *linearly independent*, meaning that the corresponding vectors $\mathbf{a}_i$ are linearly independent. With this terminology, statement (a) in Theorem 2.2 requires that there exist n linearly independent constraints that are active at $\mathbf{x}^*$.

We are now ready to provide an algebraic definition of a corner point, as a feasible solution at which there are n linearly independent active constraints. Note that since we are interested in a feasible solution, all equality

constraints must be active. This suggests the following way of looking for corner points: first impose the equality constraints and then require that enough additional constraints be active, so that we get a total of n linearly independent active constraints. Once we have n linearly independent active constraints, a unique vector $\mathbf{x}^*$ is determined (Theorem 2.2). However, this procedure has no guarantee of leading to a feasible vector $\mathbf{x}^*$, because some of the inactive constraints could be violated; in the latter case we say that we have a basic (but not basic feasible) solution.

Definition 2.9 *Consider a polyhedron P defined by linear equality and inequality constraints, and let $\mathbf{x}^*$ be an element of $\Re^n$.*

(a) *The vector $\mathbf{x}^*$ is a* **basic solution** *if:*

(i) All equality constraints are active;

(ii) Out of the constraints that are active at $\mathbf{x}^*$, there are n of them that are linearly independent.

(b) *If $\mathbf{x}^*$ is a basic solution that satisfies all of the constraints, we say that it is a* **basic feasible solution**.

In reference to Figure 2.6, we note that points A, B, and C are basic feasible solutions. Point D is not a basic solution because it fails to satisfy the equality constraint. Point E is feasible, but not basic. If the equality constraint $x_1 + x_2 + x_3 = 1$ were to be replaced by the constraints $x_1 + x_2 + x_3 \leq 1$ and $x_1 + x_2 + x_3 \geq 1$, then D would be a basic solution, according to our definition. This shows that whether a point is a basic solution or not may depend on the way that a polyhedron is represented. Definition 2.9 is also illustrated in Figure 2.7.

Note that if the number m of constraints used to define a polyhedron $P \subset \Re^n$ is less than n, the number of active constraints at any given point must also be less than n, and there are no basic or basic feasible solutions.

We have given so far three different definitions that are meant to capture the same concept; two of them are geometric (extreme point, vertex) and the third is algebraic (basic feasible solution). Fortunately, all three definitions are equivalent as we prove next and, for this reason, the three terms can be used interchangeably.

Theorem 2.3 *Let P be a nonempty polyhedron and let $\mathbf{x}^* \in P$. Then, the following are equivalent:*

(a) *$\mathbf{x}^*$ is a vertex;*

(b) *$\mathbf{x}^*$ is an extreme point;*

(c) *$\mathbf{x}^*$ is a basic feasible solution.*

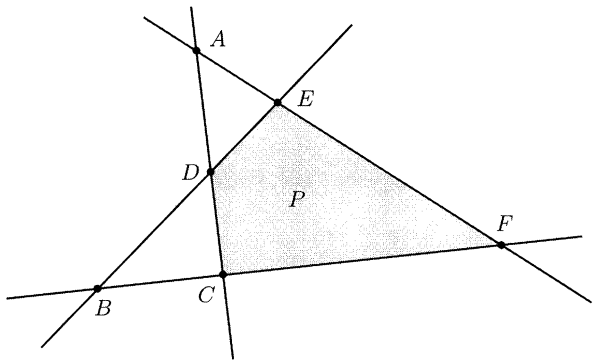


Figure 2.7: The points A, B, C, D, E, F are all basic solutions because at each one of them, there are two linearly independent constraints that are active. Points C, D, E, F are basic feasible solutions.

Proof. For the purposes of this proof and without loss of generality, we assume that P is represented in terms of constraints of the form $\mathbf{a}_i'\mathbf{x} \geq b_i$ and $\mathbf{a}_i'\mathbf{x} = b_i$.

Vertex $\Rightarrow$ Extreme point

Suppose that $\mathbf{x}^* \in P$ is a vertex. Then, by Definition 2.7, there exists some $\mathbf{c} \in \Re^n$ such that $\mathbf{c}'\mathbf{x}^* < \mathbf{c}'\mathbf{y}$ for every $\mathbf{y}$ satisfying $\mathbf{y} \in P$ and $\mathbf{y} \neq \mathbf{x}^*$. If $\mathbf{y} \in P$, $\mathbf{z} \in P$, $\mathbf{y} \neq \mathbf{x}^*$, $\mathbf{z} \neq \mathbf{x}^*$, and $0 \leq \lambda \leq 1$, then $\mathbf{c}'\mathbf{x}^* < \mathbf{c}'\mathbf{y}$ and $\mathbf{c}'\mathbf{x}^* < \mathbf{c}'\mathbf{z}$, which implies that $\mathbf{c}'\mathbf{x}^* < \mathbf{c}'(\lambda\mathbf{y} + (1-\lambda)\mathbf{z})$ and, therefore, $\mathbf{x}^* \neq \lambda\mathbf{y}+(1-\lambda)\mathbf{z}$. Thus, $\mathbf{x}^*$ cannot be expressed as a convex combination of two other elements of P and is, therefore, an extreme point (cf. Definition 2.6).

Extreme point $\Rightarrow$ Basic feasible solution

Suppose that $\mathbf{x}^* \in P$ is not a basic feasible solution. We will show that $\mathbf{x}^*$ is not an extreme point of P. Let $I = \{i \mid \mathbf{a}_i'\mathbf{x}^* = b_i\}$. Since $\mathbf{x}^*$ is not a basic feasible solution, there do not exist n linearly independent vectors in the family $\mathbf{a}_i$, $i \in I$. Thus, the vectors $\mathbf{a}_i$, $i \in I$, lie in a proper subspace of $\Re^n$, and there exists some nonzero vector $\mathbf{d} \in \Re^n$ such that $\mathbf{a}_i'\mathbf{d} = 0$, for all $i \in I$. Let ϵ be a small positive number and consider the vectors $\mathbf{y} = \mathbf{x}^* + \epsilon\mathbf{d}$ and $\mathbf{z} = \mathbf{x}^* - \epsilon\mathbf{d}$. Notice that $\mathbf{a}_i'\mathbf{y} = \mathbf{a}_i'\mathbf{x}^* = b_i$, for $i \in I$. Furthermore, for $i \notin I$, we have $\mathbf{a}_i'\mathbf{x}^* > b_i$ and, provided that ϵ is small, we will also have $\mathbf{a}_i'\mathbf{y} > b_i$. (It suffices to choose ϵ so that $\epsilon|\mathbf{a}_i'\mathbf{d}| < \mathbf{a}_i'\mathbf{x}^* - b_i$ for all $i \notin I$.) Thus, when ϵ is small enough, $\mathbf{y} \in P$ and, by a similar argument, $\mathbf{z} \in P$. We finally notice that $\mathbf{x}^* = (\mathbf{y} + \mathbf{z})/2$, which implies that $\mathbf{x}^*$ is not an extreme point.

Basic feasible solution $\Rightarrow$ Vertex

Let $\mathbf{x}^*$ be a basic feasible solution and let $I = \{i \mid \mathbf{a}_i'\mathbf{x}^* = b_i\}$. Let $\mathbf{c} = \sum_{i \in I} \mathbf{a}_i$. We then have

$$\mathbf{c}'\mathbf{x}^* = \sum_{i \in I} \mathbf{a}_i'\mathbf{x}^* = \sum_{i \in I} b_i.$$

Furthermore, for any $\mathbf{x} \in P$ and any i, we have $\mathbf{a}_i'\mathbf{x} \geq b_i$, and

$$\mathbf{c}'\mathbf{x} = \sum_{i \in I} \mathbf{a}_i'\mathbf{x} \geq \sum_{i \in I} b_i. \tag{2.2}$$

This shows that $\mathbf{x}^*$ is an optimal solution to the problem of minimizing $\mathbf{c}'\mathbf{x}$ over the set P. Furthermore, equality holds in (2.2) if and only if $\mathbf{a}_i'\mathbf{x} = b_i$ for all $i \in I$. Since $\mathbf{x}^*$ is a basic feasible solution, there are n linearly independent constraints that are active at $\mathbf{x}^*$, and $\mathbf{x}^*$ is the unique solution to the system of equations $\mathbf{a}_i'\mathbf{x} = b_i$, $i \in I$ (Theorem 2.2). It follows that $\mathbf{x}^*$ is the unique minimizer of $\mathbf{c}'\mathbf{x}$ over the set P and, therefore, $\mathbf{x}^*$ is a vertex of P. □

Since a vector is a basic feasible solution if and only if it is an extreme point, and since the definition of an extreme point does not refer to any particular representation of a polyhedron, we conclude that the property of being a basic feasible solution is also independent of the representation used. (This is in contrast to the definition of a basic solution, which is representation dependent, as pointed out in the discussion that followed Definition 2.9.)

We finally note the following important fact.

Corollary 2.1 *Given a finite number of linear inequality constraints, there can only be a finite number of basic or basic feasible solutions.*

Proof. Consider a system of m linear inequality constraints imposed on a vector $\mathbf{x} \in \Re^n$. At any basic solution, there are n linearly independent active constraints. Since any n linearly independent active constraints define a unique point, it follows that different basic solutions correspond to different sets of n linearly independent active constraints. Therefore, the number of basic solutions is bounded above by the number of ways that we can choose n constraints out of a total of m, which is finite. □

Although the number of basic and, therefore, basic feasible solutions is guaranteed to be finite, it can be very large. For example, the unit cube $\{\mathbf{x} \in \Re^n \mid 0 \leq x_i \leq 1,\ i = 1, \ldots, n\}$ is defined in terms of $2n$ constraints, but has 2^n basic feasible solutions.

Adjacent basic solutions

Two distinct basic solutions to a set of linear constraints in $\Re^n$ are said to be *adjacent* if we can find $n-1$ linearly independent constraints that are active at both of them. In reference to Figure 2.7, D and E are adjacent to B; also, A and C are adjacent to D. If two adjacent basic solutions are also feasible, then the line segment that joins them is called an *edge* of the feasible set (see also Exercise 2.15).

2.3 Polyhedra in standard form

The definition of a basic solution (Definition 2.9) refers to general polyhedra. We will now specialize to polyhedra in standard form. The definitions and the results in this section are central to the development of the simplex method in the next chapter.

Let $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ be a polyhedron in standard form, and let the dimensions of $\mathbf{A}$ be $m \times n$, where m is the number of equality constraints. In most of our discussion of standard form problems, we will make the assumption that the m rows of the matrix $\mathbf{A}$ are linearly independent. (Since the rows are n-dimensional, this requires that $m \leq n$.) At the end of this section, we show that when P is nonempty, linearly dependent rows of $\mathbf{A}$ correspond to redundant constraints that can be discarded; therefore, our linear independence assumption can be made without loss of generality.

Recall that at any basic solution, there must be n linearly independent constraints that are active. Furthermore, every basic solution must satisfy the equality constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$, which provides us with m active constraints; these are linearly independent because of our assumption on the rows of $\mathbf{A}$. In order to obtain a total of n active constraints, we need to choose $n-m$ of the variables x_i and set them to zero, which makes the corresponding nonnegativity constraints $x_i \geq 0$ active. However, for the resulting set of n active constraints to be linearly independent, the choice of these $n-m$ variables is not entirely arbitrary, as shown by the following result.

Theorem 2.4 *Consider the constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$ and assume that the $m \times n$ matrix $\mathbf{A}$ has linearly independent rows. A vector $\mathbf{x} \in \Re^n$ is a basic solution if and only if we have $\mathbf{A}\mathbf{x} = \mathbf{b}$, and there exist indices $B(1), \ldots, B(m)$ such that:*

(a) *The columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$ are linearly independent;*

(b) *If $i \neq B(1), \ldots, B(m)$, then $x_i = 0$.*

Proof. Consider some $\mathbf{x} \in \Re^n$ and suppose that there are indices $B(1), \ldots,$

$B(m)$ that satisfy (a) and (b) in the statement of the theorem. The active constraints $x_i = 0$, $i \neq B(1), \ldots, B(m)$, and $\mathbf{Ax} = \mathbf{b}$ imply that

$$\sum_{i=1}^{m} \mathbf{A}_{B(i)} x_{B(i)} = \sum_{i=1}^{n} \mathbf{A}_i x_i = \mathbf{Ax} = \mathbf{b}.$$

Since the columns $\mathbf{A}_{B(i)}$, $i = 1, \ldots, m$, are linearly independent, $x_{B(1)}, \ldots, x_{B(m)}$ are uniquely determined. Thus, the system of equations formed by the active constraints has a unique solution. By Theorem 2.2, there are n linearly independent active constraints, and this implies that $\mathbf{x}$ is a basic solution.

For the converse, we assume that $\mathbf{x}$ is a basic solution and we will show that conditions (a) and (b) in the statement of the theorem are satisfied. Let $x_{B(1)}, \ldots, x_{B(k)}$ be the components of $\mathbf{x}$ that are nonzero. Since $\mathbf{x}$ is a basic solution, the system of equations formed by the active constraints $\sum_{i=1}^{n} \mathbf{A}_i x_i = \mathbf{b}$ and $x_i = 0$, $i \neq B(1), \ldots, B(k)$, have a unique solution (cf. Theorem 2.2); equivalently, the equation $\sum_{i=1}^{k} \mathbf{A}_{B(i)} x_{B(i)} = \mathbf{b}$ has a unique solution. It follows that the columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(k)}$ are linearly independent. [If they were not, we could find scalars $\lambda_1, \ldots, \lambda_k$, not all of them zero, such that $\sum_{i=1}^{k} \mathbf{A}_{B(i)} \lambda_i = 0$. This would imply that $\sum_{i=1}^{k} \mathbf{A}_{B(i)} (x_{B(i)} + \lambda_i) = \mathbf{b}$, contradicting the uniqueness of the solution.]

We have shown that the columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(k)}$ are linearly independent and this implies that $k \leq m$. Since $\mathbf{A}$ has m linearly independent rows, it also has m linearly independent columns, which span $\Re^m$. It follows [cf. Theorem 1.3(b) in Section 1.5] that we can find $m-k$ additional columns $\mathbf{A}_{B(k+1)}, \ldots, \mathbf{A}_{B(m)}$ so that the columns $\mathbf{A}_{B(i)}$, $i = 1, \ldots, m$, are linearly independent. In addition, if $i \neq B(1), \ldots, B(m)$, then $i \neq B(1), \ldots, B(k)$ (because $k \leq m$), and $x_i = 0$. Therefore, both conditions (a) and (b) in the statement of the theorem are satisfied. □

In view of Theorem 2.4, all basic solutions to a standard form polyhedron can be constructed according to the following procedure.

Procedure for constructing basic solutions

1. Choose m linearly independent columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$.
2. Let $x_i = 0$ for all $i \neq B(1), \ldots, B(m)$.
3. Solve the system of m equations $\mathbf{Ax} = \mathbf{b}$ for the unknowns $x_{B(1)}, \ldots, x_{B(m)}$.

If a basic solution constructed according to this procedure is nonnegative, then it is feasible, and it is a basic feasible solution. Conversely, since every basic feasible solution is a basic solution, it can be obtained from this procedure. If $\mathbf{x}$ is a basic solution, the variables $x_{B(1)}, \ldots, x_{B(m)}$ are called

basic variables; the remaining variables are called *nonbasic*. The columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$ are called the *basic columns* and, since they are linearly independent, they form a *basis* of $\Re^m$. We will sometimes talk about two bases being *distinct* or *different*; our convention is that distinct bases involve different sets $\{B(1), \ldots, B(m)\}$ of *basic indices*; if two bases involve the same set of indices in a different order, they will be viewed as one and the same basis.

By arranging the m basic columns next to each other, we obtain an $m \times m$ matrix $\mathbf{B}$, called a *basis matrix*. (Note that this matrix is invertible because the basic columns are required to be linearly independent.) We can similarly define a vector $\mathbf{x}_B$ with the values of the basic variables. Thus,

$$\mathbf{B} = \begin{bmatrix} | & | & & | \\ \mathbf{A}_{B(1)} & \mathbf{A}_{B(2)} & \cdots & \mathbf{A}_{B(m)} \\ | & | & & | \end{bmatrix}, \qquad \mathbf{x}_B = \begin{bmatrix} x_{B(1)} \\ \vdots \\ x_{B(m)} \end{bmatrix}.$$

The basic variables are determined by solving the equation $\mathbf{B}\mathbf{x}_B = \mathbf{b}$ whose unique solution is given by

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}.$$

Example 2.1 Let the constraint $\mathbf{A}\mathbf{x} = \mathbf{b}$ be of the form

$$\begin{bmatrix} 1 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 6 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 8 \\ 12 \\ 4 \\ 6 \end{bmatrix}.$$

Let us choose $\mathbf{A}_4, \mathbf{A}_5, \mathbf{A}_6, \mathbf{A}_7$ as our basic columns. Note that they are linearly independent and the corresponding basis matrix is the identity. We then obtain the basic solution $\mathbf{x} = (0, 0, 0, 8, 12, 4, 6)$ which is nonnegative and, therefore, is a basic feasible solution. Another basis is obtained by choosing the columns $\mathbf{A}_3, \mathbf{A}_5, \mathbf{A}_6, \mathbf{A}_7$ (note that they are linearly independent). The corresponding basic solution is $\mathbf{x} = (0, 0, 4, 0, -12, 4, 6)$, which is not feasible because $x_5 = -12 < 0$.

Suppose now that there was an eighth column $\mathbf{A}_8$, identical to $\mathbf{A}_7$. Then, the two sets of columns $\{\mathbf{A}_3, \mathbf{A}_5, \mathbf{A}_6, \mathbf{A}_7\}$ and $\{\mathbf{A}_3, \mathbf{A}_5, \mathbf{A}_6, \mathbf{A}_8\}$ coincide. On the other hand the corresponding sets of basic indices, which are $\{3, 5, 6, 7\}$ and $\{3, 5, 6, 8\}$, are different and we have two different bases, according to our conventions.

For an intuitive view of basic solutions, recall our interpretation of the constraint $\mathbf{A}\mathbf{x} = \mathbf{b}$, or $\sum_{i=1}^{n} \mathbf{A}_i x_i = \mathbf{b}$, as a requirement to synthesize the vector $\mathbf{b} \in \Re^m$ using the resource vectors $\mathbf{A}_i$ (Section 1.1). In a basic solution, we use only m of the resource vectors, those associated with the basic variables. Furthermore, in a basic feasible solution, this is accomplished using a nonnegative amount of each basic vector; see Figure 2.8.

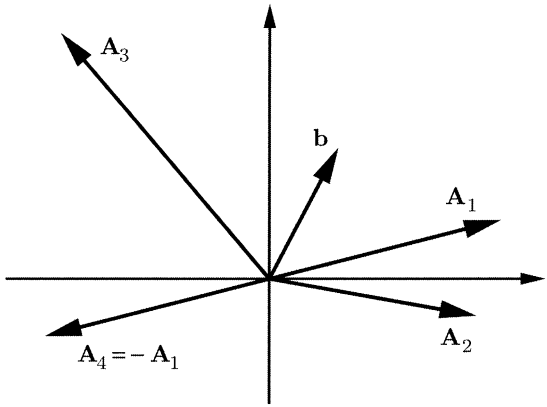


Figure 2.8: Consider a standard form problem with $n = 4$ and $m = 2$, and let the vectors $\mathbf{b}, \mathbf{A}_1, \ldots, \mathbf{A}_4$ be as shown. The vectors $\mathbf{A}_1, \mathbf{A}_2$ form a basis; the corresponding basic solution is infeasible because a negative value of x_2 is needed to synthesize $\mathbf{b}$ from $\mathbf{A}_1$, $\mathbf{A}_2$. The vectors $\mathbf{A}_1, \mathbf{A}_3$ form another basis; the corresponding basic solution is feasible. Finally, the vectors $\mathbf{A}_1, \mathbf{A}_4$ do not form a basis because they are linearly dependent.

Correspondence of bases and basic solutions

We now elaborate on the correspondence between basic solutions and bases. Different basic solutions must correspond to different bases, because a basis uniquely determines a basic solution. However, two different bases may lead to the same basic solution. (For an extreme example, if we have $\mathbf{b} = \mathbf{0}$, then every basis matrix leads to the same basic solution, namely, the zero vector.) This phenomenon has some important algorithmic implications, and is closely related to degeneracy, which is the subject of the next section.

Adjacent basic solutions and adjacent bases

Recall that two distinct basic solutions are said to be adjacent if there are $n - 1$ linearly independent constraints that are active at both of them. For standard form problems, we also say that two bases are *adjacent* if they share all but one basic column. Then, it is not hard to check that adjacent basic solutions can always be obtained from two adjacent bases. Conversely, if two adjacent bases lead to distinct basic solutions, then the latter are adjacent.

Example 2.2 In reference to Example 2.1, the bases $\{\mathbf{A}_4, \mathbf{A}_5, \mathbf{A}_6, \mathbf{A}_7\}$ and $\{\mathbf{A}_3, \mathbf{A}_5, \mathbf{A}_6, \mathbf{A}_7\}$ are adjacent because all but one columns are the same. The corresponding basic solutions $\mathbf{x} = (0, 0, 0, 8, 12, 4, 6)$ and $\mathbf{x} = (0, 0, 4, 0, -12, 4, 6)$

are adjacent: we have $n = 7$ and a total of six common linearly independent active constraints; these are $x_1 \geq 0$, $x_2 \geq 0$, and the four equality constraints.

The full row rank assumption on A

We close this section by showing that the full row rank assumption on the matrix $\mathbf{A}$ results in no loss of generality.

Theorem 2.5 *Let $P = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ be a nonempty polyhedron, where $\mathbf{A}$ is a matrix of dimensions $m \times n$, with rows $\mathbf{a}_1', \ldots, \mathbf{a}_m'$. Suppose that $\text{rank}(\mathbf{A}) = k < m$ and that the rows $\mathbf{a}_{i_1}', \ldots, \mathbf{a}_{i_k}'$ are linearly independent. Consider the polyhedron*

$$Q = \left\{\mathbf{x} \mid \mathbf{a}_{i_1}'\mathbf{x} = b_{i_1}, \ldots, \mathbf{a}_{i_k}'\mathbf{x} = b_{i_k},\ \mathbf{x} \geq \mathbf{0}\right\}.$$

Then $Q = P$.

Proof. We provide the proof for the case where $i_1 = 1, \ldots, i_k = k$, that is, the first k rows of $\mathbf{A}$ are linearly independent. The general case can be reduced to this one by rearranging the rows of $\mathbf{A}$.

Clearly $P \subset Q$ since any element of P automatically satisfies the constraints defining Q. We will now show that $Q \subset P$.

Since $\text{rank}(\mathbf{A}) = k$, the row space of $\mathbf{A}$ has dimension k and the rows $\mathbf{a}_1', \ldots, \mathbf{a}_k'$ form a basis of the row space. Therefore, every row $\mathbf{a}_i'$ of $\mathbf{A}$ can be expressed in the form $\mathbf{a}_i' = \sum_{j=1}^{k} \lambda_{ij}\mathbf{a}_j'$, for some scalars λ_{ij}. Let $\mathbf{x}$ be an element of P and note that

$$b_i = \mathbf{a}_i'\mathbf{x} = \sum_{j=1}^{k} \lambda_{ij}\mathbf{a}_j'\mathbf{x} = \sum_{j=1}^{k} \lambda_{ij}b_j, \qquad i = 1, \ldots, m.$$

Consider now an element $\mathbf{y}$ of Q. We will show that it belongs to P. Indeed, for any i,

$$\mathbf{a}_i'\mathbf{y} = \sum_{j=1}^{k} \lambda_{ij}\mathbf{a}_j'\mathbf{y} = \sum_{j=1}^{k} \lambda_{ij}b_j = b_i,$$

which establishes that $\mathbf{y} \in P$ and $Q \subset P$. □

Notice that the polyhedron Q in Theorem 2.5 is in standard form; namely, $Q = \{\mathbf{x} \mid \mathbf{D}\mathbf{x} = \mathbf{f},\ \mathbf{x} \geq \mathbf{0}\}$ where $\mathbf{D}$ is a $k \times n$ submatrix of $\mathbf{A}$, with rank equal to k, and $\mathbf{f}$ is a k-dimensional subvector of $\mathbf{b}$. We conclude that as long as the feasible set is nonempty, a linear programming problem in standard form can be reduced to an equivalent standard form problem (with the same feasible set) in which the equality constraints are linearly independent.

Example 2.3 Consider the (nonempty) polyhedron defined by the constraints

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

The corresponding matrix $\mathbf{A}$ has rank two. This is because the last two rows $(1,1,0)$ and $(1,0,1)$ are linearly independent, but the first row is equal to the sum of the other two. Thus, the first constraint is redundant and after it is eliminated, we still have the same polyhedron.

2.4 Degeneracy

According to our definition, at a basic solution, we must have n linearly independent active constraints. This allows for the possibility that the number of active constraints is greater than n. (Of course, in n dimensions, no more than n of them can be linearly independent.) In this case, we say that we have a *degenerate* basic solution. In other words, at a degenerate basic solution, the number of active constraints is greater than the minimum necessary.

Definition 2.10 *A basic solution $\mathbf{x} \in \Re^n$ is said to be* **degenerate** *if more than n of the constraints are active at $\mathbf{x}$.*

In two dimensions, a degenerate basic solution is at the intersection of three or more lines; in three dimensions, a degenerate basic solution is at the intersection of four or more planes; see Figure 2.9 for an illustration. It turns out that the presence of degeneracy can strongly affect the behavior of linear programming algorithms and for this reason, we will now develop some more intuition.

Example 2.4 Consider the polyhedron P defined by the constraints

$$\begin{aligned} x_1 + x_2 + 2x_3 &\leq 8 \\ x_2 + 6x_3 &\leq 12 \\ x_1 &\leq 4 \\ x_2 &\leq 6 \\ x_1, x_2, x_3 &\geq 0. \end{aligned}$$

The vector $\mathbf{x} = (2,6,0)$ is a nondegenerate basic feasible solution, because there are exactly three active and linearly independent constraints, namely, $x_1 + x_2 + 2x_3 \leq 8$, $x_2 \leq 6$, and $x_3 \geq 0$. The vector $\mathbf{x} = (4,0,2)$ is a degenerate basic feasible solution, because there are four active constraints, three of them linearly independent, namely, $x_1 + x_2 + 2x_3 \leq 8$, $x_2 + 6x_3 \leq 12$, $x_1 \leq 4$, and $x_2 \geq 0$.

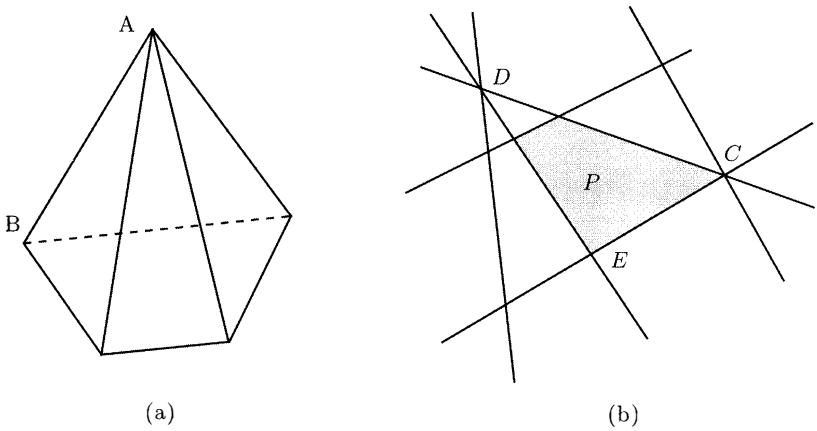


Figure 2.9: The points A and C are degenerate basic feasible solutions. The points B and E are nondegenerate basic feasible solutions. The point D is a degenerate basic solution.

Degeneracy in standard form polyhedra

At a basic solution of a polyhedron in standard form, the m equality constraints are always active. Therefore, having more than n active constraints is the same as having more than $n - m$ variables at zero level. This leads us to the next definition which is a special case of Definition 2.10.

Definition 2.11 *Consider the standard form polyhedron* $P = \{\mathbf{x} \in \Re^n \mid \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ *and let* $\mathbf{x}$ *be a basic solution. Let* m *be the number of rows of* $\mathbf{A}$*. The vector* $\mathbf{x}$ *is a* **degenerate** *basic solution if more than* $n - m$ *of the components of* $\mathbf{x}$ *are zero.*

Example 2.5 Consider once more the polyhedron of Example 2.4. By introducing the slack variables $x_4, \ldots, x_7$, we can transform it into the standard form $P = \{\mathbf{x} = (x_1, \ldots, x_7) \mid \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$, where

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 6 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 8 \\ 12 \\ 4 \\ 6 \end{bmatrix}.$$

Consider the basis consisting of the linearly independent columns $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$, $\mathbf{A}_7$. To calculate the corresponding basic solution, we first set the nonbasic variables x_4, x_5, and x_6 to zero, and then solve the system $\mathbf{Ax} = \mathbf{b}$ for the remaining variables, to obtain $\mathbf{x} = (4, 0, 2, 0, 0, 0, 6)$. This is a degenerate basic feasible solution, because we have a total of four variables that are zero, whereas

$n - m = 7 - 4 = 3$. Thus, while we initially set only the three nonbasic variables to zero, the solution to the system $\mathbf{Ax} = \mathbf{b}$ turned out to satisfy one more of the constraints (namely, the constraint $x_2 \geq 0$) with equality. Consider now the basis consisting of the linearly independent columns $\mathbf{A}_1$, $\mathbf{A}_3$, $\mathbf{A}_4$, and $\mathbf{A}_7$. The corresponding basic feasible solution is again $\mathbf{x} = (4, 0, 2, 0, 0, 0, 6)$.

The preceding example suggests that we can think of degeneracy in the following terms. We pick a basic solution by picking n linearly independent constraints to be satisfied with equality, and we realize that certain other constraints are also satisfied with equality. If the entries of $\mathbf{A}$ or $\mathbf{b}$ were chosen at random, this would almost never happen. Also, Figure 2.10 illustrates that if the coefficients of the active constraints are slightly perturbed, degeneracy can disappear (cf. Exercise 2.18). In practical problems, however, the entries of $\mathbf{A}$ and $\mathbf{b}$ often have a special (nonrandom) structure, and degeneracy is more common than the preceding argument would seem to suggest.

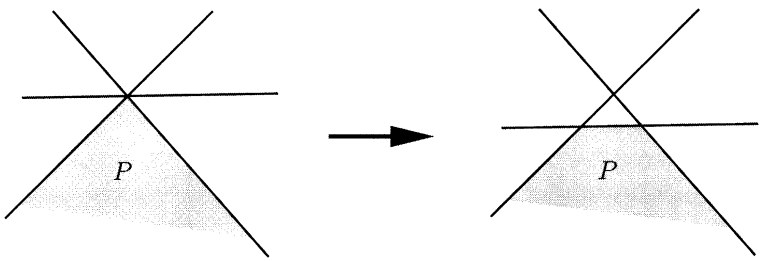


Figure 2.10: Small changes in the constraining inequalities can remove degeneracy.

In order to visualize degeneracy in standard form polyhedra, we assume that $n - m = 2$ and we draw the feasible set as a subset of the two-dimensional set defined by the equality constraints $\mathbf{Ax} = \mathbf{b}$; see Figure 2.11. At a nondegenerate basic solution, exactly $n-m$ of the constraints $x_i \geq 0$ are active; the corresponding variables are nonbasic. In the case of a degenerate basic solution, more than $n - m$ of the constraints $x_i \geq 0$ are active, and there are usually several ways of choosing which $n-m$ variables to call nonbasic; in that case, there are several bases corresponding to that same basic solution. (This discussion refers to the typical case. However, there are examples of degenerate basic solutions to which there corresponds only one basis.)

Degeneracy is not a purely geometric property

We close this section by pointing out that degeneracy of basic feasible solutions is not, in general, a geometric (representation independent) property,

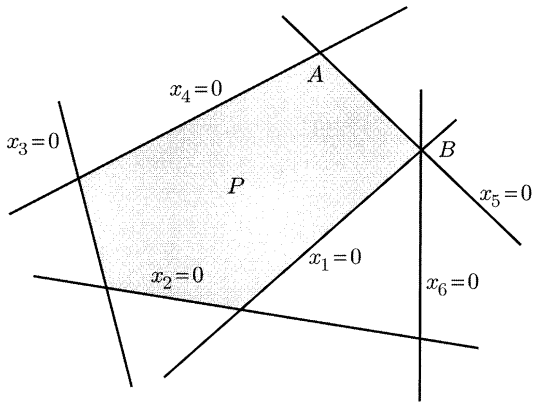


Figure 2.11: An $(n-m)$-dimensional illustration of degeneracy. Here, $n=6$ and $m=4$. The basic feasible solution A is nondegenerate and the basic variables are x_1, x_2, x_3, x_6. The basic feasible solution B is degenerate. We can choose x_1, x_6 as the nonbasic variables. Other possibilities are to choose x_1, x_5, or to choose x_5, x_6. Thus, there are three possible bases, for the same basic feasible solution B.

but rather it may depend on the particular representation of a polyhedron. To illustrate this point, consider the standard form polyhedron (cf. Figure 2.12)

$$P = \Big\{(x_1, x_2, x_3) \mid x_1 - x_2 = 0,\ x_1 + x_2 + 2x_3 = 2,\ x_1, x_2, x_3 \geq 0\Big\}.$$

We have $n=3$, $m=2$ and $n-m=1$. The vector $(1,1,0)$ is nondegenerate because only one variable is zero. The vector (0,0,1) is degenerate because two variables are zero. However, the same polyhedron can also be described

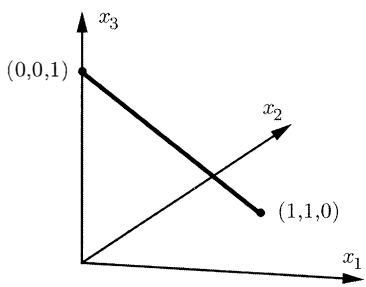


Figure 2.12: An example of degeneracy in a standard form problem.

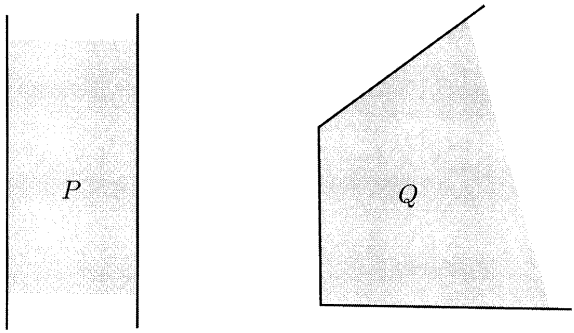


Figure 2.13: The polyhedron P contains a line and does not have an extreme point, while Q does not contain a line and has extreme points.

in the (nonstandard) form

$$P = \Big\{(x_1, x_2, x_3) \mid x_1 - x_2 = 0,\ x_1 + x_2 + 2x_3 = 2,\ x_1 \geq 0,\ x_3 \geq 0\Big\}.$$

The vector (0,0,1) is now a nondegenerate basic feasible solution, because there are only three active constraints.

For another example, consider a nondegenerate basic feasible solution $\mathbf{x}^*$ of a standard form polyhedron $P = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$, where $\mathbf{A}$ is of dimensions $m \times n$. In particular, exactly $n - m$ of the variables x_i^* are equal to zero. Let us now represent P in the form $P = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq \mathbf{b},\ -\mathbf{A}\mathbf{x} \geq -\mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$. Then, at the basic feasible solution $\mathbf{x}^*$, we have $n - m$ variables set to zero and an additional $2m$ inequality constraints are satisfied with equality. We therefore have $n + m$ active constraints and $\mathbf{x}^*$ is degenerate. Hence, under the second representation, every basic feasible solution is degenerate.

We have established that a degenerate basic feasible solution under one representation could be nondegenerate under another representation. Still, it can be shown that if a basic feasible solution is degenerate under one particular standard form representation, then it is degenerate under every standard form representation of the same polyhedron (Exercise 2.19).

2.5 Existence of extreme points

We obtain in this section necessary and sufficient conditions for a polyhedron to have at least one extreme point. We first observe that not every polyhedron has this property. For example, if $n > 1$, a halfspace in $\Re^n$ is a polyhedron without extreme points. Also, as argued in Section 2.2 (cf. the discussion after Definition 2.9), if the matrix $\mathbf{A}$ has fewer than n rows, then the polyhedron $\{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ does not have a basic feasible solution.

It turns out that the existence of an extreme point depends on whether a polyhedron contains an infinite line or not; see Figure 2.13. We need the following definition.

Definition 2.12 *A polyhedron $P \subset \Re^n$* **contains a line** *if there exists a vector $\mathbf{x} \in P$ and a nonzero vector $\mathbf{d} \in \Re^n$ such that $\mathbf{x} + \lambda \mathbf{d} \in P$ for all scalars λ.*

We then have the following result.

Theorem 2.6 *Suppose that the polyhedron $P = \{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \geq b_i,\ i = 1, \ldots, m\}$ is nonempty. Then, the following are equivalent:*

(a) *The polyhedron P has at least one extreme point.*

(b) *The polyhedron P does not contain a line.*

(c) *There exist n vectors out of the family $\mathbf{a}_1, \ldots, \mathbf{a}_m$, which are linearly independent.*

Proof.

(b) $\Rightarrow$ (a)

We first prove that if P does not contain a line, then it has a basic feasible solution and, therefore, an extreme point. A geometric interpretation of this proof is provided in Figure 2.14.

Let $\mathbf{x}$ be an element of P and let $I = \{i \mid \mathbf{a}_i'\mathbf{x} = b_i\}$. If n of the vectors $\mathbf{a}_i$, $i \in I$, corresponding to the active constraints are linearly independent, then $\mathbf{x}$ is, by definition, a basic feasible solution and, therefore, a basic feasible solution exists. If this is not the case, then all of the vectors $\mathbf{a}_i$, $i \in I$, lie in a proper subspace of $\Re^n$ and there exists a nonzero vector $\mathbf{d} \in \Re^n$ such that $\mathbf{a}_i'\mathbf{d} = 0$, for every $i \in I$. Let us consider the line consisting of all points of the form $\mathbf{y} = \mathbf{x} + \lambda \mathbf{d}$, where λ is an arbitrary scalar. For $i \in I$, we have $\mathbf{a}_i'\mathbf{y} = \mathbf{a}_i'\mathbf{x} + \lambda \mathbf{a}_i'\mathbf{d} = \mathbf{a}_i'\mathbf{x} = b_i$. Thus, those constraints that were active at $\mathbf{x}$ remain active at all points on the line. However, since the polyhedron is assumed to contain no lines, it follows that as we vary λ, some constraint will be eventually violated. At the point where some constraint is about to be violated, a new constraint must become active, and we conclude that there exists some λ^* and some $j \notin I$ such that $\mathbf{a}_j'(\mathbf{x} + \lambda^* \mathbf{d}) = b_j$.

We claim that $\mathbf{a}_j$ is not a linear combination of the vectors $\mathbf{a}_i$, $i \in I$. Indeed, we have $\mathbf{a}_j'\mathbf{x} \neq b_j$ (because $j \notin I$) and $\mathbf{a}_j'(\mathbf{x} + \lambda^* \mathbf{d}) = b_j$ (by the definition of λ^*). Thus, $\mathbf{a}_j'\mathbf{d} \neq 0$. On the other hand, $\mathbf{a}_i'\mathbf{d} = 0$ for every $i \in I$ (by the definition of $\mathbf{d}$) and therefore, $\mathbf{d}$ is orthogonal to any linear combination of the vectors $\mathbf{a}_i$, $i \in I$. Since $\mathbf{d}$ is not orthogonal to $\mathbf{a}_j$, we

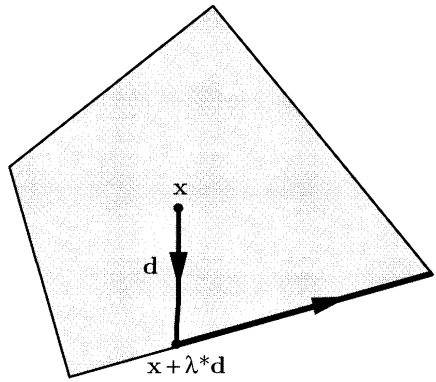


Figure 2.14: Starting from an arbitrary point of a polyhedron, we choose a direction along which all currently active constraints remain active. We then move along that direction until a new constraint is about to be violated. At that point, the number of linearly independent active constraints has increased by at least one. We repeat this procedure until we end up with n linearly independent active constraints, at which point we have a basic feasible solution.

conclude that $\mathbf{a}_j$ is a not a linear combination of the vectors $\mathbf{a}_i$, $i \in I$. Thus, by moving from $\mathbf{x}$ to $\mathbf{x} + \lambda^*\mathbf{d}$, the number of linearly independent active constraints has been increased by at least one. By repeating the same argument, as many times as needed, we eventually end up with a point at which there are n linearly independent active constraints. Such a point is, by definition, a basic solution; it is also feasible since we have stayed within the feasible set.

(a) $\Rightarrow$ (c)

If P has an extreme point $\mathbf{x}$, then $\mathbf{x}$ is also a basic feasible solution (cf. Theorem 2.3), and there exist n constraints that are active at $\mathbf{x}$, with the corresponding vectors $\mathbf{a}_i$ being linearly independent.

(c) $\Rightarrow$ (b)

Suppose that n of the vectors $\mathbf{a}_i$ are linearly independent and, without loss of generality, let us assume that $\mathbf{a}_1, \ldots, \mathbf{a}_n$ are linearly independent. Suppose that P contains a line $\mathbf{x} + \lambda\mathbf{d}$, where $\mathbf{d}$ is a nonzero vector. We then have $\mathbf{a}_i'(\mathbf{x} + \lambda\mathbf{d}) \geq b_i$ for all i and all λ. We conclude that $\mathbf{a}_i'\mathbf{d} = 0$ for all i. (If $\mathbf{a}_i'\mathbf{d} < 0$, we can violate the constraint by picking λ very large; a symmetric argument applies if $\mathbf{a}_i'\mathbf{d} > 0$.) Since the vectors $\mathbf{a}_i$, $i = 1, \ldots, n$, are linearly independent, this implies that $\mathbf{d} = \mathbf{0}$. This is a contradiction and establishes that P does not contain a line. □

Notice that a bounded polyhedron does not contain a line. Similarly,

the positive orthant $\{\mathbf{x} \mid \mathbf{x} \geq \mathbf{0}\}$ does not contain a line. Since a polyhedron in standard form is contained in the positive orthant, it does not contain a line either. These observations establish the following important corollary of Theorem 2.6.

Corollary 2.2 *Every nonempty bounded polyhedron and every nonempty polyhedron in standard form has at least one basic feasible solution.*

2.6 Optimality of extreme points

Having established the conditions for the existence of extreme points, we will now confirm the intuition developed in Chapter 1: as long as a linear programming problem has an optimal solution and as long as the feasible set has at least one extreme point, we can always find an optimal solution within the set of extreme points of the feasible set. Later in this section, we prove a somewhat stronger result, at the expense of a more complicated proof.

Theorem 2.7 *Consider the linear programming problem of minimizing* $\mathbf{c}'\mathbf{x}$ *over a polyhedron* P*. Suppose that* P *has at least one extreme point and that there exists an optimal solution. Then, there exists an optimal solution which is an extreme point of* P*.*

Proof. (See Figure 2.15 for an illustration.) Let Q be the set of all optimal solutions, which we have assumed to be nonempty. Let P be of the form $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ and let v be the optimal value of the cost $\mathbf{c}'\mathbf{x}$. Then, $Q = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b},\ \mathbf{c}'\mathbf{x} = v\}$, which is also a polyhedron. Since

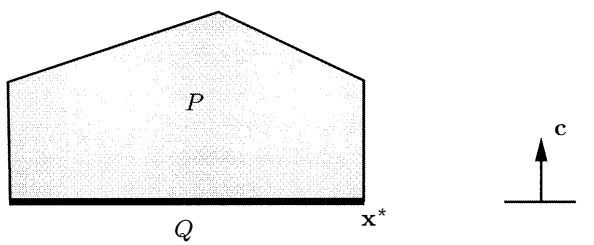


Figure 2.15: Illustration of the proof of Theorem 2.7. Here, Q is the set of optimal solutions and an extreme point $\mathbf{x}^*$ of Q is also an extreme point of P.

$Q \subset P$, and since P contains no lines (cf. Theorem 2.6), Q contains no lines either. Therefore, Q has an extreme point.

Let $\mathbf{x}^*$ be an extreme point of Q. We will show that $\mathbf{x}^*$ is also an extreme point of P. Suppose, in order to derive a contradiction, that $\mathbf{x}^*$ is not an extreme point of P. Then, there exist $\mathbf{y} \in P$, $\mathbf{z} \in P$, such that $\mathbf{y} \neq \mathbf{x}^*$, $\mathbf{z} \neq \mathbf{x}^*$, and some $\lambda \in [0,1]$ such that $\mathbf{x}^* = \lambda\mathbf{y} + (1-\lambda)\mathbf{z}$. It follows that $v = \mathbf{c}'\mathbf{x}^* = \lambda\mathbf{c}'\mathbf{y} + (1-\lambda)\mathbf{c}'\mathbf{z}$. Furthermore, since v is the optimal cost, $\mathbf{c}'\mathbf{y} \geq v$ and $\mathbf{c}'\mathbf{z} \geq v$. This implies that $\mathbf{c}'\mathbf{y} = \mathbf{c}'\mathbf{z} = v$ and therefore $\mathbf{z} \in Q$ and $\mathbf{y} \in Q$. But this contradicts the fact that $\mathbf{x}^*$ is an extreme point of Q. The contradiction establishes that $\mathbf{x}^*$ is an extreme point of P. In addition, since $\mathbf{x}^*$ belongs to Q, it is optimal. □

The above theorem applies to polyhedra in standard form, as well as to bounded polyhedra, since they do not contain a line.

Our next result is stronger than Theorem 2.7. It shows that the existence of an optimal solution can be taken for granted, as long as the optimal cost is finite.

Theorem 2.8 *Consider the linear programming problem of minimizing* $\mathbf{c}'\mathbf{x}$ *over a polyhedron* P. *Suppose that* P *has at least one extreme point. Then, either the optimal cost is equal to* $-\infty$, *or there exists an extreme point which is optimal.*

Proof. The proof is essentially a repetition of the proof of Theorem 2.6. The difference is that as we move towards a basic feasible solution, we will also make sure that the costs do not increase. We will use the following terminology: an element $\mathbf{x}$ of P has *rank* k if we can find k, but not more than k, linearly independent constraints that are active at $\mathbf{x}$.

Let us assume that the optimal cost is finite. Let $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ and consider some $\mathbf{x} \in P$ of rank $k < n$. We will show that there exists some $\mathbf{y} \in P$ which has greater rank and satisfies $\mathbf{c}'\mathbf{y} \leq \mathbf{c}'\mathbf{x}$. Let $I = \{i \mid \mathbf{a}_i'\mathbf{x} = b_i\}$, where $\mathbf{a}_i'$ is the ith row of $\mathbf{A}$. Since $k < n$, the vectors $\mathbf{a}_i$, $i \in I$, lie in a proper subspace of $\Re^n$, and we can choose some nonzero $\mathbf{d} \in \Re^n$ orthogonal to every $\mathbf{a}_i$, $i \in I$. Furthermore, by possibly taking the negative of $\mathbf{d}$, we can assume that $\mathbf{c}'\mathbf{d} \leq 0$.

Suppose that $\mathbf{c}'\mathbf{d} < 0$. Let us consider the half-line $\mathbf{y} = \mathbf{x} + \lambda\mathbf{d}$, where λ is a positive scalar. As in the proof of Theorem 2.6, all points on this half-line satisfy the relations $\mathbf{a}_i'\mathbf{y} = b_i$, $i \in I$. If the entire half-line were contained in P, the optimal cost would be $-\infty$, which we have assumed not to be the case. Therefore, the half-line eventually exits P. When this is about to happen, we have some $\lambda^* > 0$ and $j \notin I$ such that $\mathbf{a}_j'(\mathbf{x} + \lambda^*\mathbf{d}) = b_j$. We let $\mathbf{y} = \mathbf{x} + \lambda^*\mathbf{d}$ and note that $\mathbf{c}'\mathbf{y} < \mathbf{c}'\mathbf{x}$. As in the proof of Theorem 2.6, $\mathbf{a}_j$ is linearly independent from $\mathbf{a}_i$, $i \in I$, and the rank of $\mathbf{y}$ is at least $k+1$.

Suppose now that $\mathbf{c}'\mathbf{d} = 0$. We consider the line $\mathbf{y} = \mathbf{x} + \lambda\mathbf{d}$, where λ is an arbitrary scalar. Since P contains no lines, the line must eventually exit P and when that is about to happen, we are again at a vector $\mathbf{y}$ of rank greater than that of $\mathbf{x}$. Furthermore, since $\mathbf{c}'\mathbf{d} = 0$, we have $\mathbf{c}'\mathbf{y} = \mathbf{c}'\mathbf{x}$.

In either case, we have found a new point $\mathbf{y}$ such that $\mathbf{c}'\mathbf{y} \leq \mathbf{c}'\mathbf{x}$, and whose rank is greater than that of $\mathbf{x}$. By repeating this process as many times as needed, we end up with a vector $\mathbf{w}$ of rank n (thus, $\mathbf{w}$ is a basic feasible solution) such that $\mathbf{c}'\mathbf{w} \leq \mathbf{c}'\mathbf{x}$.

Let $\mathbf{w}^1, \ldots, \mathbf{w}^r$ be the basic feasible solutions in P and let $\mathbf{w}^*$ be a basic feasible solution such that $\mathbf{c}'\mathbf{w}^* \leq \mathbf{c}'\mathbf{w}^i$ for all i. We have already shown that for every $\mathbf{x}$ there exists some i such that $\mathbf{c}'\mathbf{w}^i \leq \mathbf{c}'\mathbf{x}$. It follows that $\mathbf{c}'\mathbf{w}^* \leq \mathbf{c}'\mathbf{x}$ for all $\mathbf{x} \in P$, and the basic feasible solution $\mathbf{w}^*$ is optimal. □

For a general linear programming problem, if the feasible set has no extreme points, then Theorem 2.8 does not apply directly. On the other hand, any linear programming problem can be transformed into an equivalent problem in standard form to which Theorem 2.8 does apply. This establishes the following corollary.

Corollary 2.3 *Consider the linear programming problem of minimizing $\mathbf{c}'\mathbf{x}$ over a nonempty polyhedron. Then, either the optimal cost is equal to $-\infty$ or there exists an optimal solution.*

The result in Corollary 2.3 should be contrasted with what may happen in optimization problems with a nonlinear cost function. For example, in the problem of minimizing $1/x$ subject to $x \geq 1$, the optimal cost is not $-\infty$, but an optimal solution does not exist.

2.7 Representation of bounded polyhedra*

So far, we have been representing polyhedra in terms of their defining inequalities. In this section, we provide an alternative, by showing that a bounded polyhedron can also be represented as the convex hull of its extreme points. The proof that we give here is elementary and constructive, and its main idea is summarized in Figure 2.16. There is a similar representation of unbounded polyhedra involving extreme points and "extreme rays" (edges that extend to infinity). This representation can be developed using the tools that we already have, at the expense of a more complicated proof. A more elegant argument, based on duality theory, will be presented in Section 4.9 and will also result in an alternative proof of Theorem 2.9 below.

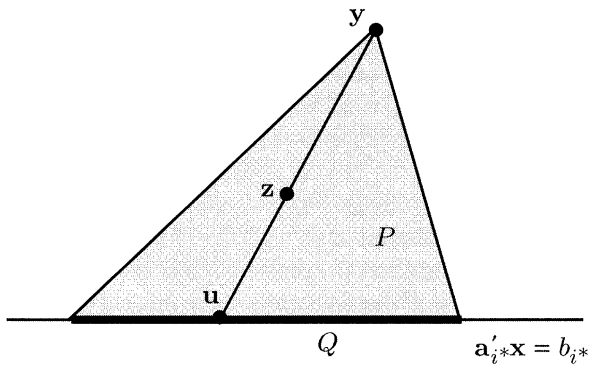


Figure 2.16: Given the vector $\mathbf{z}$, we express it as a convex combination of $\mathbf{y}$ and $\mathbf{u}$. The vector $\mathbf{u}$ belongs to the polyhedron Q whose dimension is lower than that of P. Using induction on dimension, we can express the vector $\mathbf{u}$ as a convex combination of extreme points of Q. These are also extreme points of P.

Theorem 2.9 *A nonempty and bounded polyhedron is the convex hull of its extreme points.*

Proof. Every convex combination of extreme points is an element of the polyhedron, since polyhedra are convex sets. Thus, we only need to prove the converse result and show that every element of a bounded polyhedron can be represented as a convex combination of extreme points.

We define the *dimension* of a polyhedron $P \subset \Re^n$ as the smallest integer k such that P is contained in some k-dimensional affine subspace of $\Re^n$. (Recall from Section 1.5, that a k-dimensional affine subspace is a translation of a k-dimensional subspace.) Our proof proceeds by induction on the dimension of the polyhedron P. If P is zero-dimensional, it consists of a single point. This point is an extreme point of P and the result is true.

Let us assume that the result is true for all polyhedra of dimension less than k. Let $P = \{\mathbf{x} \in \Re^n \mid \mathbf{a}'_i\mathbf{x} \geq b_i,\ i = 1, \ldots, m\}$ be a nonempty bounded k-dimensional polyhedron. Then, P is contained in a k-dimensional affine subspace S of $\Re^n$, which can be assumed to be of the form

$$S = \{\mathbf{x}^0 + \lambda_1\mathbf{x}^1 + \cdots + \lambda_k\mathbf{x}^k \mid \lambda_1, \ldots, \lambda_k \in \Re\},$$

where $\mathbf{x}^1, \ldots, \mathbf{x}^k$ are some vectors in $\Re^n$. Let $\mathbf{f}_1, \ldots, \mathbf{f}_{n-k}$ be $n-k$ linearly independent vectors that are orthogonal to $\mathbf{x}^1, \ldots, \mathbf{x}^k$. Let $g_i = \mathbf{f}'_i\mathbf{x}^0$, for

$i = 1, \ldots, n-k$. Then, every element $\mathbf{x}$ of S satisfies

$$\mathbf{f}_i'\mathbf{x} = g_i, \qquad i = 1, \ldots, n-k. \tag{2.3}$$

Since $P \subset S$, the same must be true for every element of P.

Let $\mathbf{z}$ be an element of P. If $\mathbf{z}$ is an extreme point of P, then $\mathbf{z}$ is a trivial convex combination of the extreme points of P and there is nothing more to be proved. If $\mathbf{z}$ is not an extreme point of P, let us choose an arbitrary extreme point $\mathbf{y}$ of P and form the half-line consisting of all points of the form $\mathbf{z} + \lambda(\mathbf{z} - \mathbf{y})$, where λ is a nonnegative scalar. Since P is bounded, this half-line must eventually exit P and violate one of the constraints, say the constraint $\mathbf{a}_{i^*}'\mathbf{x} \geq b_{i^*}$. By considering what happens when this constraint is just about to be violated, we find some $\lambda^* \geq 0$ and $\mathbf{u} \in P$, such that

$$\mathbf{u} = \mathbf{z} + \lambda^*(\mathbf{z} - \mathbf{y}),$$

and

$$\mathbf{a}_{i^*}'\mathbf{u} = b_{i^*}.$$

Since the constraint $\mathbf{a}_{i^*}'\mathbf{x} \geq b_{i^*}$ is violated if λ grows beyond λ^*, it follows that $\mathbf{a}_{i^*}'(\mathbf{z} - \mathbf{y}) < 0$.

Let Q be the polyhedron defined by

$$\begin{aligned} Q &= \left\{\mathbf{x} \in P \mid \mathbf{a}_{i^*}'\mathbf{x} = b_{i^*}\right\} \\ &= \left\{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \geq b_i,\ i = 1, \ldots, m,\ \mathbf{a}_{i^*}'\mathbf{x} = b_{i^*}\right\}. \end{aligned}$$

Since $\mathbf{z}, \mathbf{y} \in P$, we have $\mathbf{f}_i'\mathbf{z} = g_i = \mathbf{f}_i'\mathbf{y}$ which shows that $\mathbf{z} - \mathbf{y}$ is orthogonal to each vector $\mathbf{f}_i$, for $i = 1, \ldots, n-k$. On the other hand, we have shown that $\mathbf{a}_{i^*}'(\mathbf{z} - \mathbf{y}) < 0$, which implies that the vector $\mathbf{a}_{i^*}$ is not a linear combination of, and is therefore linearly independent from, the vectors $\mathbf{f}_i$. Note that

$$Q \subset \left\{\mathbf{x} \in \Re^n \mid \mathbf{a}_{i^*}'\mathbf{x} = b_i,\ \mathbf{f}_i'\mathbf{x} = g_i,\ i = 1, \ldots, n-k\right\},$$

since Eq. (2.3) holds for every element of P. The set on the right is defined by $n-k+1$ linearly independent equality constraints. Hence, it is an affine subspace of dimension $k-1$ (see the discussion at the end of Section 1.5). Therefore, Q has dimension at most $k-1$.

By applying the induction hypothesis to Q and $\mathbf{u}$, we see that $\mathbf{u}$ can be expressed as a convex combination

$$\mathbf{u} = \sum_i \lambda_i \mathbf{v}^i$$

of the extreme points $\mathbf{v}^i$ of Q, where λ_i are nonnegative scalars that sum to one. Note that at an extreme point $\mathbf{v}$ of Q, we must have $\mathbf{a}_i'\mathbf{v} = b_i$ for n linearly independent vectors $\mathbf{a}_i$; therefore, $\mathbf{v}$ must also be an extreme point of P. Using the definition of λ^*, we also have

$$\mathbf{z} = \frac{\mathbf{u} + \lambda^*\mathbf{y}}{1 + \lambda^*}.$$

Therefore,

$$\mathbf{z} = \frac{\lambda^* \mathbf{y}}{1+\lambda^*} + \sum_i \frac{\lambda_i}{1+\lambda^*} \mathbf{v}^i,$$

which shows that $\mathbf{z}$ is a convex combination of the extreme points of P. □

Example 2.6 Consider the polyhedron

$$P = \left\{(x_1, x_2, x_3) \mid x_1 + x_2 + x_3 \leq 1,\ x_1, x_2, x_3 \geq 0\right\}.$$

It has four extreme points, namely, $\mathbf{x}^1 = (1,0,0)$, $\mathbf{x}^2 = (0,1,0)$, $\mathbf{x}^3 = (0,0,1)$, and $\mathbf{x}^4 = (0,0,0)$. The vector $\mathbf{x} = (1/3, 1/3, 1/4)$ belongs to P. It can be represented as

$$\mathbf{x} = \frac{1}{3}\mathbf{x}^1 + \frac{1}{3}\mathbf{x}^2 + \frac{1}{4}\mathbf{x}^3 + \frac{1}{12}\mathbf{x}^4.$$

There is a converse to Theorem 2.9 asserting that the convex hull of a finite number of points is a polyhedron. This result is proved in the next section and again in Section 4.9.

2.8 Projections of polyhedra: Fourier-Motzkin elimination*

In this section, we present perhaps the oldest method for solving linear programming problems. This method is not practical because it requires a very large number of steps, but it has some interesting theoretical corollaries.

The key to this method is the concept of a *projection*, defined as follows: if $\mathbf{x} = (x_1, \ldots, x_n)$ is a vector in $\Re^n$ and $k \leq n$, the projection mapping $\pi_k : \Re^n \mapsto \Re^k$ projects $\mathbf{x}$ onto its first k coordinates:

$$\pi_k(\mathbf{x}) = \pi_k(x_1, \ldots, x_n) = (x_1, \ldots, x_k).$$

We also define the projection $\Pi_k(S)$ of a set $S \subset \Re^n$ by letting

$$\Pi_k(S) = \left\{\pi_k(\mathbf{x}) \mid \mathbf{x} \in S\right\};$$

see Figure 2.17 for an illustration. Note that S is nonempty if and only if $\Pi_k(S)$ is nonempty. An equivalent definition is

$$\Pi_k(S) = \left\{(x_1, \ldots, x_k) \mid \text{there exist } x_{k+1}, \ldots, x_n \text{ s.t. } (x_1, \ldots, x_n) \in S\right\}.$$

Suppose now that we wish to decide whether a given polyhedron $P \subset \Re^n$ is nonempty. If we can somehow eliminate the variable x_n and construct the set $\Pi_{n-1}(P) \subset \Re^{n-1}$, we can instead consider the presumably easier problem of deciding whether $\Pi_{n-1}(P)$ is nonempty. If we keep eliminating variables one by one, we eventually arrive at the set $\Pi_1(P)$ that

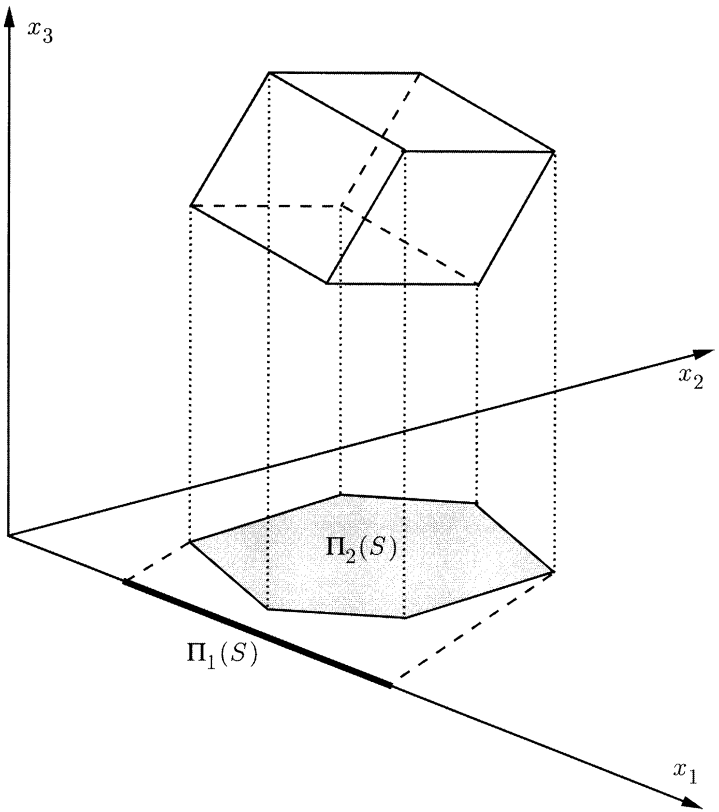


Figure 2.17: The projections $\Pi_1(S)$ and $\Pi_2(S)$ of a rotated three-dimensional cube.

involves a single variable, and whose emptiness is easy to check. The main disadvantage of this method is that while each step reduces the dimension by one, a large number of constraints is usually added. Exercise 2.20 deals with a family of examples in which the number of constraints increases exponentially with the problem dimension.

We now describe the elimination method. We are given a polyhedron P in terms of linear inequality constraints of the form

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

We wish to eliminate x_n and construct the projection $\Pi_{n-1}(P)$.

Elimination algorithm

1. Rewrite each constraint $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ in the form

$$a_{in}x_n \geq -\sum_{j=1}^{n-1} a_{ij}x_j + b_i, \qquad i = 1, \ldots, m;$$

if $a_{in} \neq 0$, divide both sides by a_{in}. By letting $\bar{\mathbf{x}} = (x_1, \ldots, x_{n-1})$, we obtain an equivalent representation of P involving the following constraints:

$$x_n \geq d_i + \mathbf{f}_i'\bar{\mathbf{x}}, \qquad \text{if } a_{in} > 0, \tag{2.4}$$

$$d_j + \mathbf{f}_j'\bar{\mathbf{x}} \geq x_n, \qquad \text{if } a_{jn} < 0, \tag{2.5}$$

$$0 \geq d_k + \mathbf{f}_k'\bar{\mathbf{x}}, \qquad \text{if } a_{kn} = 0. \tag{2.6}$$

Here, each d_i, d_j, d_k is a scalar, and each $\mathbf{f}_i$, $\mathbf{f}_j$, $\mathbf{f}_k$ is a vector in $\mathfrak{R}^{n-1}$.

2. Let Q be the polyhedron in $\mathfrak{R}^{n-1}$ defined by the constraints

$$d_j + \mathbf{f}_j'\bar{\mathbf{x}} \geq d_i + \mathbf{f}_i'\bar{\mathbf{x}}, \qquad \text{if } a_{in} > 0 \text{ and } a_{jn} < 0, \tag{2.7}$$

$$0 \geq d_k + \mathbf{f}_k'\bar{\mathbf{x}}, \qquad \text{if } a_{kn} = 0. \tag{2.8}$$

Example 2.7 Consider the polyhedron defined by the constraints

$$\begin{aligned} x_1 + x_2 &\geq 1 \\ x_1 + x_2 + 2x_3 &\geq 2 \\ 2x_1 + 3x_3 &\geq 3 \\ x_1 - 4x_3 &\geq 4 \\ -2x_1 + x_2 - x_3 &\geq 5. \end{aligned}$$

We rewrite these constraints in the form

$$\begin{aligned} 0 &\geq 1 - x_1 - x_2 \\ x_3 &\geq 1 - (x_1/2) - (x_2/2) \\ x_3 &\geq 1 - (2x_1/3) \\ -1 + (x_1/4) &\geq x_3 \\ -5 - 2x_1 + x_2 &\geq x_3. \end{aligned}$$

Then, the set Q is defined by the constraints

$$\begin{aligned} 0 &\geq 1 - x_1 - x_2 \\ -1 + x_1/4 &\geq 1 - (x_1/2) - (x_2/2) \end{aligned}$$

$$\begin{aligned} -1 + x_1/4 &\geq 1 - (2x_1/3) \\ -5 - 2x_1 + x_2 &\geq 1 - (x_1/2) - (x_2/2) \\ -5 - 2x_1 + x_2 &\geq 1 - (2x_1/3). \end{aligned}$$

Theorem 2.10 *The polyhedron Q constructed by the elimination algorithm is equal to the projection $\Pi_{n-1}(P)$ of P.*

Proof. If $\overline{\mathbf{x}} \in \Pi_{n-1}(P)$, there exists some x_n such that $(\overline{\mathbf{x}}, x_n) \in P$. In particular, the vector $\mathbf{x} = (\overline{\mathbf{x}}, x_n)$ satisfies Eqs. (2.4)-(2.6), from which it follows immediately that $\overline{\mathbf{x}}$ satisfies Eqs. (2.7)-(2.8), and $\overline{\mathbf{x}} \in Q$. This shows that $\Pi_{n-1}(P) \subset Q$.

We will now prove that $Q \subset \Pi_{n-1}(P)$. Let $\overline{\mathbf{x}} \in Q$. It follows from Eq. (2.7) that

$$\min_{\{j | a_{jn} < 0\}} \left(d_j + \mathbf{f}_j' \overline{\mathbf{x}}\right) \geq \max_{\{i | a_{in} > 0\}} \left(d_i + \mathbf{f}_i' \overline{\mathbf{x}}\right).$$

Let x_n be any number between the two sides of the above inequality. It then follows that $(\overline{\mathbf{x}}, x_n)$ satisfies Eqs. (2.4)-(2.6) and, therefore, belongs to the polyhedron P. □

Notice that for any vector $\mathbf{x} = (x_1, \ldots, x_n)$, we have

$$\pi_{n-2}\big(\pi_{n-1}(\mathbf{x})\big) = (x_1, \ldots, x_{n-2}) = \pi_{n-2}(\mathbf{x}).$$

Accordingly, for any polyhedron P, we also have

$$\Pi_{n-2}\big(\Pi_{n-1}(P)\big) = \Pi_{n-2}(P).$$

By generalizing this observation, we see that if we apply the elimination algorithm k times, we end up with the set $\Pi_{n-k}(P)$; if we apply it $n-1$ times, we end up with $\Pi_1(P)$. Unfortunately, each application of the elimination algorithm can increase the number of constraints substantially, leading to a polyhedron $\Pi_1(P)$ described by a very large number of constraints. Of course, since $\Pi_1(P)$ is one-dimensional, almost all of these constraints will be redundant, but this is of no help: in order to decide which ones are redundant, we must, in general, enumerate them.

The elimination algorithm has an important theoretical consequence: since the projection $\Pi_k(P)$ can be generated by repeated application of the elimination algorithm, and since the elimination algorithm always produces a polyhedron, it follows that a projection $\Pi_k(P)$ of a polyhedron is also a polyhedron. This fact might be considered obvious, but a proof simpler than the one we gave is not apparent. We now restate it in somewhat different language.

Corollary 2.4 *Let $P \subset \Re^{n+k}$ be a polyhedron. Then, the set*

$$\{\mathbf{x} \in \Re^n \mid \text{there exists } \mathbf{y} \in \Re^k \text{ such that } (\mathbf{x}, \mathbf{y}) \in P\}$$

is also a polyhedron.

A variation of Corollary 2.4 states that the image of a polyhedron under a linear mapping is also a polyhedron.

Corollary 2.5 *Let $P \subset \Re^n$ be a polyhedron and let $\mathbf{A}$ be an $m \times n$ matrix. Then, the set $Q = \{\mathbf{Ax} \mid \mathbf{x} \in P\}$ is also a polyhedron.*

Proof. We have $Q = \{\mathbf{y} \in \Re^m \mid \text{there exists } \mathbf{x} \in \Re^n \text{ such that } \mathbf{Ax} = \mathbf{y},\ \mathbf{x} \in P\}$. Therefore, Q is the projection of the polyhedron $\{(\mathbf{x}, \mathbf{y}) \in \Re^{n+m} \mid \mathbf{Ax} = \mathbf{y},\ \mathbf{x} \in P\}$ onto the $\mathbf{y}$ coordinates. □

Corollary 2.6 *The convex hull of a finite number of vectors is a polyhedron.*

Proof. The convex hull

$$\left\{ \sum_{i=1}^{k} \lambda_i \mathbf{x}^i \;\middle|\; \sum_{i=1}^{k} \lambda_i = 1,\ \lambda_i \geq 0 \right\}$$

of a finite number of vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$ is the image of the polyhedron

$$\left\{ (\lambda_1, \ldots, \lambda_k) \;\middle|\; \sum_{i=1}^{k} \lambda_i = 1,\ \lambda_i \geq 0 \right\}$$

under the linear mapping that maps $(\lambda_1, \ldots, \lambda_k)$ to $\sum_{i=1}^{k} \lambda_i \mathbf{x}^i$ and is, therefore, a polyhedron. □

We finally indicate how the elimination algorithm can be used to solve linear programming problems. Consider the problem of minimizing $\mathbf{c}'\mathbf{x}$ subject to $\mathbf{x}$ belonging to a polyhedron P. We define a new variable x_0 and introduce the constraint $x_0 = \mathbf{c}'\mathbf{x}$. If we use the elimination algorithm n times to eliminate the variables $x_1, \ldots, x_n$, we are left with the set

$$Q = \{x_0 \mid \text{there exists } \mathbf{x} \in P \text{ such that } x_0 = \mathbf{c}'\mathbf{x}\},$$

and the optimal cost is equal to the smallest element of Q. An optimal solution $\mathbf{x}$ can be recovered by backtracking (Exercise 2.21).

2.9 Summary

We summarize our main conclusions so far regarding the solutions to linear programming problems.

(a) If the feasible set is nonempty and bounded, there exists an optimal solution. Furthermore, there exists an optimal solution which is an extreme point.

(b) If the feasible set is unbounded, there are the following possibilities:

  (i) There exists an optimal solution which is an extreme point.

  (ii) There exists an optimal solution, but no optimal solution is an extreme point. (This can only happen if the feasible set has no extreme points; it never happens when the problem is in standard form.)

  (iii) The optimal cost is $-\infty$.

Suppose now that the optimal cost is finite and that the feasible set contains at least one extreme point. Since there are only finitely many extreme points, the problem can be solved in a finite number of steps, by enumerating all extreme points and evaluating the cost of each one. This is hardly a practical algorithm because the number of extreme points can increase exponentially with the number of variables and constraints. In the next chapter, we will exploit the geometry of the feasible set and develop the *simplex method*, a systematic procedure that moves from one extreme point to another, without having to enumerate all extreme points.

An interesting aspect of the material in this chapter is the distinction between geometric (representation independent) properties of a polyhedron and those properties that depend on a particular representation. In that respect, we have established the following:

(a) Whether or not a point is an extreme point (equivalently, vertex, or basic feasible solution) is a geometric property.

(b) Whether or not a point is a basic solution may depend on the way that a polyhedron is represented.

(c) Whether or not a basic or basic feasible solution is degenerate may depend on the way that a polyhedron is represented.

2.10 Exercises

Exercise 2.1 For each one of the following sets, determine whether it is a polyhedron.

(a) The set of all $(x, y) \in \Re^2$ satisfying the constraints

$$\begin{aligned} x\cos\theta + y\sin\theta &\leq 1, \qquad \forall\ \theta \in [0, \pi/2], \\ x &\geq 0, \\ y &\geq 0. \end{aligned}$$

(b) The set of all $x \in \Re$ satisfying the constraint $x^2 - 8x + 15 \leq 0$.

(c) The empty set.

Exercise 2.2 Let $f : \Re^n \mapsto \Re$ be a convex function and let c be some constant. Show that the set $S = \{\mathbf{x} \in \Re^n \mid f(\mathbf{x}) \leq c\}$ is convex.

Exercise 2.3 (Basic feasible solutions in standard form polyhedra with upper bounds) Consider a polyhedron defined by the constraints $\mathbf{Ax} = \mathbf{b}$ and $\mathbf{0} \leq \mathbf{x} \leq \mathbf{u}$, and assume that the matrix $\mathbf{A}$ has linearly independent rows. Provide a procedure analogous to the one in Section 2.3 for constructing basic solutions, and prove an analog of Theorem 2.4.

Exercise 2.4 We know that every linear programming problem can be converted to an equivalent problem in standard form. We also know that nonempty polyhedra in standard form have at least one extreme point. We are then tempted to conclude that every nonempty polyhedron has at least one extreme point. Explain what is wrong with this argument.

Exercise 2.5 (Extreme points of isomorphic polyhedra) A mapping f is called *affine* if it is of the form $f(\mathbf{x}) = \mathbf{Ax} + \mathbf{b}$, where $\mathbf{A}$ is a matrix and $\mathbf{b}$ is a vector. Let P and Q be polyhedra in $\Re^n$ and $\Re^m$, respectively. We say that P and Q are *isomorphic* if there exist affine mappings $f : P \mapsto Q$ and $g : Q \mapsto P$ such that $g\big(f(\mathbf{x})\big) = \mathbf{x}$ for all $\mathbf{x} \in P$, and $f\big(g(\mathbf{y})\big) = \mathbf{y}$ for all $\mathbf{y} \in Q$. (Intuitively, isomorphic polyhedra have the same shape.)

(a) If P and Q are isomorphic, show that there exists a one-to-one correspondence between their extreme points. In particular, if f and g are as above, show that $\mathbf{x}$ is an extreme point of P if and only if $f(\mathbf{x})$ is an extreme point of Q.

(b) **(Introducing slack variables leads to an isomorphic polyhedron)** Let $P = \{\mathbf{x} \in \Re^n \mid \mathbf{Ax} \geq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$, where $\mathbf{A}$ is a matrix of dimensions $k \times n$. Let $Q = \{(\mathbf{x}, \mathbf{z}) \in \Re^{n+k} \mid \mathbf{Ax} - \mathbf{z} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0},\ \mathbf{z} \geq \mathbf{0}\}$. Show that P and Q are isomorphic.

Exercise 2.6 (Carathéodory's theorem) Let $\mathbf{A}_1, \ldots, \mathbf{A}_n$ be a collection of vectors in $\Re^m$.

(a) Let

$$C = \left\{ \sum_{i=1}^{n} \lambda_i \mathbf{A}_i \;\middle|\; \lambda_1, \ldots, \lambda_n \geq 0 \right\}.$$

Show that any element of C can be expressed in the form $\sum_{i=1}^{n} \lambda_i \mathbf{A}_i$, with $\lambda_i \geq 0$, and with at most m of the coefficients λ_i being nonzero. *Hint:* Consider the polyhedron

$$\Lambda = \left\{ (\lambda_1, \ldots, \lambda_n) \in \Re^n \;\middle|\; \sum_{i=1}^{n} \lambda_i \mathbf{A}_i = \mathbf{y},\ \lambda_1, \ldots, \lambda_n \geq 0 \right\}.$$

(b) Let P be the convex hull of the vectors $\mathbf{A}_i$:

$$P = \left\{ \sum_{i=1}^{n} \lambda_i \mathbf{A}_i \;\middle|\; \sum_{i=1}^{n} \lambda_i = 1,\ \lambda_1, \ldots, \lambda_n \geq 0 \right\}.$$

Show that any element of P can be expressed in the form $\sum_{i=1}^{n} \lambda_i \mathbf{A}_i$, where $\sum_{i=1}^{n} \lambda_i = 1$ and $\lambda_i \geq 0$ for all i, with at most $m+1$ of the coefficients λ_i being nonzero.

Exercise 2.7 Suppose that $\{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \geq b_i,\ i = 1, \ldots, m\}$ and $\{\mathbf{x} \in \Re^n \mid \mathbf{g}_i'\mathbf{x} \geq h_i,\ i = 1, \ldots, k\}$ are two representations of the same nonempty polyhedron. Suppose that the vectors $\mathbf{a}_1, \ldots, \mathbf{a}_m$ span $\Re^n$. Show that the same must be true for the vectors $\mathbf{g}_1, \ldots, \mathbf{g}_k$.

Exercise 2.8 Consider the standard form polyhedron $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$, and assume that the rows of the matrix $\mathbf{A}$ are linearly independent. Let $\mathbf{x}$ be a basic solution, and let $J = \{i \mid x_i \neq 0\}$. Show that a basis is associated with the basic solution $\mathbf{x}$ if and only if every column $\mathbf{A}_i$, $i \in J$, is in the basis.

Exercise 2.9 Consider the standard form polyhedron $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$, and assume that the rows of the matrix $\mathbf{A}$ are linearly independent.

(a) Suppose that two different bases lead to the same basic solution. Show that the basic solution is degenerate.

(b) Consider a degenerate basic solution. Is it true that it corresponds to two or more distinct bases? Prove or give a counterexample.

(c) Suppose that a basic solution is degenerate. Is it true that there exists an adjacent basic solution which is degenerate? Prove or give a counterexample.

Exercise 2.10 Consider the standard form polyhedron $P = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$. Suppose that the matrix $\mathbf{A}$ has dimensions $m \times n$ and that its rows are linearly independent. For each one of the following statements, state whether it is true or false. If true, provide a proof, else, provide a counterexample.

(a) If $n = m + 1$, then P has at most two basic feasible solutions.

(b) The set of all optimal solutions is bounded.

(c) At every optimal solution, no more than m variables can be positive.

(d) If there is more than one optimal solution, then there are uncountably many optimal solutions.

(e) If there are several optimal solutions, then there exist at least two basic feasible solutions that are optimal.

(f) Consider the problem of minimizing $\max\{\mathbf{c}'\mathbf{x}, \mathbf{d}'\mathbf{x}\}$ over the set P. If this problem has an optimal solution, it must have an optimal solution which is an extreme point of P.

Exercise 2.11 Let $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$. Suppose that at a particular basic feasible solution, there are k active constraints, with $k > n$. Is it true that there exist exactly $\binom{k}{n}$ bases that lead to this basic feasible solution? Here $\binom{k}{n} = k!/\big(n!(k-n)!\big)$ is the number of ways that we can choose n out of k given items.

Exercise 2.12 Consider a nonempty polyhedron P and suppose that for each variable x_i we have either the constraint $x_i \geq 0$ or the constraint $x_i \leq 0$. Is it true that P has at least one basic feasible solution?

Exercise 2.13 Consider the standard form polyhedron $P = \{\mathbf{x} \mid \mathbf{Ax} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$. Suppose that the matrix $\mathbf{A}$, of dimensions $m \times n$, has linearly independent rows, and that all basic feasible solutions are nondegenerate. Let $\mathbf{x}$ be an element of P that has exactly m positive components.

(a) Show that $\mathbf{x}$ is a basic feasible solution.

(b) Show that the result of part (a) is false if the nondegeneracy assumption is removed.

Exercise 2.14 Let P be a bounded polyhedron in $\Re^n$, let $\mathbf{a}$ be a vector in $\Re^n$, and let b be some scalar. We define

$$Q = \left\{\mathbf{x} \in P \mid \mathbf{a}'\mathbf{x} = b\right\}.$$

Show that every extreme point of Q is either an extreme point of P or a convex combination of two adjacent extreme points of P.

Exercise 2.15 (Edges joining adjacent vertices) Consider the polyhedron $P = \{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \geq b_i,\ i = 1, \ldots, m\}$. Suppose that $\mathbf{u}$ and $\mathbf{v}$ are distinct basic feasible solutions that satisfy $\mathbf{a}_i'\mathbf{u} = \mathbf{a}_i'\mathbf{v} = b_i$, $i = 1, \ldots, n-1$, and that the vectors $\mathbf{a}_1, \ldots, \mathbf{a}_{n-1}$ are linearly independent. (In particular, $\mathbf{u}$ and $\mathbf{v}$ are adjacent.) Let $L = \{\lambda\mathbf{u} + (1-\lambda)\mathbf{v} \mid 0 \leq \lambda \leq 1\}$ be the segment that joins $\mathbf{u}$ and $\mathbf{v}$. Prove that $L = \{\mathbf{z} \in P \mid \mathbf{a}_i'\mathbf{z} = b_i,\ i = 1, \ldots, n-1\}$.

Exercise 2.16 Consider the set $\{\mathbf{x} \in \Re^n \mid x_1 = \cdots = x_{n-1} = 0,\ 0 \leq x_n \leq 1\}$. Could this be the feasible set of a problem in standard form?

Exercise 2.17 Consider the polyhedron $\{\mathbf{x} \in \Re^n \mid \mathbf{Ax} \leq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ and a nondegenerate basic feasible solution $\mathbf{x}^*$. We introduce slack variables $\mathbf{z}$ and construct a corresponding polyhedron $\{(\mathbf{x}, \mathbf{z}) \mid \mathbf{Ax} + \mathbf{z} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0},\ \mathbf{z} \geq \mathbf{0}\}$ in standard form. Show that $(\mathbf{x}^*, \mathbf{b} - \mathbf{Ax}^*)$ is a nondegenerate basic feasible solution for the new polyhedron.

Exercise 2.18 Consider a polyhedron $P = \{\mathbf{x} \mid \mathbf{Ax} \geq \mathbf{b}\}$. Given any $\epsilon > 0$, show that there exists some $\overline{\mathbf{b}}$ with the following two properties:

(a) The absolute value of every component of $\mathbf{b} - \overline{\mathbf{b}}$ is bounded by ϵ.

(b) Every basic feasible solution in the polyhedron $P = \{\mathbf{x} \mid \mathbf{Ax} \geq \overline{\mathbf{b}}\}$ is nondegenerate.

Exercise 2.19* Let $P \subset \Re^n$ be a polyhedron in standard form whose definition involves m linearly independent equality constraints. Its dimension is defined as the smallest integer k such that P is contained in some k-dimensional affine subspace of $\Re^n$.

(a) Explain why the dimension of P is at most $n - m$.

(b) Suppose that P has a nondegenerate basic feasible solution. Show that the dimension of P is equal to $n - m$.

(c) Suppose that $\mathbf{x}$ is a degenerate basic feasible solution. Show that $\mathbf{x}$ is degenerate under every standard form representation of the same polyhedron (in the same space $\Re^n$). *Hint:* Using parts (a) and (b), compare the number of equality constraints in two representations of P under which $\mathbf{x}$ is degenerate and nondegenerate, respectively. Then, count active constraints.

Exercise 2.20* Consider the Fourier-Motzkin elimination algorithm.

(a) Suppose that the number m of constraints defining a polyhedron P is even. Show, by means of an example, that the elimination algorithm may produce a description of the polyhedron $\Pi_{n-1}(P)$ involving as many as $m^2/4$ linear constraints, but no more than that.

(b) Show that the elimination algorithm produces a description of the one-dimensional polyhedron $\Pi_1(P)$ involving no more than $m^{2^{n-1}}/2^{2^n-2}$ constraints.

(c) Let $n = 2^p + p + 2$, where p is a nonnegative integer. Consider a polyhedron in $\Re^n$ defined by the $8\binom{n}{3}$ constraints

$$\pm x_i \pm x_j \pm x_k \leq 1, \qquad 1 \leq i < j < k \leq n,$$

where all possible combinations are present. Show that after p eliminations, we have at least

$$2^{2^p+2}$$

constraints. (Note that this number increases exponentially with n.)

Exercise 2.21 Suppose that Fourier-Motzkin elimination is used in the manner described at the end of Section 2.8 to find the optimal cost in a linear programming problem. Show how this approach can be augmented to obtain an optimal solution as well.

Exercise 2.22 Let P and Q be polyhedra in $\Re^n$. Let $P + Q = \{\mathbf{x} + \mathbf{y} \mid \mathbf{x} \in P,\ \mathbf{y} \in Q\}$.

(a) Show that $P + Q$ is a polyhedron.

(b) Show that every extreme point of $P + Q$ is the sum of an extreme point of P and an extreme point of Q.

2.11 Notes and sources

The relation between algebra and geometry goes far back in the history of mathematics, but was limited to two and three-dimensional spaces. The insight that the same relation goes through in higher dimensions only came in the middle of the nineteenth century.

2.2. Our algebraic definition of basic (feasible) solutions for general polyhedra, in terms of the number of linearly independent active constraints, is not common. Nevertheless, we consider it to be quite central, because it provides the main bridge between the algebraic and geometric viewpoint, it allows for a unified treatment, and shows that there is not much that is special about standard form problems.

2.8. Fourier-Motzkin elimination is due to Fourier (1827), Dines (1918), and Motzkin (1936).

Chapter 3

The simplex method

Contents

We saw in Chapter 2, that if a linear programming problem in standard form has an optimal solution, then there exists a basic feasible solution that is optimal. The simplex method is based on this fact and searches for an optimal solution by moving from one basic feasible solution to another, along the edges of the feasible set, always in a cost reducing direction. Eventually, a basic feasible solution is reached at which none of the available edges leads to a cost reduction; such a basic feasible solution is optimal and the algorithm terminates. In this chapter, we provide a detailed development of the simplex method and discuss a few different implementations, including the simplex tableau and the revised simplex method. We also address some difficulties that may arise in the presence of degeneracy. We provide an interpretation of the simplex method in terms of column geometry, and we conclude with a discussion of its running time, as a function of the dimension of the problem being solved.

Throughout this chapter, we consider the standard form problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

and we let P be the corresponding feasible set. We assume that the dimensions of the matrix $\mathbf{A}$ are $m \times n$ and that its rows are linearly independent. We continue using our previous notation: $\mathbf{A}_i$ is the ith column of the matrix $\mathbf{A}$, and $\mathbf{a}_i'$ is its ith row.

3.1 Optimality conditions

Many optimization algorithms are structured as follows: given a feasible solution, we search its neighborhood to find a nearby feasible solution with lower cost. If no nearby feasible solution leads to a cost improvement, the algorithm terminates and we have a *locally optimal* solution. For general optimization problems, a locally optimal solution need not be (globally) optimal. Fortunately, in linear programming, local optimality implies global optimality; this is because we are minimizing a convex function over a convex set (cf. Exercise 3.1). In this section, we concentrate on the problem of searching for a direction of cost decrease in a neighborhood of a given basic feasible solution, and on the associated optimality conditions.

Suppose that we are at a point $\mathbf{x} \in P$ and that we contemplate moving away from $\mathbf{x}$, in the direction of a vector $\mathbf{d} \in \Re^n$. Clearly, we should only consider those choices of $\mathbf{d}$ that do not immediately take us outside the feasible set. This leads to the following definition, illustrated in Figure 3.1.

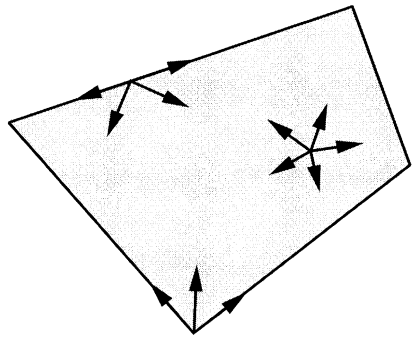


Figure 3.1: Feasible directions at different points of a polyhedron.

Definition 3.1 *Let* $\mathbf{x}$ *be an element of a polyhedron* P*. A vector* $\mathbf{d} \in \Re^n$ *is said to be a* **feasible direction** *at* $\mathbf{x}$*, if there exists a positive scalar* θ *for which* $\mathbf{x} + \theta\mathbf{d} \in P$*.*

Let $\mathbf{x}$ be a basic feasible solution to the standard form problem, let $B(1), \ldots, B(m)$ be the indices of the basic variables, and let $\mathbf{B} = [\mathbf{A}_{B(1)} \cdots \mathbf{A}_{B(m)}]$ be the corresponding basis matrix. In particular, we have $x_i = 0$ for every nonbasic variable, while the vector $\mathbf{x}_B = \big(x_{B(1)}, \ldots, x_{B(m)}\big)$ of basic variables is given by

$$\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}.$$

We consider the possibility of moving away from $\mathbf{x}$, to a new vector $\mathbf{x} + \theta\mathbf{d}$, by selecting a nonbasic variable x_j (which is initially at zero level), and increasing it to a positive value θ, while keeping the remaining nonbasic variables at zero. Algebraically, $d_j = 1$, and $d_i = 0$ for every nonbasic index i other than j. At the same time, the vector $\mathbf{x}_B$ of basic variables changes to $\mathbf{x}_B + \theta\mathbf{d}_B$, where $\mathbf{d}_B = \big(d_{B(1)}, d_{B(2)}, \ldots, d_{B(m)}\big)$ is the vector with those components of $\mathbf{d}$ that correspond to the basic variables.

Given that we are only interested in feasible solutions, we require $\mathbf{A}(\mathbf{x} + \theta\mathbf{d}) = \mathbf{b}$, and since $\mathbf{x}$ is feasible, we also have $\mathbf{A}\mathbf{x} = \mathbf{b}$. Thus, for the equality constraints to be satisfied for $\theta > 0$, we need $\mathbf{A}\mathbf{d} = \mathbf{0}$. Recall now that $d_j = 1$, and that $d_i = 0$ for all other nonbasic indices i. Then,

$$\mathbf{0} = \mathbf{A}\mathbf{d} = \sum_{i=1}^{n} \mathbf{A}_i d_i = \sum_{i=1}^{m} \mathbf{A}_{B(i)} d_{B(i)} + \mathbf{A}_j = \mathbf{B}\mathbf{d}_B + \mathbf{A}_j.$$

Since the basis matrix $\mathbf{B}$ is invertible, we obtain

$$\mathbf{d}_B = -\mathbf{B}^{-1}\mathbf{A}_j. \tag{3.1}$$

The direction vector $\mathbf{d}$ that we have just constructed will be referred to as the jth *basic direction*. We have so far guaranteed that the equality constraints are respected as we move away from $\mathbf{x}$ along the basic direction $\mathbf{d}$. How about the nonnegativity constraints? We recall that the variable x_j is increased, and all other nonbasic variables stay at zero level. Thus, we need only worry about the basic variables. We distinguish two cases:

(a) Suppose that $\mathbf{x}$ is a nondegenerate basic feasible solution. Then, $\mathbf{x}_B > \mathbf{0}$, from which it follows that $\mathbf{x}_B + \theta \mathbf{d}_B \geq \mathbf{0}$, and feasibility is maintained, when θ is sufficiently small. In particular, $\mathbf{d}$ is a feasible direction.

(b) Suppose now that $\mathbf{x}$ is degenerate. Then, $\mathbf{d}$ is not always a feasible direction. Indeed, it is possible that a basic variable $x_{B(i)}$ is zero, while the corresponding component $d_{B(i)}$ of $\mathbf{d}_B = -\mathbf{B}^{-1}\mathbf{A}_j$ is negative. In that case, if we follow the jth basic direction, the nonnegativity constraint for $x_{B(i)}$ is immediately violated, and we are led to infeasible solutions; see Figure 3.2.

We now study the effects on the cost function if we move along a basic direction. If $\mathbf{d}$ is the jth basic direction, then the rate $\mathbf{c}'\mathbf{d}$ of cost change along the direction $\mathbf{d}$ is given by $\mathbf{c}'_B\mathbf{d}_B + c_j$, where $\mathbf{c}_B = (c_{B(1)}, \ldots, c_{B(m)})$. Using Eq. (3.1), this is the same as $c_j - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j$. This quantity is important enough to warrant a definition. For an intuitive interpretation, c_j is the cost per unit increase in the variable x_j, and the term $-\mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j$ is the cost of the compensating change in the basic variables necessitated by the constraint $\mathbf{Ax} = \mathbf{b}$.

Definition 3.2 *Let $\mathbf{x}$ be a basic solution, let $\mathbf{B}$ be an associated basis matrix, and let $\mathbf{c}_B$ be the vector of costs of the basic variables. For each j, we define the* **reduced cost** *$\bar{c}_j$ of the variable x_j according to the formula*

$$\bar{c}_j = c_j - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j.$$

Example 3.1 Consider the linear programming problem

$$\begin{array}{lrcrcrcrcl} \text{minimize} & c_1x_1 & + & c_2x_2 & + & c_3x_3 & + & c_4x_4 & & \\ \text{subject to} & x_1 & + & x_2 & + & x_3 & + & x_4 & = & 2 \\ & 2x_1 & & & + & 3x_3 & + & 4x_4 & = & 2 \\ & x_1, x_2, x_3, x_4 & \geq & 0. & & & & & & \end{array}$$

The first two columns of the matrix $\mathbf{A}$ are $\mathbf{A}_1 = (1, 2)$ and $\mathbf{A}_2 = (1, 0)$. Since they are linearly independent, we can choose x_1 and x_2 as our basic variables. The corresponding basis matrix is

$$\mathbf{B} = \begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}.$$

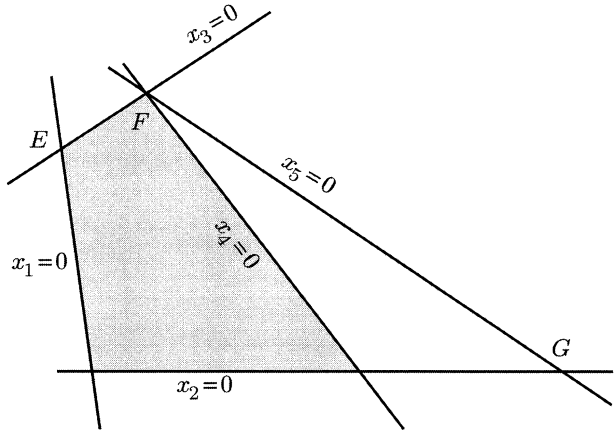


Figure 3.2: Let $n = 5$, $n-m = 2$. As discussed in Section 1.4, we can visualize the feasible set by standing on the two-dimensional set defined by the constraint $\mathbf{Ax} = \mathbf{b}$, in which case, the edges of the feasible set are associated with the nonnegativity constraints $x_i \geq 0$. At the nondegenerate basic feasible solution E, the variables x_1 and x_3 are at zero level (nonbasic) and x_2, x_4, x_5 are positive basic variables. The first basic direction is obtained by increasing x_1, while keeping the other nonbasic variable x_3 at zero level. This is the direction corresponding to the edge EF. Consider now the degenerate basic feasible solution F and let x_3, x_5 be the nonbasic variables. Note that x_4 is a basic variable at zero level. A basic direction is obtained by increasing x_3, while keeping the other nonbasic variable x_5 at zero level. This is the direction corresponding to the line FG and it takes us outside the feasible set. Thus, this basic direction is not a feasible direction.

We set $x_3 = x_4 = 0$, and solve for x_1, x_2, to obtain $x_1 = 1$ and $x_2 = 1$. We have thus obtained a nondegenerate basic feasible solution.

A basic direction corresponding to an increase in the nonbasic variable x_3, is constructed as follows. We have $d_3 = 1$ and $d_4 = 0$. The direction of change of the basic variables is obtained using Eq. (3.1):

$$\begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} d_{B(1)} \\ d_{B(2)} \end{bmatrix} = \mathbf{d}_B = -\mathbf{B}^{-1}\mathbf{A}_3 = -\begin{bmatrix} 0 & 1/2 \\ 1 & -1/2 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} -3/2 \\ 1/2 \end{bmatrix}.$$

The cost of moving along this basic direction is $\mathbf{c}'\mathbf{d} = -3c_1/2 + c_2/2 + c_3$. This is the same as the reduced cost of the variable x_3.

Consider now Definition 3.2 for the case of a basic variable. Since $\mathbf{B}$ is the matrix $[\mathbf{A}_{B(1)} \cdots \mathbf{A}_{B(m)}]$, we have $\mathbf{B}^{-1}[\mathbf{A}_{B(1)} \cdots \mathbf{A}_{B(m)}] = \mathbf{I}$, where

$\mathbf{I}$ is the $m \times m$ identity matrix. In particular, $\mathbf{B}^{-1}\mathbf{A}_{B(i)}$ is the ith column of the identity matrix, which is the ith unit vector $\mathbf{e}_i$. Therefore, for every basic variable $x_{B(i)}$, we have

$$\bar{c}_{B(i)} = c_{B(i)} - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A}_{B(i)} = c_{B(i)} - \mathbf{c}_B'\mathbf{e}_i = c_{B(i)} - c_{B(i)} = 0,$$

and we see that the reduced cost of every basic variable is zero.

Our next result provides us with optimality conditions. Given our interpretation of the reduced costs as rates of cost change along certain directions, this result is intuitive.

Theorem 3.1 *Consider a basic feasible solution* $\mathbf{x}$ *associated with a basis matrix* $\mathbf{B}$*, and let* $\bar{\mathbf{c}}$ *be the corresponding vector of reduced costs.*

(a) *If* $\bar{\mathbf{c}} \geq \mathbf{0}$*, then* $\mathbf{x}$ *is optimal.*

(b) *If* $\mathbf{x}$ *is optimal and nondegenerate, then* $\bar{\mathbf{c}} \geq \mathbf{0}$.

Proof.

(a) We assume that $\bar{\mathbf{c}} \geq \mathbf{0}$, we let $\mathbf{y}$ be an arbitrary feasible solution, and we define $\mathbf{d} = \mathbf{y} - \mathbf{x}$. Feasibility implies that $\mathbf{Ax} = \mathbf{Ay} = \mathbf{b}$ and, therefore, $\mathbf{Ad} = \mathbf{0}$. The latter equality can be rewritten in the form

$$\mathbf{B}\mathbf{d}_B + \sum_{i \in N} \mathbf{A}_i d_i = \mathbf{0},$$

where N is the set of indices corresponding to the nonbasic variables under the given basis. Since $\mathbf{B}$ is invertible, we obtain

$$\mathbf{d}_B = -\sum_{i \in N} \mathbf{B}^{-1}\mathbf{A}_i d_i,$$

and

$$\mathbf{c}'\mathbf{d} = \mathbf{c}_B'\mathbf{d}_B + \sum_{i \in N} c_i d_i = \sum_{i \in N} (c_i - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A}_i) d_i = \sum_{i \in N} \bar{c}_i d_i.$$

For any nonbasic index $i \in N$, we must have $x_i = 0$ and, since $\mathbf{y}$ is feasible, $y_i \geq 0$. Thus, $d_i \geq 0$ and $\bar{c}_i d_i \geq 0$, for all $i \in N$. We conclude that $\mathbf{c}'(\mathbf{y} - \mathbf{x}) = \mathbf{c}'\mathbf{d} \geq 0$, and since $\mathbf{y}$ was an arbitrary feasible solution, $\mathbf{x}$ is optimal.

(b) Suppose that $\mathbf{x}$ is a nondegenerate basic feasible solution and that $\bar{c}_j < 0$ for some j. Since the reduced cost of a basic variable is always zero, x_j must be a nonbasic variable and $\bar{c}_j$ is the rate of cost change along the jth basic direction. Since $\mathbf{x}$ is nondegenerate, the jth basic direction is a feasible direction of cost decrease, as discussed earlier. By moving in that direction, we obtain feasible solutions whose cost is less than that of $\mathbf{x}$, and $\mathbf{x}$ is not optimal. □

Note that Theorem 3.1 allows the possibility that $\mathbf{x}$ is a (degenerate) optimal basic feasible solution, but that $\bar{c}_j < 0$ for some nonbasic index j. There is an analog of Theorem 3.1 that provides conditions under which a basic feasible solution $\mathbf{x}$ is a unique optimal solution; see Exercise 3.6. A related view of the optimality conditions is developed in Exercises 3.2 and 3.3.

According to Theorem 3.1, in order to decide whether a nondegenerate basic feasible solution is optimal, we need only check whether all reduced costs are nonnegative, which is the same as examining the $n - m$ basic directions. If $\mathbf{x}$ is a degenerate basic feasible solution, an equally simple computational test for determining whether $\mathbf{x}$ is optimal is not available (see Exercises 3.7 and 3.8). Fortunately, the simplex method, as developed in subsequent sections, manages to get around this difficulty in an effective manner.

Note that in order to use Theorem 3.1 and assert that a certain basic solution is optimal, we need to satisfy two conditions: feasibility, and nonnegativity of the reduced costs. This leads us to the following definition.

Definition 3.3 *A basis matrix* $\mathbf{B}$ *is said to be* **optimal** *if:*

(a) $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$, *and*

(b) $\bar{\mathbf{c}}' = \mathbf{c}' - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{0}'$.

Clearly, if an optimal basis is found, the corresponding basic solution is feasible, satisfies the optimality conditions, and is therefore optimal. On the other hand, in the degenerate case, having an optimal basic feasible solution does not necessarily mean that the reduced costs are nonnegative.

3.2 Development of the simplex method

We will now complete the development of the simplex method. Our main task is to work out the details of how to move to a better basic feasible solution, whenever a profitable basic direction is discovered.

Let us assume that every basic feasible solution is nondegenerate. This assumption will remain in effect until it is explicitly relaxed later in this section. Suppose that we are at a basic feasible solution $\mathbf{x}$ and that we have computed the reduced costs $\bar{c}_j$ of the nonbasic variables. If all of them are nonnegative, Theorem 3.1 shows that we have an optimal solution, and we stop. If on the other hand, the reduced cost $\bar{c}_j$ of a nonbasic variable x_j is negative, the jth basic direction $\mathbf{d}$ is a feasible direction of cost decrease. [This is the direction obtained by letting $d_j = 1$, $d_i = 0$ for $i \neq B(1), \ldots, B(m), j$, and $\mathbf{d}_B = -\mathbf{B}^{-1}\mathbf{A}_j$.] While moving along this direction $\mathbf{d}$, the nonbasic variable x_j becomes positive and all other nonbasic

variables remain at zero. We describe this situation by saying that x_j (or $\mathbf{A}_j$) *enters* or *is brought into the basis.*

Once we start moving away from $\mathbf{x}$ along the direction $\mathbf{d}$, we are tracing points of the form $\mathbf{x} + \theta\mathbf{d}$, where $\theta \geq 0$. Since costs decrease along the direction $\mathbf{d}$, it is desirable to move as far as possible. This takes us to the point $\mathbf{x} + \theta^*\mathbf{d}$, where

$$\theta^* = \max\{\theta \geq 0 \mid \mathbf{x} + \theta\mathbf{d} \in P\}.$$

The resulting cost change is $\theta^*\mathbf{c}'\mathbf{d}$, which is the same as $\theta^*\bar{c}_j$.

We now derive a formula for θ^*. Given that $\mathbf{Ad} = \mathbf{0}$, we have $\mathbf{A}(\mathbf{x} + \theta\mathbf{d}) = \mathbf{Ax} = \mathbf{b}$ for all θ, and the equality constraints will never be violated. Thus, $\mathbf{x} + \theta\mathbf{d}$ can become infeasible only if one of its components becomes negative. We distinguish two cases:

(a) If $\mathbf{d} \geq \mathbf{0}$, then $\mathbf{x} + \theta\mathbf{d} \geq \mathbf{0}$ for all $\theta \geq 0$, the vector $\mathbf{x} + \theta\mathbf{d}$ never becomes infeasible, and we let $\theta^* = \infty$.

(b) If $d_i < 0$ for some i, the constraint $x_i + \theta d_i \geq 0$ becomes $\theta \leq -x_i/d_i$. This constraint on θ must be satisfied for every i with $d_i < 0$. Thus, the largest possible value of θ is

$$\theta^* = \min_{\{i \mid d_i < 0\}} \left(-\frac{x_i}{d_i}\right).$$

Recall that if x_i is a nonbasic variable, then either x_i is the entering variable and $d_i = 1$, or else $d_i = 0$. In either case, d_i is nonnegative. Thus, we only need to consider the basic variables and we have the equivalent formula

$$\theta^* = \min_{\{i=1,\ldots,m \mid d_{B(i)} < 0\}} \left(-\frac{x_{B(i)}}{d_{B(i)}}\right). \tag{3.2}$$

Note that $\theta^* > 0$, because $x_{B(i)} > 0$ for all i, as a consequence of nondegeneracy.

Example 3.2 This is a continuation of Example 3.1 from the previous section, dealing with the linear programming problem

$$\begin{array}{lrcrcrcrcl} \text{minimize} & c_1x_1 & + & c_2x_2 & + & c_3x_3 & + & c_4x_4 & & \\ \text{subject to} & x_1 & + & x_2 & + & x_3 & + & x_4 & = & 2 \\ & 2x_1 & & & + & 3x_3 & + & 4x_4 & = & 2 \\ & x_1, x_2, x_3, x_4 & \geq & 0. & & & & & & \end{array}$$

Let us again consider the basic feasible solution $\mathbf{x} = (1, 1, 0, 0)$ and recall that the reduced cost $\bar{c}_3$ of the nonbasic variable x_3 was found to be $-3c_1/2 + c_2/2 + c_3$. Suppose that $\mathbf{c} = (2, 0, 0, 0)$, in which case, we have $\bar{c}_3 = -3$. Since $\bar{c}_3$ is negative, we form the corresponding basic direction, which is $\mathbf{d} = (-3/2, 1/2, 1, 0)$, and consider vectors of the form $\mathbf{x} + \theta\mathbf{d}$, with $\theta \geq 0$. As θ increases, the only component of $\mathbf{x}$ that decreases is the first one (because $d_1 < 0$). The largest possible value

of θ is given by $\theta^* = -(x_1/d_1) = 2/3$. This takes us to the point $\mathbf{y} = \mathbf{x} + 2\mathbf{d}/3 = (0, 4/3, 2/3, 0)$. Note that the columns $\mathbf{A}_2$ and $\mathbf{A}_3$ corresponding to the nonzero variables at the new vector $\mathbf{y}$ are $(1, 0)$ and $(1, 3)$, respectively, and are linearly independent. Therefore, they form a basis and the vector $\mathbf{y}$ is a new basic feasible solution. In particular, the variable x_3 has entered the basis and the variable x_1 has exited the basis.

Once θ^* is chosen, and assuming it is finite, we move to the new feasible solution $\mathbf{y} = \mathbf{x} + \theta^*\mathbf{d}$. Since $x_j = 0$ and $d_j = 1$, we have $y_j = \theta^* > 0$. Let ℓ be a minimizing index in Eq. (3.2), that is,

$$-\frac{x_{B(\ell)}}{d_{B(\ell)}} = \min_{\{i=1,\ldots,m \mid d_{B(i)}<0\}} \left(-\frac{x_{B(i)}}{d_{B(i)}}\right) = \theta^*;$$

in particular,

$$d_{B(\ell)} < 0,$$

and

$$x_{B(\ell)} + \theta^* d_{B(\ell)} = 0.$$

We observe that the basic variable $x_{B(\ell)}$ has become zero, whereas the nonbasic variable x_j has now become positive, which suggests that x_j should replace $x_{B(\ell)}$ in the basis. Accordingly, we take the old basis matrix $\mathbf{B}$ and replace $\mathbf{A}_{B(\ell)}$ with $\mathbf{A}_j$, thus obtaining the matrix

$$\overline{\mathbf{B}} = \begin{bmatrix} | & & | & | & | & & | \\ \mathbf{A}_{B(1)} & \cdots & \mathbf{A}_{B(\ell-1)} & \mathbf{A}_j & \mathbf{A}_{B(\ell+1)} & \cdots & \mathbf{A}_{B(m)} \\ | & & | & | & | & & | \end{bmatrix}. \quad (3.3)$$

Equivalently, we are replacing the set $\{B(1), \ldots, B(m)\}$ of basic indices by a new set $\{\overline{B}(1), \ldots, \overline{B}(m)\}$ of indices given by

$$\overline{B}(i) = \begin{cases} B(i), & i \neq \ell, \\ j, & i = \ell. \end{cases} \quad (3.4)$$

Theorem 3.2

(a) *The columns $\mathbf{A}_{B(i)}$, $i \neq \ell$, and $\mathbf{A}_j$ are linearly independent and, therefore, $\overline{\mathbf{B}}$ is a basis matrix.*

(b) *The vector $\mathbf{y} = \mathbf{x} + \theta^*\mathbf{d}$ is a basic feasible solution associated with the basis matrix $\overline{\mathbf{B}}$.*

Proof.

(a) If the vectors $\mathbf{A}_{\overline{B}(i)}$, $i = 1, \ldots, m$, are linearly dependent, then there exist coefficients $\lambda_1, \ldots, \lambda_m$, not all of them zero, such that

$$\sum_{i=1}^{m} \lambda_i \mathbf{A}_{\overline{B}(i)} = \mathbf{0},$$

which implies that

$$\sum_{i=1}^{m} \lambda_i \mathbf{B}^{-1}\mathbf{A}_{\overline{B}(i)} = \mathbf{0},$$

and the vectors $\mathbf{B}^{-1}\mathbf{A}_{\overline{B}(i)}$ are also linearly dependent. To show that this is not the case, we will prove that the vectors $\mathbf{B}^{-1}\mathbf{A}_{B(i)}$, $i \neq \ell$, and $\mathbf{B}^{-1}\mathbf{A}_j$ are linearly independent. We have $\mathbf{B}^{-1}\mathbf{B} = \mathbf{I}$. Since $\mathbf{A}_{B(i)}$ is the ith column of $\mathbf{B}$, it follows that the vectors $\mathbf{B}^{-1}\mathbf{A}_{B(i)}$, $i \neq \ell$, are all the unit vectors except for the ℓth unit vector. In particular, they are linearly independent and their ℓth component is zero. On the other hand, $\mathbf{B}^{-1}\mathbf{A}_j$ is equal to $-\mathbf{d}_B$. Its ℓth entry, $-d_{B(\ell)}$, is nonzero by the definition of ℓ. Thus, $\mathbf{B}^{-1}\mathbf{A}_j$ is linearly independent from the unit vectors $\mathbf{B}^{-1}\mathbf{A}_{B(i)}$, $i \neq \ell$.

(b) We have $\mathbf{y} \geq \mathbf{0}$, $\mathbf{A}\mathbf{y} = \mathbf{b}$, and $y_i = 0$ for $i \neq \overline{B}(1), \ldots, \overline{B}(m)$. Furthermore, the columns $\mathbf{A}_{\overline{B}(1)}, \ldots, \mathbf{A}_{\overline{B}(m)}$ have just been shown to be linearly independent. It follows that $\mathbf{y}$ is a basic feasible solution associated with the basis matrix $\overline{\mathbf{B}}$. □

Since θ^* is positive, the new basic feasible solution $\mathbf{x} + \theta^*\mathbf{d}$ is distinct from $\mathbf{x}$; since $\mathbf{d}$ is a direction of cost decrease, the cost of this new basic feasible solution is strictly smaller. We have therefore accomplished our objective of moving to a new basic feasible solution with lower cost. We can now summarize a typical iteration of the simplex method, also known as a *pivot* (see Section 3.6 for a discussion of the origins of this term). For our purposes, it is convenient to define a vector $\mathbf{u} = (u_1, \ldots, u_m)$ by letting

$$\mathbf{u} = -\mathbf{d}_B = \mathbf{B}^{-1}\mathbf{A}_j,$$

where $\mathbf{A}_j$ is the column that enters the basis; in particular, $u_i = -d_{B(i)}$, for $i = 1, \ldots, m$.

An iteration of the simplex method

1. In a typical iteration, we start with a basis consisting of the basic columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$, and an associated basic feasible solution $\mathbf{x}$.
2. Compute the reduced costs $\overline{c}_j = c_j - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j$ for all nonbasic indices j. If they are all nonnegative, the current basic feasible solution is optimal, and the algorithm terminates; else, choose some j for which $\overline{c}_j < 0$.
3. Compute $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$. If no component of $\mathbf{u}$ is positive, we have $\theta^* = \infty$, the optimal cost is $-\infty$, and the algorithm terminates.

4. If some component of $\mathbf{u}$ is positive, let

$$\theta^* = \min_{\{i=1,\ldots,m | u_i > 0\}} \frac{x_{B(i)}}{u_i}.$$

5. Let ℓ be such that $\theta^* = x_{B(\ell)}/u_\ell$. Form a new basis by replacing $\mathbf{A}_{B(\ell)}$ with $\mathbf{A}_j$. If $\mathbf{y}$ is the new basic feasible solution, the values of the new basic variables are $y_j = \theta^*$ and $y_{B(i)} = x_{B(i)} - \theta^* u_i$, $i \neq \ell$.

The simplex method is initialized with an arbitrary basic feasible solution, which, for feasible standard form problems, is guaranteed to exist. The following theorem states that, in the nondegenerate case, the simplex method works correctly and terminates after a finite number of iterations.

Theorem 3.3 *Assume that the feasible set is nonempty and that every basic feasible solution is nondegenerate. Then, the simplex method terminates after a finite number of iterations. At termination, there are the following two possibilities:*

(a) *We have an optimal basis* $\mathbf{B}$ *and an associated basic feasible solution which is optimal.*

(b) *We have found a vector* $\mathbf{d}$ *satisfying* $\mathbf{Ad} = \mathbf{0}$, $\mathbf{d} \geq \mathbf{0}$, *and* $\mathbf{c}'\mathbf{d} < 0$, *and the optimal cost is* $-\infty$.

Proof. If the algorithm terminates due to the stopping criterion in Step 2, then the optimality conditions in Theorem 3.1 have been met, $\mathbf{B}$ is an optimal basis, and the current basic feasible solution is optimal.

If the algorithm terminates because the criterion in Step 3 has been met, then we are at a basic feasible solution $\mathbf{x}$ and we have discovered a nonbasic variable x_j such that $\bar{c}_j < 0$ and such that the corresponding basic direction $\mathbf{d}$ satisfies $\mathbf{Ad} = \mathbf{0}$ and $\mathbf{d} \geq \mathbf{0}$. In particular, $\mathbf{x} + \theta\mathbf{d} \in P$ for all $\theta > 0$. Since $\mathbf{c}'\mathbf{d} = \bar{c}_j < 0$, by taking θ arbitrarily large, the cost can be made arbitrarily negative, and the optimal cost is $-\infty$.

At each iteration, the algorithm moves by a positive amount θ^* along a direction $\mathbf{d}$ that satisfies $\mathbf{c}'\mathbf{d} < 0$. Therefore, the cost of every successive basic feasible solution visited by the algorithm is strictly less than the cost of the previous one, and no basic feasible solution can be visited twice. Since there is a finite number of basic feasible solutions, the algorithm must eventually terminate. □

Theorem 3.3 provides an independent proof of some of the results of Chapter 2 for nondegenerate standard form problems. In particular, it shows that for feasible and nondegenerate problems, either the optimal

cost is $-\infty$, or there exists a basic feasible solution which is optimal (cf. Theorem 2.8 in Section 2.6). While the proof given here might appear more elementary, its extension to the degenerate case is not as simple.

The simplex method for degenerate problems

We have been working so far under the assumption that all basic feasible solutions are nondegenerate. Suppose now that the exact same algorithm is used in the presence of degeneracy. Then, the following new possibilities may be encountered in the course of the algorithm.

(a) If the current basic feasible solution $\mathbf{x}$ is degenerate, θ^* can be equal to zero, in which case, the new basic feasible solution $\mathbf{y}$ is the same as $\mathbf{x}$. This happens if some basic variable $x_{B(\ell)}$ is equal to zero and the corresponding component $d_{B(\ell)}$ of the direction vector $\mathbf{d}$ is negative. Nevertheless, we can still define a new basis $\overline{\mathbf{B}}$, by replacing $\mathbf{A}_{B(\ell)}$ with $\mathbf{A}_j$ [cf. Eqs. (3.3)-(3.4)], and Theorem 3.2 is still valid.

(b) Even if θ^* is positive, it may happen that more than one of the original basic variables becomes zero at the new point $\mathbf{x}+\theta^*\mathbf{d}$. Since only one of them exits the basis, the others remain in the basis at zero level, and the new basic feasible solution is degenerate.

Basis changes while staying at the same basic feasible solution are not in vain. As illustrated in Figure 3.3, a sequence of such basis changes may lead to the eventual discovery of a cost reducing feasible direction. On the other hand, a sequence of basis changes might lead back to the initial basis, in which case the algorithm may loop indefinitely. This undesirable phenomenon is called *cycling*. An example of cycling is given in Section 3.3, after we develop some bookkeeping tools for carrying out the mechanics of the algorithm. It is sometimes maintained that cycling is an exceptionally rare phenomenon. However, for many highly structured linear programming problems, most basic feasible solutions are degenerate, and cycling is a real possibility. Cycling can be avoided by judiciously choosing the variables that will enter or exit the basis (see Section 3.4). We now discuss the freedom available in this respect.

Pivot Selection

The simplex algorithm, as we described it, has certain degrees of freedom: in Step 2, we are free to choose any j whose reduced cost $\bar{c}_j$ is negative; also, in Step 5, there may be several indices ℓ that attain the minimum in the definition of θ^*, and we are free to choose any one of them. Rules for making such choices are called *pivoting rules*.

Regarding the choice of the entering column, the following rules are some natural candidates:

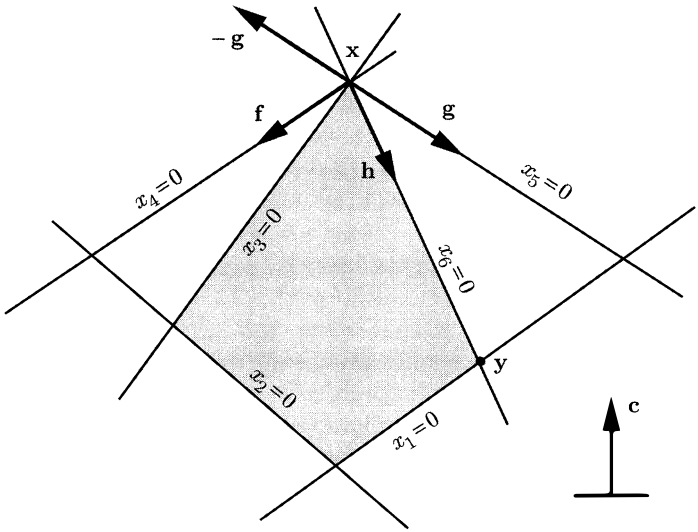


Figure 3.3: We visualize a problem in standard form, with $n - m = 2$, by standing on the two-dimensional plane defined by the equality constraints $\mathbf{Ax} = \mathbf{b}$. The basic feasible solution $\mathbf{x}$ is degenerate. If x_4 and x_5 are the nonbasic variables, then the two corresponding basic directions are the vectors $\mathbf{g}$ and $\mathbf{f}$. For either of these two basic directions, we have $\theta^* = 0$. However, if we perform a change of basis, with x_4 entering the basis and x_6 exiting, the new nonbasic variables are x_5 and x_6, and the two basic directions are $\mathbf{h}$ and $-\mathbf{g}$. (The direction $-\mathbf{g}$ is the one followed if x_6 is increased while x_5 is kept at zero.) In particular, we can now follow direction $\mathbf{h}$ to reach a new basic feasible solution $\mathbf{y}$ with lower cost.

(a) *Choose a column $\mathbf{A}_j$, with $\bar{c}_j < 0$, whose reduced cost is the most negative.* Since the reduced cost is the rate of change of the cost function, this rule chooses a direction along which costs decrease at the fastest rate. However, the actual cost decrease depends on how far we move along the chosen direction. This suggests the next rule.

(b) *Choose a column with $\bar{c}_j < 0$ for which the corresponding cost decrease $\theta^*|\bar{c}_j|$ is largest.* This rule offers the possibility of reaching optimality after a smaller number of iterations. On the other hand, the computational burden at each iteration is larger, because we need to compute θ^* for each column with $\bar{c}_j < 0$. The available empirical evidence suggests that the overall running time does not improve.

For large problems, even the rule that chooses the most negative $\bar{c}_j$ can be computationally expensive, because it requires the computation of the reduced cost of every variable. In practice, simpler rules are sometimes

used, such as the *smallest subscript* rule, that chooses the smallest j for which $\bar{c}_j$ is negative. Under this rule, once a negative reduced cost is discovered, there is no reason to compute the remaining reduced costs. Other criteria that have been found to improve the overall running time are the *Devex* (Harris, 1973) and the *steepest edge* rule (Goldfarb and Reid, 1977). Finally, there are methods based on *candidate lists* whereby one examines the reduced costs of nonbasic variables by picking them one at a time from a prioritized list. There are different ways of maintaining such prioritized lists, depending on the rule used for adding, removing, or reordering elements of the list.

Regarding the choice of the exiting column, the simplest option is again the *smallest subscript* rule: out of all variables eligible to exit the basis, choose one with the smallest subscript. It turns out that by following the smallest subscript rule for both the entering and the exiting column, cycling can be avoided (cf. Section 3.4).

3.3 Implementations of the simplex method

In this section, we discuss some ways of carrying out the mechanics of the simplex method. It should be clear from the statement of the algorithm that the vectors $\mathbf{B}^{-1}\mathbf{A}_j$ play a key role. If these vectors are available, the reduced costs, the direction of motion, and the stepsize θ^* are easily computed. Thus, the main difference between alternative implementations lies in the way that the vectors $\mathbf{B}^{-1}\mathbf{A}_j$ are computed and on the amount of related information that is carried from one iteration to the next.

When comparing different implementations, it is important to keep the following facts in mind (cf. Section 1.6). If $\mathbf{B}$ is a given $m \times m$ matrix and $\mathbf{b} \in \Re^m$ is a given vector, computing the inverse of $\mathbf{B}$ or solving a linear system of the form $\mathbf{Bx} = \mathbf{b}$ takes $O(m^3)$ arithmetic operations. Computing a matrix-vector product $\mathbf{Bb}$ takes $O(m^2)$ operations. Finally, computing an inner product $\mathbf{p}'\mathbf{b}$ of two m-dimensional vectors takes $O(m)$ arithmetic operations.

Naive implementation

We start by describing the most straightforward implementation in which no auxiliary information is carried from one iteration to the next. At the beginning of a typical iteration, we have the indices $B(1), \ldots, B(m)$ of the current basic variables. We form the basis matrix $\mathbf{B}$ and compute $\mathbf{p}' = \mathbf{c}'_B\mathbf{B}^{-1}$, by solving the linear system $\mathbf{p}'\mathbf{B} = \mathbf{c}'_B$ for the unknown vector $\mathbf{p}$. (This vector $\mathbf{p}$ is called the vector of *simplex multipliers* associated with the basis $\mathbf{B}$.) The reduced cost $\bar{c}_j = c_j - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}_j$ of any variable x_j is then obtained according to the formula

$$\bar{c}_j = c_j - \mathbf{p}'\mathbf{A}_j.$$

Depending on the pivoting rule employed, we may have to compute all of the reduced costs or we may compute them one at a time until a variable with a negative reduced cost is encountered. Once a column $\mathbf{A}_j$ is selected to enter the basis, we solve the linear system $\mathbf{Bu} = \mathbf{A}_j$ in order to determine the vector $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$. At this point, we can form the direction along which we will be moving away from the current basic feasible solution. We finally determine θ^* and the variable that will exit the basis, and construct the new basic feasible solution.

We note that we need $O(m^3)$ arithmetic operations to solve the systems $\mathbf{p}'\mathbf{B} = \mathbf{c}'_B$ and $\mathbf{Bu} = \mathbf{A}_j$. In addition, computing the reduced costs of all variables requires $O(mn)$ arithmetic operations, because we need to form the inner product of the vector $\mathbf{p}$ with each one of the nonbasic columns $\mathbf{A}_j$. Thus, the total computational effort per iteration is $O(m^3 + mn)$. We will see shortly that alternative implementations require only $O(m^2+mn)$ arithmetic operations. Therefore, the implementation described here is rather inefficient, in general. On the other hand, for certain problems with a special structure, the linear systems $\mathbf{p}'\mathbf{B} = \mathbf{c}'_B$ and $\mathbf{Bu} = \mathbf{A}_j$ can be solved very fast, in which case this implementation can be of practical interest. We will revisit this point in Chapter 7, when we apply the simplex method to network flow problems.

Revised simplex method

Much of the computational burden in the naive implementation is due to the need for solving two linear systems of equations. In an alternative implementation, the matrix $\mathbf{B}^{-1}$ is made available at the beginning of each iteration, and the vectors $\mathbf{c}'_B\mathbf{B}^{-1}$ and $\mathbf{B}^{-1}\mathbf{A}_j$ are computed by a matrix-vector multiplication. For this approach to be practical, we need an efficient method for updating the matrix $\mathbf{B}^{-1}$ each time that we effect a change of basis. This is discussed next.

Let

$$\mathbf{B} = \begin{bmatrix} \mathbf{A}_{B(1)} \cdots \mathbf{A}_{B(m)} \end{bmatrix}$$

be the basis matrix at the beginning of an iteration and let

$$\overline{\mathbf{B}} = \begin{bmatrix} \mathbf{A}_{B(1)} \cdots \mathbf{A}_{B(\ell-1)} & \mathbf{A}_j & \mathbf{A}_{B(\ell+1)} \cdots \mathbf{A}_{B(m)} \end{bmatrix}$$

be the basis matrix at the beginning of the next iteration. These two basis matrices have the same columns except that the ℓth column $\mathbf{A}_{B(\ell)}$ (the one that exits the basis) has been replaced by $\mathbf{A}_j$. It is then reasonable to expect that $\mathbf{B}^{-1}$ contains information that can be exploited in the computation of $\overline{\mathbf{B}}^{-1}$. After we develop some needed tools and terminology, we will see that this is indeed the case. An alternative explanation and line of development is outlined in Exercise 3.13.

Definition 3.4 *Given a matrix, not necessarily square, the operation of adding a constant multiple of one row to the same or to another row is called an* **elementary row operation**.

The example that follows indicates that performing an elementary row operation on a matrix $\mathbf{C}$ is equivalent to forming the matrix $\mathbf{QC}$, where $\mathbf{Q}$ is a suitably constructed square matrix.

Example 3.3 Let

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{C} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix},$$

and note that

$$\mathbf{QC} = \begin{bmatrix} 11 & 14 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}.$$

In particular, multiplication from the left by the matrix $\mathbf{Q}$ has the effect of multiplying the third row of $\mathbf{C}$ by two and adding it to the first row.

Generalizing Example 3.3, we see that multiplying the jth row by β and adding it to the ith row (for $i \neq j$) is the same as left-multiplying by the matrix $\mathbf{Q} = \mathbf{I} + \mathbf{D}_{ij}$, where $\mathbf{D}_{ij}$ is a matrix with all entries equal to zero, except for the (i,j)th entry which is equal to β. The determinant of such a matrix $\mathbf{Q}$ is equal to 1 and, therefore, $\mathbf{Q}$ is invertible.

Suppose now that we apply a *sequence* of K elementary row operations and that the kth such operation corresponds to left-multiplication by a certain invertible matrix $\mathbf{Q}_k$. Then, the sequence of these elementary row operations is the same as left-multiplication by the invertible matrix $\mathbf{Q}_K \mathbf{Q}_{K-1} \cdots \mathbf{Q}_2 \mathbf{Q}_1$. We conclude that performing a *sequence* of elementary row operations on a given matrix is equivalent to left-multiplying that matrix by a certain invertible matrix.

Since $\mathbf{B}^{-1}\mathbf{B} = \mathbf{I}$, we see that $\mathbf{B}^{-1}\mathbf{A}_{B(i)}$ is the ith unit vector $\mathbf{e}_i$. Using this observation, we have

$$\mathbf{B}^{-1}\overline{\mathbf{B}} = \begin{bmatrix} | & & | & | & | & & | \\ \mathbf{e}_1 & \cdots & \mathbf{e}_{\ell-1} & \mathbf{u} & \mathbf{e}_{\ell+1} & \cdots & \mathbf{e}_m \\ | & & | & | & | & & | \end{bmatrix}$$

$$= \begin{bmatrix} 1 & & u_1 & & \\ & \ddots & \vdots & & \\ & & u_\ell & & \\ & & \vdots & \ddots & \\ & & u_m & & 1 \end{bmatrix},$$

where $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$. Let us apply a sequence of elementary row operations that will change the above matrix to the identity matrix. In particular, consider the following sequence of elementary row operations.

(a) For each $i \neq \ell$, we add the ℓth row times $-u_i/u_\ell$ to the ith row. (Recall that $u_\ell > 0$.) This replaces u_i by zero.

(b) We divide the ℓth row by u_ℓ. This replaces u_ℓ by one.

In words, we are adding to each row a multiple of the ℓth row to replace the ℓth column $\mathbf{u}$ by the ℓth unit vector $\mathbf{e}_\ell$. This sequence of elementary row operations is equivalent to left-multiplying $\mathbf{B}^{-1}\overline{\mathbf{B}}$ by a certain invertible matrix $\mathbf{Q}$. Since the result is the identity, we have $\mathbf{Q}\mathbf{B}^{-1}\overline{\mathbf{B}} = \mathbf{I}$, which yields $\mathbf{Q}\mathbf{B}^{-1} = \overline{\mathbf{B}}^{-1}$. The last equation shows that if we apply the same sequence of row operations to the matrix $\mathbf{B}^{-1}$ (equivalently, left-multiply by $\mathbf{Q}$), we obtain $\overline{\mathbf{B}}^{-1}$. We conclude that all it takes to generate $\overline{\mathbf{B}}^{-1}$, is to start with $\mathbf{B}^{-1}$ and apply the sequence of elementary row operations described above.

Example 3.4 Let

$$\mathbf{B}^{-1} = \begin{bmatrix} 1 & 2 & 3 \\ -2 & 3 & 1 \\ 4 & -3 & -2 \end{bmatrix}, \qquad \mathbf{u} = \begin{bmatrix} -4 \\ 2 \\ 2 \end{bmatrix},$$

and suppose that $\ell = 3$. Thus, our objective is to transform the vector $\mathbf{u}$ to the unit vector $\mathbf{e}_3 = (0, 0, 1)$. We multiply the third row by 2 and add it to the first row. We subtract the third row from the second row. Finally, we divide the third row by 2. We obtain

$$\overline{\mathbf{B}}^{-1} = \begin{bmatrix} 9 & -4 & -1 \\ -6 & 6 & 3 \\ 2 & -1.5 & -1 \end{bmatrix}.$$

When the matrix $\mathbf{B}^{-1}$ is updated in the manner we have described, we obtain an implementation of the simplex method known as the *revised simplex method*, which we summarize below.

An iteration of the revised simplex method

1. In a typical iteration, we start with a basis consisting of the basic columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$, an associated basic feasible solution $\mathbf{x}$, and the inverse $\mathbf{B}^{-1}$ of the basis matrix.
2. Compute the row vector $\mathbf{p}' = \mathbf{c}'_B\mathbf{B}^{-1}$ and then compute the reduced costs $\overline{c}_j = c_j - \mathbf{p}'\mathbf{A}_j$. If they are all nonnegative, the current basic feasible solution is optimal, and the algorithm terminates; else, choose some j for which $\overline{c}_j < 0$.
3. Compute $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$. If no component of $\mathbf{u}$ is positive, the optimal cost is $-\infty$, and the algorithm terminates.

4. If some component of $\mathbf{u}$ is positive, let

$$\theta^* = \min_{\{i=1,\ldots,m | u_i > 0\}} \frac{x_{B(i)}}{u_i}.$$

5. Let ℓ be such that $\theta^* = x_{B(\ell)}/u_\ell$. Form a new basis by replacing $\mathbf{A}_{B(\ell)}$ with $\mathbf{A}_j$. If $\mathbf{y}$ is the new basic feasible solution, the values of the new basic variables are $y_j = \theta^*$ and $y_{B(i)} = x_{B(i)} - \theta^* u_i$, $i \neq \ell$.
6. Form the $m \times (m+1)$ matrix $[\mathbf{B}^{-1} \mid \mathbf{u}]$. Add to each one of its rows a multiple of the ℓth row to make the last column equal to the unit vector $\mathbf{e}_\ell$. The first m columns of the result is the matrix $\overline{\mathbf{B}}^{-1}$.

The full tableau implementation

We finally describe the implementation of simplex method in terms of the so-called *full tableau*. Here, instead of maintaining and updating the matrix $\mathbf{B}^{-1}$, we maintain and update the $m \times (n+1)$ matrix

$$\mathbf{B}^{-1}\Big[\mathbf{b} \mid \mathbf{A}\Big]$$

with columns $\mathbf{B}^{-1}\mathbf{b}$ and $\mathbf{B}^{-1}\mathbf{A}_1, \ldots, \mathbf{B}^{-1}\mathbf{A}_n$. This matrix is called the *simplex tableau*. Note that the column $\mathbf{B}^{-1}\mathbf{b}$, called the *zeroth column*, contains the values of the basic variables. The column $\mathbf{B}^{-1}\mathbf{A}_i$ is called the ith column of the tableau. The column $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$ corresponding to the variable that enters the basis is called the *pivot column*. If the ℓth basic variable exits the basis, the ℓth row of the tableau is called the *pivot row*. Finally, the element belonging to both the pivot row and the pivot column is called the *pivot element*. Note that the pivot element is u_ℓ and is always positive (unless $\mathbf{u} \leq \mathbf{0}$, in which case the algorithm has met the termination condition in Step 3).

The information contained in the rows of the tableau admits the following interpretation. The equality constraints are initially given to us in the form $\mathbf{b} = \mathbf{A}\mathbf{x}$. Given the current basis matrix $\mathbf{B}$, these equality constraints can also be expressed in the equivalent form

$$\mathbf{B}^{-1}\mathbf{b} = \mathbf{B}^{-1}\mathbf{A}\mathbf{x},$$

which is precisely the information in the tableau. In other words, the rows of the tableau provide us with the coefficients of the equality constraints $\mathbf{B}^{-1}\mathbf{b} = \mathbf{B}^{-1}\mathbf{A}\mathbf{x}$.

At the end of each iteration, we need to update the tableau $\mathbf{B}^{-1}[\mathbf{b} \mid \mathbf{A}]$ and compute $\overline{\mathbf{B}}^{-1}[\mathbf{b} \mid \mathbf{A}]$. This can be accomplished by left-multiplying the

simplex tableau with a matrix $\mathbf{Q}$ satisfying $\mathbf{Q}\mathbf{B}^{-1} = \overline{\mathbf{B}}^{-1}$. As explained earlier, this is the same as performing those elementary row operations that turn $\mathbf{B}^{-1}$ to $\overline{\mathbf{B}}^{-1}$; that is, we add to each row a multiple of the pivot row to set all entries of the pivot column to zero, with the exception of the pivot element which is set to one.

Regarding the determination of the exiting column $\mathbf{A}_{B(\ell)}$ and the stepsize θ^*, Steps 4 and 5 in the summary of the simplex method amount to the following: $x_{B(i)}/u_i$ is the ratio of the ith entry in the zeroth column of the tableau to the ith entry in the pivot column of the tableau. We only consider those i for which u_i is positive. The smallest ratio is equal to θ^* and determines ℓ.

It is customary to augment the simplex tableau by including a top row, to be referred to as the *zeroth row*. The entry at the top left corner contains the value $-\mathbf{c}'_B\mathbf{x}_B$, which is the negative of the current cost. (The reason for the minus sign is that it allows for a simple update rule, as will be seen shortly.) The rest of the zeroth row is the row vector of reduced costs, that is, the vector $\overline{\mathbf{c}}' = \mathbf{c}' - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}$. Thus, the structure of the tableau is:

$-\mathbf{c}'_B\mathbf{B}^{-1}\mathbf{b}$	$\mathbf{c}' - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A}$
$\mathbf{B}^{-1}\mathbf{b}$	$\mathbf{B}^{-1}\mathbf{A}$

or, in more detail,

$-\mathbf{c}'_B\mathbf{x}_B$	$\overline{c}_1$	$\cdots$	$\overline{c}_n$
$x_{B(1)}$	|		|
$\vdots$	$\mathbf{B}^{-1}\mathbf{A}_1$	$\cdots$	$\mathbf{B}^{-1}\mathbf{A}_n$
$x_{B(m)}$	|		|

The rule for updating the zeroth row turns out to be identical to the rule used for the other rows of the tableau: add a multiple of the pivot row to the zeroth row to set the reduced cost of the entering variable to zero. We will now verify that this update rule produces the correct results for the zeroth row.

At the beginning of a typical iteration, the zeroth row is of the form

$$[0 \mid \mathbf{c}'] - \mathbf{g}'[\mathbf{b} \mid \mathbf{A}],$$

where $\mathbf{g}' = \mathbf{c}'_B\mathbf{B}^{-1}$. Hence, the zeroth row is equal to $[0 \mid \mathbf{c}']$ plus a linear combination of the rows of $[\mathbf{b} \mid \mathbf{A}]$. Let column j be the pivot column, and row ℓ be the pivot row. Note that the pivot row is of the form $\mathbf{h}'[\mathbf{b} \mid \mathbf{A}]$, where the vector $\mathbf{h}'$ is the ℓth row of $\mathbf{B}^{-1}$. Hence, after a multiple of the

pivot row is added to the zeroth row, that row is again equal to $[0 \mid \mathbf{c}']$ plus a (different) linear combination of the rows of $[\mathbf{b} \mid \mathbf{A}]$, and is of the form

$$[0 \mid \mathbf{c}'] - \mathbf{p}'[\mathbf{b} \mid \mathbf{A}],$$

for some vector $\mathbf{p}$. Recall that our update rule is such that the pivot column entry of the zeroth row becomes zero, that is,

$$c_{\overline{B}(\ell)} - \mathbf{p}'\mathbf{A}_{\overline{B}(\ell)} = c_j - \mathbf{p}'\mathbf{A}_j = 0.$$

Consider now the $\overline{B}(i)$th column for $i \neq \ell$. (This is a column corresponding to a basic variable that stays in the basis.) The zeroth row entry of that column is zero, before the change of basis, since it is the reduced cost of a basic variable. Because $\mathbf{B}^{-1}\mathbf{A}_{B(i)}$ is the ith unit vector and $i \neq \ell$, the entry in the pivot row for that column is also equal to zero. Hence, adding a multiple of the pivot row to the zeroth row of the tableau does not affect the zeroth row entry of that column, which is left at zero. We conclude that the vector $\mathbf{p}$ satisfies $c_{\overline{B}(i)} - \mathbf{p}'\mathbf{A}_{\overline{B}(i)} = 0$ for every column $\mathbf{A}_{\overline{B}(i)}$ in the new basis. This implies that $\mathbf{c}'_{\overline{B}} - \mathbf{p}'\overline{\mathbf{B}} = \mathbf{0}$, and $\mathbf{p}' = \mathbf{c}'_{\overline{B}}\overline{\mathbf{B}}^{-1}$. Hence, with our update rule, the updated zeroth row of the tableau is equal to

$$[0 \mid \mathbf{c}'] - \mathbf{c}'_{\overline{B}}\overline{\mathbf{B}}^{-1}[\mathbf{b} \mid \mathbf{A}],$$

as desired.

We can now summarize the mechanics of the full tableau implementation.

An iteration of the full tableau implementation

1. A typical iteration starts with the tableau associated with a basis matrix $\mathbf{B}$ and the corresponding basic feasible solution $\mathbf{x}$.
2. Examine the reduced costs in the zeroth row of the tableau. If they are all nonnegative, the current basic feasible solution is optimal, and the algorithm terminates; else, choose some j for which $\overline{c}_j < 0$.
3. Consider the vector $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$, which is the jth column (the pivot column) of the tableau. If no component of $\mathbf{u}$ is positive, the optimal cost is $-\infty$, and the algorithm terminates.
4. For each i for which u_i is positive, compute the ratio $x_{B(i)}/u_i$. Let ℓ be the index of a row that corresponds to the smallest ratio. The column $\mathbf{A}_{B(\ell)}$ exits the basis and the column $\mathbf{A}_j$ enters the basis.
5. Add to each row of the tableau a constant multiple of the ℓth row (the pivot row) so that u_ℓ (the pivot element) becomes one and all other entries of the pivot column become zero.

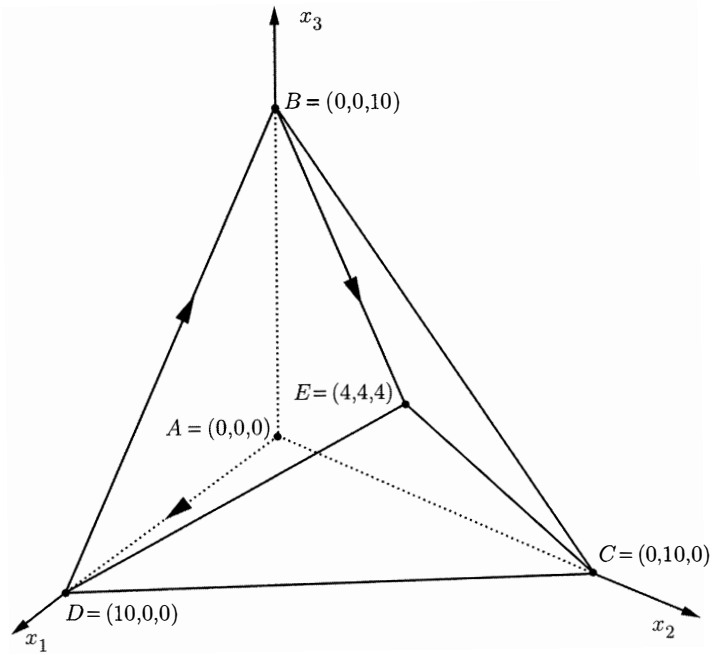


Figure 3.4: The feasible set in Example 3.5. Note that we have five extreme points. These are $A = (0, 0, 0)$ with cost 0, $B = (0, 0, 10)$ with cost -120, $C = (0, 10, 0)$ with cost -120, $D = (10, 0, 0)$ with cost -100, and $E = (4, 4, 4)$ with cost -136. In particular, E is the unique optimal solution.

Example 3.5 Consider the problem

$$\begin{array}{lrcrcrcl} \text{minimize} & -10x_1 & - & 12x_2 & - & 12x_3 & & \\ \text{subject to} & x_1 & + & 2x_2 & + & 2x_3 & \leq & 20 \\ & 2x_1 & + & x_2 & + & 2x_3 & \leq & 20 \\ & 2x_1 & + & 2x_2 & + & x_3 & \leq & 20 \\ & x_1, x_2, x_3 & \geq & 0. & & & & \end{array}$$

The feasible set is shown in Figure 3.4.

After introducing slack variables, we obtain the following standard form problem:

$$\begin{array}{lrcrcrcrcrcl} \text{minimize} & -10x_1 & - & 12x_2 & - & 12x_3 & & & & & \\ \text{subject to} & x_1 & + & 2x_2 & + & 2x_3 & + & x_4 & & & = 20 \\ & 2x_1 & + & x_2 & + & 2x_3 & & & + & x_5 & = 20 \\ & 2x_1 & + & 2x_2 & + & x_3 & & & & + x_6 & = 20 \\ & x_1, \ldots, x_6 & \geq & 0. & & & & & & & \end{array}$$

Note that $\mathbf{x} = (0, 0, 0, 20, 20, 20)$ is a basic feasible solution and can be used to start the algorithm. Let accordingly, $B(1) = 4$, $B(2) = 5$, and $B(3) = 6$. The

corresponding basis matrix is the identity matrix $\mathbf{I}$. To obtain the zeroth row of the initial tableau, we note that $\mathbf{c}_B = \mathbf{0}$ and, therefore, $\mathbf{c}_B'\mathbf{x}_B = 0$ and $\overline{\mathbf{c}} = \mathbf{c}$. Hence, we have the following initial tableau:

		x_1	x_2	x_3	x_4	x_5	x_6
	0	−10	−12	−12	0	0	0
$x_4 =$	20	1	2	2	1	0	0
$x_5 =$	20	2*	1	2	0	1	0
$x_6 =$	20	2	2	1	0	0	1

We note a few conventions in the format of the above tableau: the label x_i on top of the ith column indicates the variable associated with that column. The labels "$x_i =$" to the left of the tableau tell us which are the basic variables and in what order. For example, the first basic variable $x_{B(1)}$ is x_4, and its value is 20. Similarly, $x_{B(2)} = x_5 = 20$, and $x_{B(3)} = x_6 = 20$. Strictly speaking, these labels are not quite necessary. We know that the column in the tableau associated with the first basic variable must be the first unit vector. Once we observe that the column associated with the variable x_4 is the first unit vector, it follows that x_4 is the first basic variable.

We continue with our example. The reduced cost of x_1 is negative and we let that variable enter the basis. The pivot column is $\mathbf{u} = (1, 2, 2)$. We form the ratios $x_{B(i)}/u_i$, $i = 1, 2, 3$; the smallest ratio corresponds to $i = 2$ and $i = 3$. We break this tie by choosing $\ell = 2$. This determines the pivot element, which we indicate by an asterisk. The second basic variable $x_{B(2)}$, which is x_5, exits the basis. The new basis is given by $\overline{B}(1) = 4$, $\overline{B}(2) = 1$, and $\overline{B}(3) = 6$. We multiply the pivot row by 5 and add it to the zeroth row. We multiply the pivot row by 1/2 and subtract it from the first row. We subtract the pivot row from the third row. Finally, we divide the pivot row by 2. This leads us to the new tableau:

		x_1	x_2	x_3	x_4	x_5	x_6
	100	0	−7	−2	0	5	0
$x_4 =$	10	0	1.5	1*	1	−0.5	0
$x_1 =$	10	1	0.5	1	0	0.5	0
$x_6 =$	0	0	1	−1	0	−1	1

The corresponding basic feasible solution is $\mathbf{x} = (10, 0, 0, 10, 0, 0)$. In terms of the original variables x_1, x_2, x_3, we have moved to point $D = (10, 0, 0)$ in Figure 3.4. Note that this is a degenerate basic feasible solution, because the basic variable x_6 is equal to zero. This agrees with Figure 3.4 where we observe that there are four active constraints at point D.

We have mentioned earlier that the rows of the tableau (other than the zeroth row) amount to a representation of the equality constraints $\mathbf{B}^{-1}\mathbf{A}\mathbf{x} = \mathbf{B}^{-1}\mathbf{b}$, which are equivalent to the original constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$. In our current

example, the tableau indicates that the equality constraints can be written in the equivalent form:

$$\begin{aligned} 10 &= \quad 1.5x_2 + x_3 + x_4 - 0.5x_5 \\ 10 &= x_1 + 0.5x_2 + x_3 \quad + 0.5x_5 \\ 0 &= \quad x_2 - x_3 \quad - x_5 + x_6. \end{aligned}$$

We now return to the simplex method. With the current tableau, the variables x_2 and x_3 have negative reduced costs. Let us choose x_3 to be the one that enters the basis. The pivot column is $\mathbf{u} = (1, 1, -1)$. Since $u_3 < 0$, we only form the ratios $x_{B(i)}/u_i$, for $i = 1, 2$. There is again a tie, which we break by letting $\ell = 1$, and the first basic variable, x_4, exits the basis. The pivot element is again indicated by an asterisk. After carrying out the necessary elementary row operations, we obtain the following new tableau:

		x_1	x_2	x_3	x_4	x_5	x_6
	120	0	−4	0	2	4	0
$x_3 =$	10	0	1.5	1	1	−0.5	0
$x_1 =$	0	1	−1	0	−1	1	0
$x_6 =$	10	0	2.5*	0	1	−1.5	1

In terms of Figure 3.4, we have moved to point $B = (0, 0, 10)$, and the cost has been reduced to -120. At this point, x_2 is the only variable with negative reduced cost. We bring x_2 into the basis, x_6 exits, and the resulting tableau is:

		x_1	x_2	x_3	x_4	x_5	x_6
	136	0	0	0	3.6	1.6	1.6
$x_3 =$	4	0	0	1	0.4	0.4	−0.6
$x_1 =$	4	1	0	0	−0.6	0.4	0.4
$x_2 =$	4	0	1	0	0.4	−0.6	0.4

We have now reached point E in Figure 3.4. Its optimality is confirmed by observing that all reduced costs are nonnegative.

In this example, the simplex method took three changes of basis to reach the optimal solution, and it traced the path $A - D - B - E$ in Figure 3.4. With different pivoting rules, a different path would have been traced. Could the simplex method have solved the problem by tracing the path $A - D - E$, which involves only two edges, with only two iterations? The answer is no. The initial and final bases differ in three columns, and therefore at least three basis changes are required. In particular, if the method were to trace the path $A - D - E$, there would be a degenerate change of basis at point D (with no edge being traversed), which would again bring the total to three.

Example 3.6 This example shows that the simplex method can indeed cycle. We consider a problem described in terms of the following initial tableau.

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	−3/4	20	−1/2	6	0	0	0
$x_5 =$	0	1/4*	−8	−1	9	1	0	0
$x_6 =$	0	1/2	−12	−1/2	3	0	1	0
$x_7 =$	1	0	0	1	0	0	0	1

We use the following pivoting rules:

(a) We select a nonbasic variable with the most negative reduced cost $\bar{c}_j$ to be the one that enters the basis.

(b) Out of all basic variables that are eligible to exit the basis, we select the one with the smallest subscript.

We then obtain the following sequence of tableaux (the pivot element is indicated by an asterisk):

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	0	−4	−7/2	33	3	0	0
$x_1 =$	0	1	−32	−4	36	4	0	0
$x_6 =$	0	0	4*	3/2	−15	−2	1	0
$x_7 =$	1	0	0	1	0	0	0	1

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	0	0	−2	18	1	1	0
$x_1 =$	0	1	0	8*	−84	−12	8	0
$x_2 =$	0	0	1	3/8	−15/4	−1/2	1/4	0
$x_7 =$	1	0	0	1	0	0	0	1

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	1/4	0	0	−3	−2	3	0
$x_3 =$	0	1/8	0	1	−21/2	−3/2	1	0
$x_2 =$	0	−3/64	1	0	3/16*	1/16	−1/8	0
$x_7 =$	1	−1/8	0	0	21/2	3/2	−1	1

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	$-1/2$	16	0	0	-1	1	0
$x_3 =$	0	$-5/2$	56	1	0	2*	-6	0
$x_4 =$	0	$-1/4$	$16/3$	0	1	$1/3$	$-2/3$	0
$x_7 =$	1	$5/2$	-56	0	0	-2	6	1

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	$-7/4$	44	$1/2$	0	0	-2	0
$x_5 =$	0	$-5/4$	28	$1/2$	0	1	-3	0
$x_4 =$	0	$1/6$	-4	$-1/6$	1	0	$1/3$*	0
$x_7 =$	1	0	0	1	0	0	0	1

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	3	$-3/4$	20	$-1/2$	6	0	0	0
$x_5 =$	0	$1/4$	-8	-1	9	1	0	0
$x_6 =$	0	$1/2$	-12	$-1/2$	3	0	1	0
$x_7 =$	1	0	0	1	0	0	0	1

After six pivots, we have the same basis and the same tableau that we started with. At each basis change, we had $\theta^* = 0$. In particular, for each intermediate tableau, we had the same feasible solution and the same cost. The same sequence of pivots can be repeated over and over, and the simplex method never terminates.

Comparison of the full tableau and the revised simplex methods

Let us pretend that the problem is changed to

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + \mathbf{0}'\mathbf{y} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{I}\mathbf{y} = \mathbf{b} \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

We implement the simplex method on this new problem, except that we never allow any of the components of the vector $\mathbf{y}$ to become basic. Then, the simplex method performs basis changes as if the vector $\mathbf{y}$ were entirely

absent. Note also that the vector of reduced costs in the augmented problem is

$$[\mathbf{c}' \mid \mathbf{0}'] - \mathbf{c}'_B\mathbf{B}^{-1}[\mathbf{A} \mid \mathbf{I}] = [\bar{\mathbf{c}}' \mid -\mathbf{c}'_B\mathbf{B}^{-1}].$$

Thus, the simplex tableau for the augmented problem takes the form

$-\mathbf{c}'_B\mathbf{B}^{-1}\mathbf{b}$	$\bar{\mathbf{c}}'$	$-\mathbf{c}'_B\mathbf{B}^{-1}$
$\mathbf{B}^{-1}\mathbf{b}$	$\mathbf{B}^{-1}\mathbf{A}$	$\mathbf{B}^{-1}$

In particular, by following the mechanics of the full tableau method on the above tableau, the inverse basis matrix $\mathbf{B}^{-1}$ is made available at each iteration. We can now think of the revised simplex method as being essentially the same as the full tableau method applied to the above augmented problem, except that the part of the tableau containing $\mathbf{B}^{-1}\mathbf{A}$ is never formed explicitly; instead, once the entering variable x_j is chosen, the pivot column $\mathbf{B}^{-1}\mathbf{A}_j$ is computed on the fly. Thus, the revised simplex method is just a variant of the full tableau method, with more efficient bookkeeping. If the revised simplex method also updates the zeroth row entries that lie on top of $\mathbf{B}^{-1}$ (by the usual elementary operations), the simplex multipliers $\mathbf{p}' = \mathbf{c}'_B\mathbf{B}^{-1}$ become available, thus eliminating the need for solving the linear system $\mathbf{p}'\mathbf{B} = \mathbf{c}'_B$ at each iteration.

We now discuss the relative merits of the two methods. The full tableau method requires a constant (and small) number of arithmetic operations for updating each entry of the tableau. Thus, the amount of computation per iteration is proportional to the size of the tableau, which is $O(mn)$. The revised simplex method uses similar computations to update $\mathbf{B}^{-1}$ and $\mathbf{c}'_B\mathbf{B}^{-1}$, and since only $O(m^2)$ entries are updated, the computational requirements per iteration are $O(m^2)$. In addition, the reduced cost of each variable x_j can be computed by forming the inner product $\mathbf{p}'\mathbf{A}_j$, which requires $O(m)$ operations. In the worst case, the reduced cost of every variable is computed, for a total of $O(mn)$ computations per iteration. Since $m \leq n$, the worst-case computational effort per iteration is $O(mn+m^2) = O(mn)$, under either implementation. On the other hand, if we consider a pivoting rule that evaluates one reduced cost at a time, until a negative reduced cost is found, a typical iteration of the revised simplex method might require a lot less work. In the best case, if the first reduced cost computed is negative, and the corresponding variable is chosen to enter the basis, the total computational effort is only $O(m^2)$. The conclusion is that the revised simplex method cannot be slower than the full tableau method, and could be much faster during most iterations.

Another important element in favor of the revised simplex method is that memory requirements are reduced from $O(mn)$ to $O(m^2)$. As n is often much larger than m, this effect can be quite significant. It could be counterargued that the memory requirements of the revised simplex method

are also $O(mn)$ because of the need to store the matrix $\mathbf{A}$. However, in most large scale problems that arise in applications, the matrix $\mathbf{A}$ is very sparse (has many zero entries) and can be stored compactly. (Note that the sparsity of $\mathbf{A}$ does not usually help in the storage of the full simplex tableau because even if $\mathbf{A}$ and $\mathbf{B}$ are sparse, $\mathbf{B}^{-1}\mathbf{A}$ is not sparse, in general.)

We summarize this discussion in the following table:

	Full tableau	**Revised simplex**
Memory	$O(mn)$	$O(m^2)$
Worst-case time	$O(mn)$	$O(mn)$
Best-case time	$O(mn)$	$O(m^2)$

Table 3.1: Comparison of the full tableau method and revised simplex. The time requirements refer to a single iteration.

Practical performance enhancements

Practical implementations of the simplex method aimed at solving problems of moderate or large size incorporate a number of additional ideas from numerical linear algebra which we briefly mention.

The first idea is related to *reinversion*. Recall that at each iteration of the revised simplex method, the inverse basis matrix $\mathbf{B}^{-1}$ is updated according to certain rules. Each such iteration may introduce roundoff or truncation errors which accumulate and may eventually lead to highly inaccurate results. For this reason, it is customary to recompute the matrix $\mathbf{B}^{-1}$ from scratch once in a while. The efficiency of such reinversions can be greatly enhanced by using suitable data structures and certain techniques from computational linear algebra.

Another set of ideas is related to the way that the inverse basis matrix $\mathbf{B}^{-1}$ is represented. Suppose that a reinversion has been just carried out and $\mathbf{B}^{-1}$ is available. Subsequent to the current iteration of the revised simplex method, we have the option of generating explicitly and storing the new inverse basis matrix $\overline{\mathbf{B}}^{-1}$. An alternative that carries the same information, is to store a matrix $\mathbf{Q}$ such that $\mathbf{Q}\mathbf{B}^{-1} = \overline{\mathbf{B}}^{-1}$. Note that $\mathbf{Q}$ basically prescribes which elementary row operations need to be applied to $\mathbf{B}^{-1}$ in order to produce $\overline{\mathbf{B}}^{-1}$. It is not a full matrix, and can be completely specified in terms of m coefficients: for each row, we need to know what multiple of the pivot row must be added to it.

Suppose now that we wish to solve the system $\overline{\mathbf{B}}\mathbf{u} = \mathbf{A}_j$ for $\mathbf{u}$, where $\mathbf{A}_j$ is the entering column, as is required by the revised simplex method. We have $\mathbf{u} = \overline{\mathbf{B}}^{-1}\mathbf{A}_j = \mathbf{Q}\mathbf{B}^{-1}\mathbf{A}_j$, which shows that we can first compute

$\mathbf{B}^{-1}\mathbf{A}_j$ and then left-multiply by $\mathbf{Q}$ (equivalently, apply a sequence of elementary row operations) to produce $\mathbf{u}$. The same idea can also be used to represent the inverse basis matrix after several simplex iterations, as a product of the initial inverse basis matrix and several sparse matrices like $\mathbf{Q}$.

The last idea we mention is the following. Subsequent to a "reinversion," one does not usually compute $\mathbf{B}^{-1}$ explicitly, but $\mathbf{B}^{-1}$ is instead represented in terms of sparse triangular matrices with a special structure.

The methods discussed in this subsection are designed to accomplish two objectives: improve numerical stability (minimize the effect of roundoff errors) and exploit sparsity in the problem data to improve both running time and memory requirements. These methods have a critical effect in practice. Besides having a better chance of producing numerically trustworthy results, they can also speed up considerably the running time of the simplex method. These techniques lie much closer to the subject of numerical linear algebra, as opposed to optimization, and for this reason we do not pursue them in any greater depth.

3.4 Anticycling: lexicography and Bland's rule

In this section, we discuss anticycling rules under which the simplex method is guaranteed to terminate, thus extending Theorem 3.3 to degenerate problems. As an important corollary, we conclude that if the optimal cost is finite, then there exists an optimal basis, that is, a basis satisfying $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$ and $\overline{\mathbf{c}}' = \mathbf{c}' - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{0}'$.

Lexicography

We present here the lexicographic pivoting rule and prove that it prevents the simplex method from cycling. Historically, this pivoting rule was derived by analyzing the behavior of the simplex method on a nondegenerate problem obtained by means of a small perturbation of the right-hand side vector $\mathbf{b}$. This connection is pursued in Exercise 3.15.

We start with a definition.

Definition 3.5 *A vector* $\mathbf{u} \in \Re^n$ *is said to be* **lexicographically larger** *(or* **smaller***) than another vector* $\mathbf{v} \in \Re^n$ *if* $\mathbf{u} \neq \mathbf{v}$ *and the first nonzero component of* $\mathbf{u} - \mathbf{v}$ *is positive (or negative, respectively). Symbolically, we write*

$$\mathbf{u} \overset{L}{>} \mathbf{v} \qquad \text{or} \qquad \mathbf{u} \overset{L}{<} \mathbf{v}.$$

For example,

$$(0,\ 2,\ 3,\ 0) \overset{L}{>} (0,\ 2,\ 1,\ 4),$$

$$(0,\ 4,\ 5,\ 0) \overset{L}{<} (1,\ 2,\ 1,\ 2).$$

Lexicographic pivoting rule

1. Choose an entering column $\mathbf{A}_j$ arbitrarily, as long as its reduced cost $\bar{c}_j$ is negative. Let $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$ be the jth column of the tableau.
2. For each i with $u_i > 0$, divide the ith row of the tableau (including the entry in the zeroth column) by u_i and choose the lexicographically smallest row. If row ℓ is lexicographically smallest, then the ℓth basic variable $x_{B(\ell)}$ exits the basis.

Example 3.7 Consider the following tableau (the zeroth row is omitted), and suppose that the pivot column is the third one ($j = 3$).

1	0	5	3	$\cdots$
2	4	6	-1	$\cdots$
3	0	7	9	$\cdots$

Note that there is a tie in trying to determine the exiting variable because $x_{B(1)}/u_1 = 1/3$ and $x_{B(3)}/u_3 = 3/9 = 1/3$. We divide the first and third rows of the tableau by $u_1 = 3$ and $u_3 = 9$, respectively, to obtain:

1/3	0	5/3	1	$\cdots$
*	*	*	*	$\cdots$
1/3	0	7/9	1	$\cdots$

The tie between the first and third rows is resolved by performing a lexicographic comparison. Since $7/9 < 5/3$, the third row is chosen to be the pivot row, and the variable $x_{B(3)}$ exits the basis.

We note that the lexicographic pivoting rule always leads to a unique choice for the exiting variable. Indeed, if this were not the case, two of the rows in the tableau would have to be proportional. But if two rows of the matrix $\mathbf{B}^{-1}\mathbf{A}$ are proportional, the matrix $\mathbf{B}^{-1}\mathbf{A}$ has rank smaller than m and, therefore, $\mathbf{A}$ also has rank less than m, which contradicts our standing assumption that $\mathbf{A}$ has linearly independent rows.

Theorem 3.4 *Suppose that the simplex algorithm starts with all the rows in the simplex tableau, other than the zeroth row, lexicographically positive. Suppose that the lexicographic pivoting rule is followed. Then:*

(a) *Every row of the simplex tableau, other than the zeroth row, remains lexicographically positive throughout the algorithm.*

(b) *The zeroth row strictly increases lexicographically at each iteration.*

(c) *The simplex method terminates after a finite number of iterations.*

Proof.

(a) Suppose that all rows of the simplex tableau, other than the zeroth row, are lexicographically positive at the beginning of a simplex iteration. Suppose that x_j enters the basis and that the pivot row is the ℓth row. According to the lexicographic pivoting rule, we have $u_\ell > 0$ and

$$\frac{(\ell\text{th row})}{u_\ell} \overset{L}{<} \frac{(i\text{th row})}{u_i}, \qquad \text{if } i \neq \ell \text{ and } u_i > 0. \tag{3.5}$$

To determine the new tableau, the ℓth row is divided by the positive pivot element u_ℓ and, therefore, remains lexicographically positive. Consider the ith row and suppose that $u_i < 0$. In order to zero the (i, j)th entry of the tableau, we need to add a positive multiple of the pivot row to the ith row. Due to the lexicographic positivity of both rows, the ith row will remain lexicographically positive after this addition. Finally, consider the ith row for the case where $u_i > 0$ and $i \neq \ell$. We have

$$(\text{new } i\text{th row}) \;=\; (\text{old } i\text{th row}) \;-\; \frac{u_i}{u_\ell}(\text{old } \ell\text{th row}).$$

Because of the lexicographic inequality (3.5), which is satisfied by the old rows, the new ith row is also lexicographically positive.

(b) At the beginning of an iteration, the reduced cost in the pivot column is negative. In order to make it zero, we need to add a positive multiple of the pivot row. Since the latter row is lexicographically positive, the zeroth row increases lexicographically.

(c) Since the zeroth row increases lexicographically at each iteration, it never returns to a previous value. Since the zeroth row is determined completely by the current basis, no basis can be repeated twice and the simplex method must terminate after a finite number of iterations. □

The lexicographic pivoting rule is straightforward to use if the simplex method is implemented in terms of the full tableau. It can also be used

in conjunction with the revised simplex method, provided that the inverse basis matrix $\mathbf{B}^{-1}$ is formed explicitly (see Exercise 3.16). On the other hand, in sophisticated implementations of the revised simplex method, the matrix $\mathbf{B}^{-1}$ is never computed explicitly, and the lexicographic rule is not really suitable.

We finally note that in order to apply the lexicographic pivoting rule, an initial tableau with lexicographically positive rows is required. Let us assume that an initial tableau is available (methods for obtaining an initial tableau are discussed in the next section). We can then rename the variables so that the basic variables are the first m ones. This is equivalent to rearranging the tableau so that the first m columns of $\mathbf{B}^{-1}\mathbf{A}$ are the m unit vectors. The resulting tableau has lexicographically positive rows, as desired.

Bland's rule

The smallest subscript pivoting rule, also known as Bland's rule, is as follows.

Smallest subscript pivoting rule

1. Find the smallest j for which the reduced cost $\bar{c}_j$ is negative and have the column $\mathbf{A}_j$ enter the basis.
2. Out of all variables x_i that are tied in the test for choosing an exiting variable, select the one with the smallest value of i.

This pivoting rule is compatible with an implementation of the revised simplex method in which the reduced costs of the nonbasic variables are computed one at a time, in the natural order, until a negative one is discovered. Under this pivoting rule, it is known that cycling never occurs and the simplex method is guaranteed to terminate after a finite number of iterations.

3.5 Finding an initial basic feasible solution

In order to start the simplex method, we need to find an initial basic feasible solution. Sometimes this is straightforward. For example, suppose that we are dealing with a problem involving constraints of the form $\mathbf{Ax} \leq \mathbf{b}$, where $\mathbf{b} \geq \mathbf{0}$. We can then introduce nonnegative slack variables $\mathbf{s}$ and rewrite the constraints in the form $\mathbf{Ax} + \mathbf{s} = \mathbf{b}$. The vector $(\mathbf{x}, \mathbf{s})$ defined by $\mathbf{x} = \mathbf{0}$ and $\mathbf{s} = \mathbf{b}$ is a basic feasible solution and the corresponding basis matrix is the identity. In general, however, finding an initial basic feasible solution is not easy and requires the solution of an auxiliary linear programming problem, as will be seen shortly.

Consider the problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

By possibly multiplying some of the equality constraints by -1, we can assume, without loss of generality, that $\mathbf{b} \geq \mathbf{0}$. We now introduce a vector $\mathbf{y} \in \Re^m$ of *artificial variables* and use the simplex method to solve the auxiliary problem

$$\begin{aligned} \text{minimize} \quad & y_1 + y_2 + \cdots + y_m \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} + \mathbf{y} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \\ & \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

Initialization is easy for the auxiliary problem: by letting $\mathbf{x} = \mathbf{0}$ and $\mathbf{y} = \mathbf{b}$, we have a basic feasible solution and the corresponding basis matrix is the identity.

If $\mathbf{x}$ is a feasible solution to the original problem, this choice of $\mathbf{x}$ together with $\mathbf{y} = \mathbf{0}$, yields a zero cost solution to the auxiliary problem. Therefore, if the optimal cost in the auxiliary problem is nonzero, we conclude that the original problem is infeasible. If on the other hand, we obtain a zero cost solution to the auxiliary problem, it must satisfy $\mathbf{y} = \mathbf{0}$, and $\mathbf{x}$ is a feasible solution to the original problem.

At this point, we have accomplished our objectives only partially. We have a method that either detects infeasibility or finds a feasible solution to the original problem. However, in order to initialize the simplex method for the original problem, we need a basic feasible solution, an associated basis matrix $\mathbf{B}$, and – depending on the implementation – the corresponding tableau. All this is straightforward if the simplex method, applied to the auxiliary problem, terminates with a basis matrix $\mathbf{B}$ consisting exclusively of columns of $\mathbf{A}$. We can simply drop the columns that correspond to the artificial variables and continue with the simplex method on the original problem, using $\mathbf{B}$ as the starting basis matrix.

Driving artificial variables out of the basis

The situation is more complex if the original problem is feasible, the simplex method applied to the auxiliary problem terminates with a feasible solution $\mathbf{x}^*$ to the original problem, but some of the artificial variables are in the final basis. (Since the final value of the artificial variables is zero, this implies that we have a degenerate basic feasible solution to the auxiliary problem.) Let k be the number of columns of $\mathbf{A}$ that belong to the final basis ($k < m$) and, without loss of generality, assume that these are the columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(k)}$. (In particular, $x_{B(1)}, \ldots, x_{B(k)}$ are the only variables

that can be at nonzero level.) Note that the columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(k)}$ must be linearly independent since they are part of a basis. Under our standard assumption that the matrix $\mathbf{A}$ has full rank, the columns of $\mathbf{A}$ span $\Re^m$, and we can choose $m - k$ additional columns $\mathbf{A}_{B(k+1)}, \ldots, \mathbf{A}_{B(m)}$ of $\mathbf{A}$, to obtain a set of m linearly independent columns, that is, a basis consisting exclusively of columns of $\mathbf{A}$. With this basis, all nonbasic components of $\mathbf{x}^*$ are at zero level, and it follows that $\mathbf{x}^*$ is the basic feasible solution associated with this new basis as well. At this point, the artificial variables and the corresponding columns of the tableau can be dropped.

The procedure we have just described is called *driving the artificial variables out of the basis*, and depends crucially on the assumption that the matrix $\mathbf{A}$ has rank m. After all, if $\mathbf{A}$ has rank less than m, constructing a basis for $\Re^m$ using the columns of $\mathbf{A}$ is impossible and there exist redundant equality constraints that must be eliminated, as described by Theorem 2.5 in Section 2.3. All of the above can be carried out mechanically, in terms of the simplex tableau, in the following manner.

Suppose that the ℓth basic variable is an artificial variable, which is in the basis at zero level. We examine the ℓth row of the tableau and find some j such that the ℓth entry of $\mathbf{B}^{-1}\mathbf{A}_j$ is nonzero. We claim that $\mathbf{A}_j$ is linearly independent from the columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(k)}$. To see this, note that $\mathbf{B}^{-1}\mathbf{A}_{B(i)} = \mathbf{e}_i$, $i = 1, \ldots, k$, and since $k < \ell$, the ℓth entry of these vectors is zero. It follows that the ℓth entry of any linear combination of the vectors $\mathbf{B}^{-1}\mathbf{A}_{B(1)}, \ldots, \mathbf{B}^{-1}\mathbf{A}_{B(k)}$ is also equal to zero. Since the ℓth entry of $\mathbf{B}^{-1}\mathbf{A}_j$ is nonzero, this vector is not a linear combination of the vectors $\mathbf{B}^{-1}\mathbf{A}_{B(1)}, \ldots, \mathbf{B}^{-1}\mathbf{A}_{B(k)}$. Equivalently, $\mathbf{A}_j$ is not a linear combination of the vectors $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(k)}$, which proves our claim. We now bring $\mathbf{A}_j$ into the basis and have the ℓth basic variable exit the basis. This is accomplished in the usual manner: perform those elementary row operations that replace $\mathbf{B}^{-1}\mathbf{A}_j$ by the ℓth unit vector. The only difference from the usual mechanics of the simplex method is that the pivot element (the ℓth entry of $\mathbf{B}^{-1}\mathbf{A}_j$) could be negative. Because the ℓth basic variable was zero, adding a multiple of the ℓth row to the other rows does not change the values of the basic variables. This means that after the change of basis, we are still at the same basic feasible solution to the auxiliary problem, but we have reduced the number of basic artificial variables by one. We repeat this procedure as many times as needed until all artificial variables are driven out of the basis.

Let us now assume that the ℓth row of $\mathbf{B}^{-1}\mathbf{A}$ is zero, in which case the above described procedure fails. Note that the ℓth row of $\mathbf{B}^{-1}\mathbf{A}$ is equal to $\mathbf{g}'\mathbf{A}$, where $\mathbf{g}'$ is the ℓth row of $\mathbf{B}^{-1}$. Hence, $\mathbf{g}'\mathbf{A} = \mathbf{0}'$ for some nonzero vector $\mathbf{g}$, and the matrix $\mathbf{A}$ has linearly dependent rows. Since we are dealing with a feasible problem, we must also have $\mathbf{g}'\mathbf{b} = 0$. Thus, the constraint $\mathbf{g}'\mathbf{A}\mathbf{x} = \mathbf{g}'\mathbf{b}$ is redundant and can be eliminated (cf. Theorem 2.5 in Section 2.3). Since this constraint is the information provided by the ℓth row of the tableau, we can eliminate that row and continue from there.

Example 3.8 Consider the linear programming problem:

$$\begin{array}{lrcrcrcl} \text{minimize} & x_1 & + & x_2 & + & x_3 & & \\ \text{subject to} & x_1 & + & 2x_2 & + & 3x_3 & = & 3 \\ & -x_1 & + & 2x_2 & + & 6x_3 & = & 2 \\ & & & 4x_2 & + & 9x_3 & = & 5 \\ & & & & & 3x_3 + x_4 & = & 1 \\ & x_1, \ldots, x_4 \geq 0. & & & & & & \end{array}$$

In order to find a feasible solution, we form the auxiliary problem

$$\begin{array}{lrcrcrcrcrcl} \text{minimize} & & & & & & x_5 & + & x_6 & + & x_7 + x_8 & \\ \text{subject to} & x_1 & + & 2x_2 & + & 3x_3 & + & x_5 & & & & = 3 \\ & -x_1 & + & 2x_2 & + & 6x_3 & & & + & x_6 & & = 2 \\ & & & 4x_2 & + & 9x_3 & & & & + x_7 & & = 5 \\ & & & & & 3x_3 + x_4 & & & & & + x_8 & = 1 \\ & x_1, \ldots, x_8 \geq 0. & & & & & & & & & & \end{array}$$

A basic feasible solution to the auxiliary problem is obtained by letting $(x_5, x_6, x_7, x_8) = \mathbf{b} = (3, 2, 5, 1)$. The corresponding basis matrix is the identity. Furthermore, we have $\mathbf{c}_B = (1, 1, 1, 1)$. We evaluate the reduced cost of each one of the original variables x_i, which is $-\mathbf{c}_B' \mathbf{A}_i$, and form the initial tableau:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	-11	0	-8	-21	-1	0	0	0	0
$x_5 =$	3	1	2	3	0	1	0	0	0
$x_6 =$	2	-1	2	6	0	0	1	0	0
$x_7 =$	5	0	4	9	0	0	0	1	0
$x_8 =$	1	0	0	3	1*	0	0	0	1

We bring x_4 into the basis and have x_8 exit the basis. The basis matrix $\mathbf{B}$ is still the identity and only the zeroth row of the tableau changes. We obtain:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	-10	0	-8	-18	0	0	0	0	1
$x_5 =$	3	1	2	3	0	1	0	0	0
$x_6 =$	2	-1	2	6	0	0	1	0	0
$x_7 =$	5	0	4	9	0	0	0	1	0
$x_4 =$	1	0	0	3*	1	0	0	0	1

We now bring x_3 into the basis and have x_4 exit the basis. The new tableau is:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	-4	0	-8	0	6	0	0	0	7
$x_5 =$	2	1	2	0	-1	1	0	0	-1
$x_6 =$	0	-1	2^*	0	-2	0	1	0	-2
$x_7 =$	2	0	4	0	-3	0	0	1	-3
$x_3 =$	1/3	0	0	1	1/3	0	0	0	1/3

We now bring x_2 into the basis and x_6 exits. Note that this is a degenerate pivot with $\theta^* = 0$. The new tableau is:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	-4	-4	0	0	-2	0	4	0	-1
$x_5 =$	2	2^*	0	0	1	1	-1	0	1
$x_2 =$	0	$-1/2$	1	0	-1	0	1/2	0	-1
$x_7 =$	2	2	0	0	1	0	-2	1	1
$x_3 =$	1/3	0	0	1	1/3	0	0	0	1/3

We now have x_1 enter the basis and x_5 exit the basis. We obtain the following tableau:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
	0	0	0	0	0	2	2	0	1
$x_1 =$	1	1	0	0	1/2	1/2	$-1/2$	0	1/2
$x_2 =$	1/2	0	1	0	$-3/4$	1/4	1/4	0	$-3/4$
$x_7 =$	0	0	0	0	0	-1	-1	1	0
$x_3 =$	1/3	0	0	1	1/3	0	0	0	1/3

Note that the cost in the auxiliary problem has dropped to zero, indicating that we have a feasible solution to the original problem. However, the artificial variable x_7 is still in the basis, at zero level. In order to obtain a basic feasible solution to the original problem, we need to drive x_7 out of the basis. Note that x_7 is the third basic variable and that the third entry of the columns $\mathbf{B}^{-1}\mathbf{A}_j$, $j = 1, \ldots, 4$, associated with the original variables, is zero. This indicates that the matrix $\mathbf{A}$ has linearly dependent rows. At this point, we remove the third row of the tableau, because it corresponds to a redundant constraint, and also remove all of the artificial variables. This leaves us with the following initial tableau for the

original problem:

		x_1	x_2	x_3	x_4
	*	*	*	*	*
$x_1 =$	1	1	0	0	1/2
$x_2 =$	1/2	0	1	0	−3/4
$x_3 =$	1/3	0	0	1	1/3

We may now compute the reduced costs of the original variables, fill in the zeroth row of the tableau, and start executing the simplex method on the original problem.

We observe that in this example, the artificial variable x_8 was unnecessary. Instead of starting with $x_8 = 1$, we could have started with $x_4 = 1$ thus eliminating the need for the first pivot. More generally, whenever there is a variable that appears in a single constraint and with a positive coefficient (slack variables being the typical example), we can always let that variable be in the initial basis and we do not have to associate an artificial variable with that constraint.

The two-phase simplex method

We can now summarize a complete algorithm for linear programming problems in standard form.

Phase I:

1. By multiplying some of the constraints by -1, change the problem so that $\mathbf{b} \geq \mathbf{0}$.
2. Introduce artificial variables $y_1, \ldots, y_m$, if necessary, and apply the simplex method to the auxiliary problem with cost $\sum_{i=1}^{m} y_i$.
3. If the optimal cost in the auxiliary problem is positive, the original problem is infeasible and the algorithm terminates.
4. If the optimal cost in the auxiliary problem is zero, a feasible solution to the original problem has been found. If no artificial variable is in the final basis, the artificial variables and the corresponding columns are eliminated, and a feasible basis for the original problem is available.
5. If the ℓth basic variable is an artificial one, examine the ℓth entry of the columns $\mathbf{B}^{-1}\mathbf{A}_j$, $j = 1, \ldots, n$. If all of these entries are zero, the ℓth row represents a redundant constraint and is eliminated. Otherwise, if the ℓth entry of the jth column is nonzero, apply a change of basis (with this entry serving as the pivot

element): the ℓth basic variable exits and x_j enters the basis. Repeat this operation until all artificial variables are driven out of the basis.

Phase II:

1. Let the final basis and tableau obtained from Phase I be the initial basis and tableau for Phase II.
2. Compute the reduced costs of all variables for this initial basis, using the cost coefficients of the original problem.
3. Apply the simplex method to the original problem.

The above two-phase algorithm is a complete method, in the sense that it can handle all possible outcomes. As long as cycling is avoided (due to either nondegeneracy, an anticycling rule, or luck), one of the following possibilities will materialize:

(a) If the problem is infeasible, this is detected at the end of Phase I.

(b) If the problem is feasible but the rows of $\mathbf{A}$ are linearly dependent, this is detected and corrected at the end of Phase I, by eliminating redundant equality constraints.

(c) If the optimal cost is equal to $-\infty$, this is detected while running Phase II.

(d) Else, Phase II terminates with an optimal solution.

The big-M method

We close by mentioning an alternative approach, the *big-M method*, that combines the two phases into a single one. The idea is to introduce a cost function of the form

$$\sum_{j=1}^{n} c_j x_j + M \sum_{i=1}^{m} y_i,$$

where M is a large positive constant, and where y_i are the same artificial variables as in Phase I simplex. For a sufficiently large choice of M, if the original problem is feasible and its optimal cost is finite, all of the artificial variables are eventually driven to zero (Exercise 3.26), which takes us back to the minimization of the original cost function. In fact, there is no reason for fixing a numerical value for M. We can leave M as an undetermined parameter and let the reduced costs be functions of M. Whenever M is compared to another number (in order to determine whether a reduced cost is negative), M will be always treated as being larger.

Example 3.9 We consider the same linear programming problem as in Example 3.8:

$$\begin{array}{lrcrcrcrcl} \text{minimize} & x_1 & + & x_2 & + & x_3 & & & & \\ \text{subject to} & x_1 & + & 2x_2 & + & 3x_3 & & & = & 3 \\ & -x_1 & + & 2x_2 & + & 6x_3 & & & = & 2 \\ & & & 4x_2 & + & 9x_3 & & & = & 5 \\ & & & & & 3x_3 & + & x_4 & = & 1 \\ & x_1, \ldots, x_4 \geq 0. & & & & & & & & \end{array}$$

We use the big-M method in conjunction with the following auxiliary problem, in which the unnecessary artificial variable x_8 is omitted.

$$\begin{array}{lrcrcrcrcrcrcl} \text{minimize} & x_1 & + & x_2 & + & x_3 & + & Mx_5 & + & Mx_6 & + & Mx_7 & \\ \text{subject to} & x_1 & + & 2x_2 & + & 3x_3 & + & x_5 & & & & & = 3 \\ & -x_1 & + & 2x_2 & + & 6x_3 & & & + & x_6 & & & = 2 \\ & & & 4x_2 & + & 9x_3 & & & & & + & x_7 & = 5 \\ & & & & & 3x_3 & + & x_4 & & & & & = 1 \\ & x_1, \ldots, x_7 \geq 0. & & & & & & & & & & & \end{array}$$

A basic feasible solution to the auxiliary problem is obtained by letting $(x_5, x_6, x_7, x_4) = \mathbf{b} = (3, 2, 5, 1)$. The corresponding basis matrix is the identity. Furthermore, we have $\mathbf{c}_B = (M, M, M, 0)$. We evaluate the reduced cost of each one of the original variables x_i, which is $c_i - \mathbf{c}_B' \mathbf{A}_i$, and form the initial tableau:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	$-10M$	1	$-8M+1$	$-18M+1$	0	0	0	0
$x_5 =$	3	1	2	3	0	1	0	0
$x_6 =$	2	-1	2	6	0	0	1	0
$x_7 =$	5	0	4	9	0	0	0	1
$x_4 =$	1	0	0	3*	1	0	0	0

The reduced cost of x_3 is negative when M is large enough. We therefore bring x_3 into the basis and have x_4 exit. Note that in order to set the reduced cost of x_3 to zero, we need to multiply the pivot row by $6M - 1/3$ and add it to the zeroth row. The new tableau is:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	$-4M-1/3$	1	$-8M+1$	0	$6M-1/3$	0	0	0
$x_5 =$	2	1	2	0	-1	1	0	0
$x_6 =$	0	-1	2*	0	-2	0	1	0
$x_7 =$	2	0	4	0	-3	0	0	1
$x_3 =$	1/3	0	0	1	1/3	0	0	0

The reduced cost of x_2 is negative when M is large enough. We therefore bring x_2 into the basis and x_6 exits. Note that this is a degenerate pivot with $\theta^* = 0$.

The new tableau is:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	$-4M-\frac{1}{3}$	$-4M+\frac{3}{2}$	0	0	$-2M+\frac{2}{3}$	0	$4M-\frac{1}{2}$	0
$x_5 =$	2	2*	0	0	1	1	−1	0
$x_2 =$	0	−1/2	1	0	−1	0	1/2	0
$x_7 =$	2	2	0	0	1	0	−2	1
$x_3 =$	1/3	0	0	1	1/3	0	0	0

We now have x_1 enter and x_5 exit the basis. We obtain the following tableau:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	−11/6	0	0	0	−1/12	$2M-3/4$	$2M+1/4$	0
$x_1 =$	1	1	0	0	1/2	1/2	−1/2	0
$x_2 =$	1/2	0	1	0	−3/4	1/4	1/4	0
$x_7 =$	0	0	0	0	0	−1	−1	1
$x_3 =$	1/3	0	0	1	1/3*	0	0	0

We now bring x_4 into the basis and x_3 exits. The new tableau is:

		x_1	x_2	x_3	x_4	x_5	x_6	x_7
	−7/4	0	0	1/4	0	$2M-3/4$	$2M+1/4$	0
$x_1 =$	1/2	1	0	−3/2	0	1/2	−1/2	0
$x_2 =$	5/4	0	1	9/4	0	1/4	1/4	0
$x_7 =$	0	0	0	0	0	−1	−1	1
$x_4 =$	1	0	0	3	1	0	0	0

With M large enough, all of the reduced costs are nonnegative and we have an optimal solution to the auxiliary problem. In addition, all of the artificial variables have been driven to zero, and we have an optimal solution to the original problem.

3.6 Column geometry and the simplex method

In this section, we introduce an alternative way of visualizing the workings of the simplex method. This approach provides some insights into why the

simplex method appears to be efficient in practice.

We consider the problem

$$\begin{array}{rl} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{e}'\mathbf{x} = 1 \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \tag{3.6}$$

where $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{e}$ is the n-dimensional vector with all components equal to one. Although this might appear to be a special type of a linear programming problem, it turns out that every problem with a bounded feasible set can be brought into this form (Exercise 3.28). The constraint $\mathbf{e}'\mathbf{x} = 1$ is called the *convexity constraint*. We also introduce an auxiliary variable z defined by $z = \mathbf{c}'\mathbf{x}$. If $\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n$ are the n columns of $\mathbf{A}$, we are dealing with the problem of minimizing z subject to the nonnegativity constraints $\mathbf{x} \geq \mathbf{0}$, the convexity constraint $\sum_{i=1}^{n} x_i = 1$, and the constraint

$$x_1 \begin{bmatrix} \mathbf{A}_1 \\ c_1 \end{bmatrix} + x_2 \begin{bmatrix} \mathbf{A}_2 \\ c_2 \end{bmatrix} + \cdots + x_n \begin{bmatrix} \mathbf{A}_n \\ c_n \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ z \end{bmatrix}.$$

In order to capture this problem geometrically, we view the horizontal plane as an m-dimensional space containing the columns of $\mathbf{A}$, and we view the vertical axis as the one-dimensional space associated with the cost components c_i. Then, each point in the resulting three-dimensional space corresponds to a point $(\mathbf{A}_i, c_i)$; see Figure 3.5.

In this geometry, our objective is to construct a vector $(\mathbf{b}, z)$, which is a convex combination of the vectors $(\mathbf{A}_i, c_i)$, such that z is as small as possible. Note that the vectors of the form $(\mathbf{b}, z)$ lie on a vertical line, which we call the *requirement line*, and which intersects the horizontal plane at $\mathbf{b}$. If the requirement line does not intersect the convex hull of the points $(\mathbf{A}_i, c_i)$, the problem is infeasible. If it does intersect it, the problem is feasible and an optimal solution corresponds to the lowest point in the intersection of the convex hull and the requirement line. For example, in Figure 3.6, the requirement line intersects the convex hull of the points $(\mathbf{A}_i, c_i)$; the point G corresponds to an optimal solution, and its height is the optimal cost.

We now need some terminology.

Definition 3.6

(a) *A collection of vectors* $\mathbf{y}^1, \ldots, \mathbf{y}^{k+1}$ *in* $\Re^n$ *are said to be* **affinely independent** *if the vectors* $\mathbf{y}^1 - \mathbf{y}^{k+1}, \mathbf{y}^2 - \mathbf{y}^{k+1}, \ldots, \mathbf{y}^k - \mathbf{y}^{k+1}$ *are linearly independent. (Note that we must have* $k \leq n$.*)*

(b) *The convex hull of* $k+1$ *affinely independent vectors in* $\Re^n$ *is called a* k*-dimensional* **simplex**.

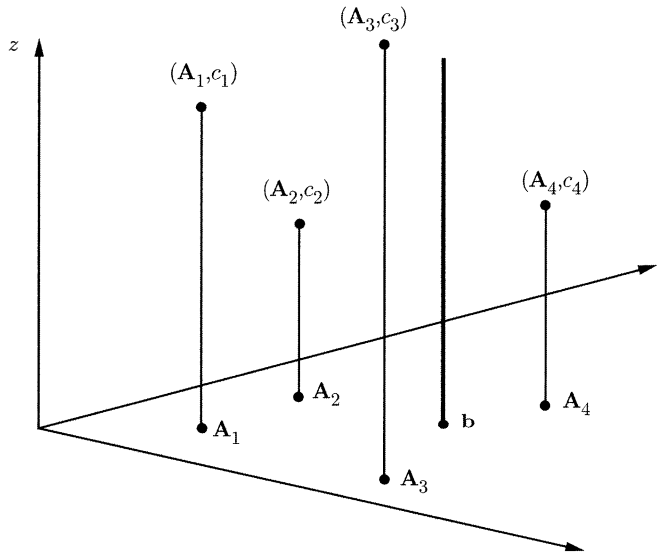


Figure 3.5: The column geometry.

Thus, three points are either collinear or they are affinely independent and determine a two-dimensional simplex (a triangle). Similarly, four points either lie on the same plane, or they are affinely independent and determine a three-dimensional simplex (a pyramid).

Let us now give an interpretation of basic feasible solutions to problem (3.6) in this geometry. Since we have added the convexity constraint, we have a total of $m+1$ equality constraints. Thus, a basic feasible solution is associated with a collection of $m+1$ linearly independent columns $(\mathbf{A}_i, 1)$ of the linear programming problem (3.6). These are in turn associated with $m+1$ of the points $(\mathbf{A}_i, c_i)$, which we call *basic points*; the remaining points $(\mathbf{A}_i, c_i)$ are called the *nonbasic points*. It is not hard to show that the $m+1$ basic points are affinely independent (Exercise 3.29) and, therefore, their convex hull is an m-dimensional simplex, which we call the *basic simplex*. Let the requirement line intersect the m-dimensional basic simplex at some point $(\mathbf{b}, z)$. The vector of weights x_i used in expressing $(\mathbf{b}, z)$ as a convex combination of the basic points, is the current basic feasible solution, and z represents its cost. For example, in Figure 3.6, the shaded triangle CDF is the basic simplex, and the point H corresponds to a basic feasible solution associated with the basic points C, D, and F.

Let us now interpret a change of basis geometrically. In a change of basis, a new point $(\mathbf{A}_j, c_j)$ becomes basic, and one of the currently basic points is to become nonbasic. For example, in Figure 3.6, if C, D, F, are the current basic points, we could make point B basic, replacing F

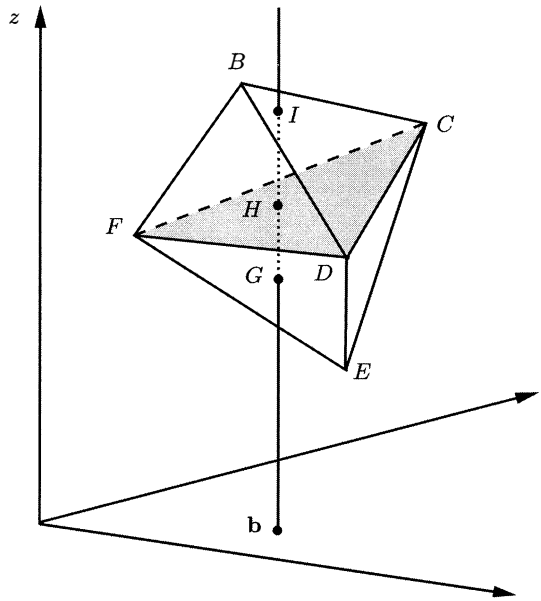


Figure 3.6: Feasibility and optimality in the column geometry.

(even though this turns out not to be profitable). The new basic simplex would be the convex hull of B, C, D, and the new basic feasible solution would correspond to point I. Alternatively, we could make point E basic, replacing C, and the new basic feasible solution would now correspond to point G. After a change of basis, the intercept of the requirement line with the new basic simplex is lower, and hence the cost decreases, if and only if the new basic point is below the plane that passes through the old basic points; we refer to the latter plane as the *dual plane*. For example, point E is below the dual plane and having it enter the basis is profitable; this is not the case for point B. In fact, the vertical distance from the dual plane to a point $(\mathbf{A}_j, c_j)$ is equal to the reduced cost of the associated variable x_j (Exercise 3.30); requiring the new basic point to be below the dual plane is therefore equivalent to requiring the entering column to have negative reduced cost.

We discuss next the selection of the basic point that will exit the basis. Each possible choice of the exiting point leads to a different basic simplex. These m basic simplices, together with the original basic simplex (before the change of basis) form the boundary (the faces) of an $(m+1)$-dimensional simplex. The requirement line exits this $(m+1)$-dimensional simplex through its top face and must therefore enter it by crossing some other face. This determines which one of the potential basic simplices will be obtained after the change of basis. In reference to Figure 3.6, the basic

points C, D, F, determine a two-dimensional basic simplex. If point E is to become basic, we obtain a three-dimensional simplex (pyramid) with vertices C, D, E, F. The requirement line exits the pyramid through its top face with vertices C, D, F. It enters the pyramid through the face with vertices D, E, F; this is the new basic simplex.

We can now visualize pivoting through the following physical analogy. Think of the original basic simplex with vertices C, D, F, as a solid object anchored at its vertices. Grasp the corner of the basic simplex at the vertex C leaving the basis, and pull the corner down to the new basic point E. While so moving, the simplex will hinge, or *pivot*, on its anchor and stretch down to the lower position. The somewhat peculiar terms (e.g., "simplex", "pivot") associated with the simplex method have their roots in this column geometry.

Example 3.10 Consider the problem illustrated in Figure 3.7, in which $m = 1$, and the following pivoting rule: choose a point $(\mathbf{A}_i, c_i)$ below the dual plane to become basic, whose vertical distance from the dual plane is largest. According to Exercise 3.30, this is identical to the pivoting rule that selects an entering variable with the most negative reduced cost. Starting from the initial basic simplex consisting of the points $(\mathbf{A}_3, c_3)$, $(\mathbf{A}_6, c_6)$, the next basic simplex is determined by the points $(\mathbf{A}_3, c_3)$, $(\mathbf{A}_5, c_5)$, and the next one by the points $(\mathbf{A}_5, c_5)$, $(\mathbf{A}_8, c_8)$. In particular, the simplex method only takes two pivots in this case. This example indicates why the simplex method may require a rather small number of pivots, even when the number of underlying variables is large.

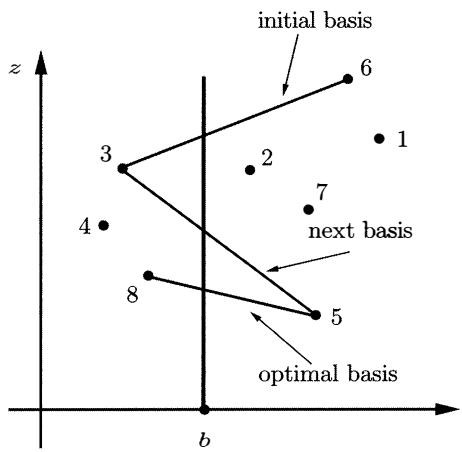


Figure 3.7: The simplex method finds the optimal basis after two iterations. Here, the point indicated by a number i corresponds to the vector $(\mathbf{A}_i, c_i)$.

3.7 Computational efficiency of the simplex method

The computational efficiency of the simplex method is determined by two factors:

(a) the computational effort at each iteration;

(b) the number of iterations.

The computational requirements of each iteration have already been discussed in Section 3.3. For example, the full tableau implementation needs $O(mn)$ arithmetic operations per iteration; the same is true for the revised simplex method in the worst case. We now turn to a discussion of the number of iterations.

The number of iterations in the worst case

Although the number of extreme points of the feasible set can increase exponentially with the number of variables and constraints, it has been observed in practice that the simplex method typically takes only $O(m)$ pivots to find an optimal solution. Unfortunately, however, this practical observation is not true for every linear programming problem. We will describe shortly a family of problems for which an exponential number of pivots may be required.

Recall that for nondegenerate problems, the simplex method always moves from one vertex to an adjacent one, each time improving the value of the cost function. We will now describe a polyhedron that has an exponential number of vertices, along with a path that visits all vertices, by taking steps from one vertex to an adjacent one that has lower cost. Once such a polyhedron is available, then the simplex method – under a pivoting rule that traces this path – needs an exponential number of pivots.

Consider the unit cube in $\Re^n$, defined by the constraints

$$0 \leq x_i \leq 1, \qquad i = 1, \ldots, n.$$

The unit cube has 2^n vertices (for each i, we may let either one of the two constraints $0 \leq x_i$ or $x_i \leq 1$ become active). Furthermore, there exists a path that travels along the edges of the cube and which visits each vertex exactly once; we call such a path a *spanning path*. It can be constructed according to the procedure illustrated in Figure 3.8.

Let us now introduce the cost function $-x_n$. Half of the vertices of the cube have zero cost and the other half have a cost of -1. Thus, the cost cannot decrease strictly with each move along the spanning path, and we do not yet have the desired example. However, if we choose some $\epsilon \in (0, 1/2)$ and consider the perturbation of the unit cube defined by the constraints

$$\epsilon \leq x_1 \leq 1, \tag{3.7}$$

$$\epsilon x_{i-1} \leq x_i \leq 1 - \epsilon x_{i-1}, \qquad i = 2, \ldots, n, \tag{3.8}$$

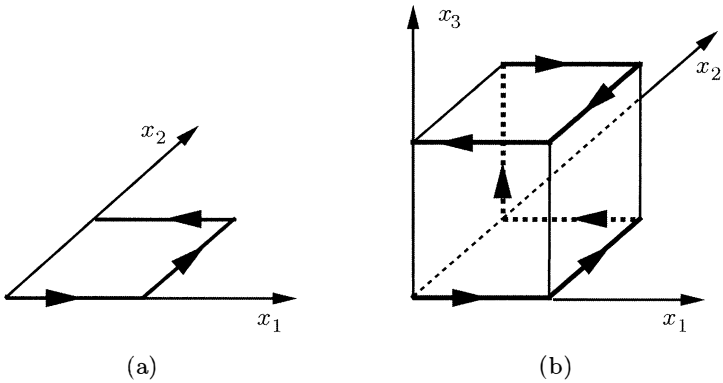


Figure 3.8: (a) A spanning path p_2 in the two-dimensional cube. (b) A spanning path p_3 in the three-dimensional cube. Notice that this path is obtained by splitting the three-dimensional cube into two two-dimensional cubes, following path p_2 in one of them, moving to the other cube, and following p_2 in the reverse order. This construction generalizes and provides a recursive definition of a spanning path for the general n-dimensional cube.

then it can be verified that the cost function decreases strictly with each move along a suitably chosen spanning path. If we start the simplex method at the first vertex on that spanning path and if our pivoting rule is to always move to the next vertex on that path, then the simplex method will require $2^n - 1$ pivots. We summarize this discussion in the following theorem whose proof is left as an exercise (Exercise 3.32).

Theorem 3.5 *Consider the linear programming problem of minimizing $-x_n$ subject to the constraints (3.7)-(3.8). Then:*

(a) *The feasible set has 2^n vertices.*

(b) *The vertices can be ordered so that each one is adjacent to and has lower cost than the previous one.*

(c) *There exists a pivoting rule under which the simplex method requires $2^n - 1$ changes of basis before it terminates.*

We observe in Figure 3.8 that the first and the last vertex in the spanning path are adjacent. This property persists in the perturbed polyhedron as well. Thus, with a different pivoting rule, the simplex method could terminate with a single pivot. We are thus led to the following question: is it true that for every pivoting rule there are examples where the simplex

method takes an exponential number of iterations? For several popular pivoting rules, such examples have been constructed. However, these examples cannot exclude the possibility that some other pivoting rule might fare better. This is one of the most important open problems in the theory of linear programming. In the next subsection, we address a closely related issue.

The diameter of polyhedra and the Hirsch conjecture

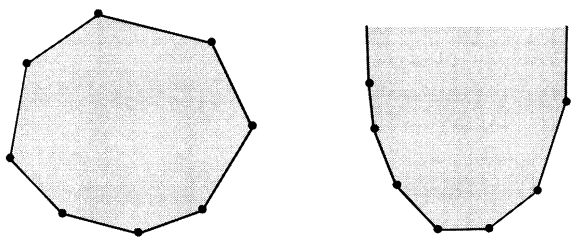
The preceding discussion leads us to the notion of the diameter of a polyhedron P, which is defined as follows. Suppose that from any vertex of the polyhedron, we are only allowed to jump to an adjacent vertex. We define the distance $d(\mathbf{x}, \mathbf{y})$ between two vertices $\mathbf{x}$ and $\mathbf{y}$ as the minimum number of such jumps required to reach $\mathbf{y}$ starting from $\mathbf{x}$. The diameter $D(P)$ of the polyhedron P is then defined as the maximum of $d(\mathbf{x}, \mathbf{y})$ over all pairs $(\mathbf{x}, \mathbf{y})$ of vertices. Finally, we define $\Delta(n, m)$ as the maximum of $D(P)$ over all *bounded* polyhedra in $\Re^n$ that are represented in terms of m inequality constraints. The quantity $\Delta_u(n, m)$ is defined similarly, except that general, possibly unbounded, polyhedra are allowed. For example, we have

$$\Delta(2, m) = \left\lfloor \frac{m}{2} \right\rfloor,$$

and

$$\Delta_u(2, m) = m - 2;$$

see Figure 3.9.



(a) (b)

Figure 3.9: Let $n = 2$ and $m = 8$. (a) A bounded polyhedron with diameter $\lfloor m/2 \rfloor = 4$. (b) An unbounded polyhedron with diameter $m - 2 = 6$.

Suppose that the feasible set in a linear programming problem has diameter d and that the distance between vertices $\mathbf{x}$ and $\mathbf{y}$ is equal to d. If

the simplex method (or any other method that proceeds from one vertex to an adjacent vertex) is initialized at $\mathbf{x}$, and if $\mathbf{y}$ happens to be the unique optimal solution, then at least d steps will be required. Now, if $\Delta(n, m)$ or $\Delta_u(n, m)$ increases exponentially with n and m, this implies that there exist examples for which the simplex method takes an exponentially increasing number of steps, no matter which pivoting rule is used. Thus, in order to have any hope of developing pivoting rules under which the simplex method requires a polynomial number of iterations, we must first establish that $\Delta(n, m)$ or $\Delta_u(n, m)$ grows with n and m at the rate of some polynomial. The practical success of the simplex method has led to the conjecture that indeed $\Delta(n, m)$ and $\Delta_u(n, m)$ do not grow exponentially fast. In fact, the following, much stronger, conjecture has been advanced:

Hirsch Conjecture: $\Delta(n, m) \leq m - n$.

Despite the significance of $\Delta(n, m)$ and $\Delta_u(n, m)$, we are far from establishing the Hirsch conjecture or even from establishing that these quantities exhibit polynomial growth. It is known (Klee and Walkup, 1967) that the Hirsch conjecture is false for unbounded polyhedra and, in particular, that

$$\Delta_u(n, m) \geq m - n + \left\lfloor \frac{n}{5} \right\rfloor .$$

Unfortunately, this is the best lower bound known; even though it disproves the Hirsch conjecture for unbounded polyhedra, it does not provide any insights as to whether the growth of $\Delta_u(n, m)$ is polynomial or exponential.

Regarding upper bounds, it has been established (Kalai and Kleitman, 1993) that the worst-case diameter grows slower than exponentially, but the available upper bound grows faster than any polynomial. In particular, the following bounds are available:

$$\Delta(n, m) \leq \Delta_u(n, m) < m^{1+\log_2 n} = (2n)^{\log_2 m}.$$

The average case behavior of the simplex method

Our discussion has been focused on the worst-case behavior of the simplex method, but this is only part of the story. Even if every pivoting rule requires an exponential number of iterations in the worst case, this is not necessarily relevant to the typical behavior of the simplex method. For this reason, there has been a fair amount of research aiming at an understanding of the typical or average behavior of the simplex method, and an explanation of its observed behavior.

The main difficulty in studying the average behavior of any algorithm lies in defining the meaning of the term "average." Basically, one needs to define a probability distribution over the set of all problems of a given size, and then take the mathematical expectation of the number of iterations

required by the algorithm, when applied to a random problem drawn according to the postulated probability distribution. Unfortunately, there is no natural probability distribution over the set of linear programming problems. Nevertheless, a fair number of positive results have been obtained for a few different types of probability distributions. In one such result, a set of vectors $\mathbf{c}, \mathbf{a}_1, \ldots, \mathbf{a}_m \in \Re^n$ and scalars $b_1, \ldots, b_m$ is given. For $i = 1, \ldots, m$, we introduce either constraint $\mathbf{a}_i'\mathbf{x} \leq b_i$ or $\mathbf{a}_i'\mathbf{x} \geq b_i$, with equal probability. We then have 2^m possible linear programming problems, and suppose that L of them are feasible. Haimovich (1983) has established that under a rather special pivoting rule, the simplex method requires no more than $n/2$ iterations, on the average over those L feasible problems. This linear dependence on the size of the problem agrees with observed behavior; some empirical evidence is discussed in Chapter 12.

3.8 Summary

This chapter was centered on the development of the simplex method, which is a complete algorithm for solving linear programming problems in standard form. The cornerstones of the simplex method are:

(a) the optimality conditions (nonnegativity of the reduced costs) that allow us to test whether the current basis is optimal;

(b) a systematic method for performing basis changes whenever the optimality conditions are violated.

At a high level, the simplex method simply moves from one extreme point of the feasible set to another, each time reducing the cost, until an optimal solution is reached. However, the lower level details of the simplex method, relating to the organization of the required computations and the associated bookkeeping, play an important role. We have described three different implementations: the naive one, the revised simplex method, and the full tableau implementation. Abstractly, they are all equivalent, but their mechanics are quite different. Practical implementations of the simplex method follow our general description of the revised simplex method, but the details are different, because an explicit computation of the inverse basis matrix is usually avoided.

We have seen that degeneracy can cause substantial difficulties, including the possibility of nonterminating behavior (cycling). This is because in the presence of degeneracy, a change of basis may keep us at the same basic feasible solution, with no cost improvement resulting. Cycling can be avoided if suitable rules for choosing the entering and exiting variables (pivoting rules) are applied (e.g., Bland's rule or the lexicographic pivoting rule).

Starting the simplex method requires an initial basic feasible solution, and an associated tableau. These are provided by the Phase I simplex algorithm, which is nothing but the simplex method applied to an auxiliary

problem. We saw that the changeover from Phase I to Phase II involves some delicate steps whenever some artificial variables are in the final basis constructed by the Phase I algorithm.

The simplex method is a rather efficient algorithm and is incorporated in most of the commercial codes for linear programming. While the number of pivots can be an exponential function of the number of variables and constraints in the worst case, its observed behavior is a lot better, hence the practical usefulness of the method.

3.9 Exercises

Exercise 3.1 (Local minima of convex functions) Let $f : \Re^n \mapsto \Re$ be a convex function and let $S \subset \Re^n$ be a convex set. Let $\mathbf{x}^*$ be an element of S. Suppose that $\mathbf{x}^*$ is a local optimum for the problem of minimizing $f(\mathbf{x})$ over S; that is, there exists some $\epsilon > 0$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in S$ for which $\|\mathbf{x} - \mathbf{x}^*\| \leq \epsilon$. Prove that $\mathbf{x}^*$ is globally optimal; that is, $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in S$.

Exercise 3.2 (Optimality conditions) Consider the problem of minimizing $\mathbf{c}'\mathbf{x}$ over a polyhedron P. Prove the following:

(a) A feasible solution $\mathbf{x}$ is optimal if and only if $\mathbf{c}'\mathbf{d} \geq 0$ for every feasible direction $\mathbf{d}$ at $\mathbf{x}$.

(b) A feasible solution $\mathbf{x}$ is the unique optimal solution if and only if $\mathbf{c}'\mathbf{d} > 0$ for every nonzero feasible direction $\mathbf{d}$ at $\mathbf{x}$.

Exercise 3.3 Let $\mathbf{x}$ be an element of the standard form polyhedron $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$. Prove that a vector $\mathbf{d} \in \Re^n$ is a feasible direction at $\mathbf{x}$ if and only if $\mathbf{A}\mathbf{d} = \mathbf{0}$ and $d_i \geq 0$ for every i such that $x_i = 0$.

Exercise 3.4 Consider the problem of minimizing $\mathbf{c}'\mathbf{x}$ over the set $P = \left\{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{D}\mathbf{x} \leq \mathbf{f},\ \mathbf{E}\mathbf{x} \leq \mathbf{g}\right\}$. Let $\mathbf{x}^*$ be an element of P that satisfies $\mathbf{D}\mathbf{x}^* = \mathbf{f}$, $\mathbf{E}\mathbf{x}^* < \mathbf{g}$. Show that the set of feasible directions at the point $\mathbf{x}^*$ is the set

$$\left\{\mathbf{d} \in \Re^n \mid \mathbf{A}\mathbf{d} = \mathbf{0},\ \mathbf{D}\mathbf{d} \leq \mathbf{0}\right\}.$$

Exercise 3.5 Let $P = \left\{\mathbf{x} \in \Re^3 \mid x_1 + x_2 + x_3 = 1,\ \mathbf{x} \geq \mathbf{0}\right\}$ and consider the vector $\mathbf{x} = (0, 0, 1)$. Find the set of feasible directions at $\mathbf{x}$.

Exercise 3.6 (Conditions for a unique optimum) Let $\mathbf{x}$ be a basic feasible solution associated with some basis matrix $\mathbf{B}$. Prove the following:

(a) If the reduced cost of every nonbasic variable is positive, then $\mathbf{x}$ is the unique optimal solution.

(b) If $\mathbf{x}$ is the unique optimal solution and is nondegenerate, then the reduced cost of every nonbasic variable is positive.

Exercise 3.7 (Optimality conditions) Consider a feasible solution $\mathbf{x}$ to a standard form problem, and let $Z = \{i \mid x_i = 0\}$. Show that $\mathbf{x}$ is an optimal solution if and only if the linear programming problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{d} \\ \text{subject to} \quad & \mathbf{A}\mathbf{d} = \mathbf{0} \\ & d_i \geq 0, \qquad i \in Z, \end{aligned}$$

has an optimal cost of zero. (In this sense, deciding optimality is equivalent to solving a new linear programming problem.)

Exercise 3.8* This exercise deals with the problem of deciding whether a given degenerate basic feasible solution is optimal and shows that this is essentially as hard as solving a general linear programming problem.

Consider the linear programming problem of minimizing $\mathbf{c}'\mathbf{x}$ over all $\mathbf{x} \in P$, where $P \subset \Re^n$ is a given bounded polyhedron. Let

$$Q = \left\{(t\mathbf{x}, t) \in \Re^{n+1} \;\middle|\; \mathbf{x} \in P,\ t \in [0,1]\right\}.$$

(a) Show that Q is a polyhedron.

(b) Give an example of P and Q, with $n = 2$, for which the zero vector (in $\Re^{n+1}$) is a degenerate basic feasible solution in Q; show the example in a figure.

(c) Show that the zero vector (in $\Re^{n+1}$) minimizes $(\mathbf{c}, 0)'\mathbf{y}$ over all $\mathbf{y} \in Q$ if and only if the optimal cost in the original linear programming problem is greater than or equal to zero.

Exercise 3.9 (Necessary and sufficient conditions for a unique optimum) Consider a linear programming problem in standard form and suppose that $\mathbf{x}^*$ is an optimal basic feasible solution. Consider an optimal basis associated with $\mathbf{x}^*$. Let B and N be the set of basic and nonbasic indices, respectively. Let I be the set of nonbasic indices i for which the corresponding reduced costs $\bar{c}_i$ are zero.

(a) Show that if I is empty, then $\mathbf{x}^*$ is the only optimal solution.

(b) Show that $\mathbf{x}^*$ is the unique optimal solution if and only if the following linear programming problem has an optimal value of zero:

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in I} x_i \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & x_i = 0, \qquad i \in N \setminus I, \\ & x_i \geq 0, \qquad i \in B \cup I. \end{aligned}$$

Exercise 3.10 * Show that if $n - m = 2$, then the simplex method will not cycle, no matter which pivoting rule is used.

Exercise 3.11 * Construct an example with $n - m = 3$ and a pivoting rule under which the simplex method will cycle.

Exercise 3.12 Consider the problem

$$\begin{array}{lrcl} \text{minimize} & -2x_1 - x_2 & & \\ \text{subject to} & x_1 - x_2 & \leq & 2 \\ & x_1 + x_2 & \leq & 6 \\ & x_1, x_2 \geq 0. & & \end{array}$$

(a) Convert the problem into standard form and construct a basic feasible solution at which $(x_1, x_2) = (0, 0)$.

(b) Carry out the full tableau implementation of the simplex method, starting with the basic feasible solution of part (a).

(c) Draw a graphical representation of the problem in terms of the original variables x_1, x_2, and indicate the path taken by the simplex algorithm.

Exercise 3.13 This exercise shows that our efficient procedures for updating a tableau can be derived from a useful fact in numerical linear algebra.

(a) **(Matrix inversion lemma)** Let $\mathbf{C}$ be an $m \times m$ invertible matrix and let $\mathbf{u}$, $\mathbf{v}$ be vectors in $\Re^m$. Show that

$$(\mathbf{C} + \mathbf{w}\mathbf{v}')^{-1} = \mathbf{C}^{-1} - \frac{\mathbf{C}^{-1}\mathbf{w}\mathbf{v}'\mathbf{C}^{-1}}{1 + \mathbf{v}'\mathbf{C}^{-1}\mathbf{w}}.$$

(Note that $\mathbf{w}\mathbf{v}'$ is an $m \times m$ matrix). *Hint:* Multiply both sides by $(\mathbf{C} + \mathbf{w}\mathbf{v}')$.

(b) Assuming that $\mathbf{C}^{-1}$ is available, explain how to obtain $(\mathbf{C} + \mathbf{w}\mathbf{v}')^{-1}$ using only $O(m^2)$ arithmetic operations.

(c) Let $\mathbf{B}$ and $\overline{\mathbf{B}}$ be basis matrices before and after an iteration of the simplex method. Let $\mathbf{A}_{B(\ell)}$, $\mathbf{A}_{\overline{B}(\ell)}$ be the exiting and entering column, respectively. Show that

$$\overline{\mathbf{B}} - \mathbf{B} = (\mathbf{A}_{\overline{B}(\ell)} - \mathbf{A}_{B(\ell)})\mathbf{e}_\ell',$$

where $\mathbf{e}_\ell$ is the ℓth unit vector.

(d) Note that $\mathbf{e}_i'\mathbf{B}^{-1}$ is the ith row of $\mathbf{B}^{-1}$ and $\mathbf{e}_\ell'\mathbf{B}^{-1}$ is the pivot row. Show that

$$\mathbf{e}_i'\overline{\mathbf{B}}^{-1} = \mathbf{e}_i'\mathbf{B}^{-1} - g_i\mathbf{e}_\ell'\mathbf{B}^{-1}, \qquad i = 1, \ldots, m,$$

for suitable scalars g_i. Provide a formula for g_i. Interpret the above equation in terms of the mechanics for pivoting in the revised simplex method.

(e) Multiply both sides of the equation in part (d) by $[\mathbf{b} \mid \mathbf{A}]$ and obtain an interpretation of the mechanics for pivoting in the full tableau implementation.

Exercise 3.14 Suppose that a feasible tableau is available. Show how to obtain a tableau with lexicographically positive rows. *Hint:* Permute the columns.

Exercise 3.15 (Perturbation approach to lexicography) Consider a standard form problem, under the usual assumption that the rows of $\mathbf{A}$ are linearly independent. Let ϵ be a scalar and define

$$\mathbf{b}(\epsilon) = \mathbf{b} + \begin{bmatrix} \epsilon \\ \epsilon^2 \\ \vdots \\ \epsilon^m \end{bmatrix}.$$

For every $\epsilon > 0$, we define the ϵ-perturbed problem to be the linear programming problem obtained by replacing $\mathbf{b}$ with $\mathbf{b}(\epsilon)$.

(a) Given a basis matrix $\mathbf{B}$, show that the corresponding basic solution $\mathbf{x}_B(\epsilon)$ in the ϵ-perturbed problem is equal to

$$\mathbf{B}^{-1}\big[\mathbf{b} \mid \mathbf{I}\big] \begin{bmatrix} 1 \\ \epsilon \\ \vdots \\ \epsilon^m \end{bmatrix}.$$

(b) Show that there exists some $\epsilon^* > 0$ such that all basic solutions to the ϵ-perturbed problem are nondegenerate, for $0 < \epsilon < \epsilon^*$.

(c) Suppose that all rows of $\mathbf{B}^{-1}\big[\mathbf{b} \mid \mathbf{I}\big]$ are lexicographically positive. Show that $\mathbf{x}_B(\epsilon)$ is a basic feasible solution to the ϵ-perturbed problem for ϵ positive and sufficiently small.

(d) Consider a feasible basis for the original problem, and assume that all rows of $\mathbf{B}^{-1}[\mathbf{b} \mid \mathbf{I}]$ are lexicographically positive. Let some nonbasic variable x_j enter the basis, and define $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$. Let the exiting variable be determined as follows. For every row i such that u_i is positive, divide the ith row of $\mathbf{B}^{-1}[\mathbf{b} \mid \mathbf{I}]$ by u_i, compare the results lexicographically, and choose the exiting variable to be the one corresponding to the lexicographically smallest row. Show that this is the same choice of exiting variable as in the original simplex method applied to the ϵ-perturbed problem, when ϵ is sufficiently small.

(e) Explain why the revised simplex method, with the lexicographic rule described in part (d), is guaranteed to terminate even in the face of degeneracy.

Exercise 3.16 (Lexicography and the revised simplex method) Suppose that we have a basic feasible solution and an associated basis matrix $\mathbf{B}$ such that every row of $\mathbf{B}^{-1}$ is lexicographically positive. Consider a pivoting rule that chooses the entering variable x_j arbitrarily (as long as $\bar{c}_j < 0$) and the exiting variable as follows. Let $\mathbf{u} = \mathbf{B}^{-1}\mathbf{A}_j$. For each i with $u_i > 0$, divide the ith row of $[\mathbf{B}^{-1}\mathbf{b} \mid \mathbf{B}^{-1}]$ by u_i and choose the row which is lexicographically smallest. If row ℓ was lexicographically smallest, then the ℓth basic variable $x_{B(\ell)}$ exits the basis. Prove the following:

(a) The row vector $(-\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{b}, -\mathbf{c}_B'\mathbf{B}^{-1})$ increases lexicographically at each iteration.

(b) Every row of $\mathbf{B}^{-1}$ is lexicographically positive throughout the algorithm.

(c) The revised simplex method terminates after a finite number of steps.

Exercise 3.17 Solve completely (i.e., both Phase I and Phase II) via the simplex method the following problem:

$$\begin{array}{lrcrcrcrcl} \text{minimize} & 2x_1 & + & 3x_2 & + & 3x_3 & + & x_4 & - & 2x_5 & \\ \text{subject to} & x_1 & + & 3x_2 & & & + & 4x_4 & + & x_5 & = 2 \\ & x_1 & + & 2x_2 & & & - & 3x_4 & + & x_5 & = 2 \\ & -x_1 & - & 4x_2 & + & 3x_3 & & & & & = 1 \\ & x_1, \ldots, x_5 & \geq & 0. & & & & & & & \end{array}$$

Exercise 3.18 Consider the simplex method applied to a standard form problem and assume that the rows of the matrix $\mathbf{A}$ are linearly independent. For each of the statements that follow, give either a proof or a counterexample.

(a) An iteration of the simplex method may move the feasible solution by a positive distance while leaving the cost unchanged.

(b) A variable that has just left the basis cannot reenter in the very next iteration.

(c) A variable that has just entered the basis cannot leave in the very next iteration.

(d) If there is a nondegenerate optimal basis, then there exists a unique optimal basis.

(e) If $\mathbf{x}$ is an optimal solution, no more than m of its components can be positive, where m is the number of equality constraints.

Exercise 3.19 While solving a standard form problem, we arrive at the following tableau, with x_3, x_4, and x_5 being the basic variables:

-10	δ	-2	0	0	0
4	-1	η	1	0	0
1	α	-4	0	1	0
β	γ	3	0	0	1

The entries α, β, γ, δ, η in the tableau are unknown parameters. For each one of the following statements, find some parameter values that will make the statement true.

(a) The current solution is optimal and there are multiple optimal solutions.

(b) The optimal cost is $-\infty$.

(c) The current solution is feasible but not optimal.

Exercise 3.20 Consider a linear programming problem in standard form, described in terms of the following initial tableau:

0	0	0	0	δ	3	γ	ξ
β	0	1	0	α	1	0	3
2	0	0	1	-2	2	η	-1
3	1	0	0	0	-1	2	1

The entries α, β, γ, δ, η, ξ in the tableau are unknown parameters. Furthermore, let $\mathbf{B}$ be the basis matrix corresponding to having x_2, x_3, and x_1 (in that order) be the basic variables. For each one of the following statements, find the ranges of values of the various parameters that will make the statement to be true.

(a) Phase II of the simplex method can be applied using this as an initial tableau.

(b) The first row in the present tableau indicates that the problem is infeasible.

(c) The corresponding basic solution is feasible, but we do not have an optimal basis.

(d) The corresponding basic solution is feasible and the first simplex iteration indicates that the optimal cost is $-\infty$.

(e) The corresponding basic solution is feasible, x_6 is a candidate for entering the basis, and when x_6 is the entering variable, x_3 leaves the basis.

(f) The corresponding basic solution is feasible, x_7 is a candidate for entering the basis, but if it does, the solution and the objective value remain unchanged.

Exercise 3.21 Consider the oil refinery problem in Exercise 1.16.

(a) Use the simplex method to find an optimal solution.

(b) Suppose that the selling price of heating oil is sure to remain fixed over the next month, but the selling price of gasoline may rise. How high can it go without causing the optimal solution to change?

(c) The refinery manager can buy crude oil B on the spot market at \$40/barrel, in unlimited quantities. How much should be bought?

Exercise 3.22 Consider the following linear programming problem with a single constraint:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{n} c_i x_i \\ \text{subject to} \quad & \sum_{i=1}^{n} a_i x_i = b \\ & x_i \geq 0, \qquad i = 1, \ldots, n. \end{aligned}$$

(a) Derive a simple test for checking the feasibility of this problem.

(b) Assuming that the optimal cost is finite, develop a simple method for obtaining an optimal solution directly.

Exercise 3.23 While solving a linear programming problem by the simplex method, the following tableau is obtained at some iteration.

	0	...	0	$\bar{c}_{m+1}$	...	$\bar{c}_n$
x_1	1	...	0	$a_{1,m+1}$	...	$a_{1,n}$
$\vdots$	$\vdots$	...	$\vdots$	$\vdots$	...	$\vdots$
x_m	0	...	1	$a_{m,m+1}$	...	$a_{m,n}$

Assume that in this tableau we have $\bar{c}_j \geq 0$ for $j = m+1, \ldots, n-1$, and $\bar{c}_n < 0$. In particular, x_n is the only candidate for entering the basis.

(a) Suppose that x_n indeed enters the basis and that this is a nondegenerate pivot (that is, $\theta^* \neq 0$). Prove that x_n will remain basic in all subsequent

iterations of the algorithm and that x_n is a basic variable in any optimal basis.

(b) Suppose that x_n indeed enters the basis and that this is a degenerate pivot (that is, $\theta^* = 0$). Show that x_n need not be basic in an optimal basic feasible solution.

Exercise 3.24 Show that in Phase I of the simplex method, if an artificial variable becomes nonbasic, it need never again become basic. Thus, when an artificial variable becomes nonbasic, its column can be eliminated from the tableau.

Exercise 3.25 (The simplex method with upper bound constraints) Consider a problem of the form

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{0} \leq \mathbf{x} \leq \mathbf{u}, \end{aligned}$$

where $\mathbf{A}$ has linearly independent rows and dimensions $m \times n$. Assume that $u_i > 0$ for all i.

(a) Let $\mathbf{A}_{B(1)} \ldots, \mathbf{A}_{B(m)}$ be m linearly independent columns of $\mathbf{A}$ (the "basic" columns). We partition the set of all $i \neq B(1), \ldots, B(m)$ into two disjoint subsets L and U. We set $x_i = 0$ for all $i \in L$, and $x_i = u_i$ for all $i \in U$. We then solve the equation $\mathbf{A}\mathbf{x} = \mathbf{b}$ for the basic variables $x_{B(1)}, \ldots, x_{B(m)}$. Show that the resulting vector $\mathbf{x}$ is a basic solution. Also, show that it is nondegenerate if and only if $0 < x_i < u_i$ for every basic variable x_i.

(b) For this part and the next, assume that the basic solution constructed in part (a) is feasible. We form the simplex tableau and compute the reduced costs as usual. Let x_j be some nonbasic variable such that $x_j = 0$ and $\bar{c}_j < 0$. As in Section 3.2, we increase x_j by θ, and adjust the basic variables from $\mathbf{x}_B$ to $\mathbf{x}_B - \theta\mathbf{B}^{-1}\mathbf{A}_j$. Given that we wish to preserve feasibility, what is the largest possible value of θ? How are the new basic columns determined?

(c) Let x_j be some nonbasic variable such that $x_j = u_j$ and $\bar{c}_j > 0$. We decrease x_j by θ, and adjust the basic variables from $\mathbf{x}_B$ to $\mathbf{x}_B + \theta\mathbf{B}^{-1}\mathbf{A}_j$. Given that we wish to preserve feasibility, what is the largest possible value of θ? How are the new basic columns determined?

(d) Assuming that every basic feasible solution is nondegenerate, show that the cost strictly decreases with each iteration and the method terminates.

Exercise 3.26 (The big-M method) Consider the variant of the big-M method in which M is treated as an undetermined large parameter. Prove the following.

(a) If the simplex method terminates with a solution $(\mathbf{x}, \mathbf{y})$ for which $\mathbf{y} = \mathbf{0}$, then $\mathbf{x}$ is an optimal solution to the original problem.

(b) If the simplex method terminates with a solution $(\mathbf{x}, \mathbf{y})$ for which $\mathbf{y} \neq \mathbf{0}$, then the original problem is infeasible.

(c) If the simplex method terminates with an indication that the optimal cost in the auxiliary problem is $-\infty$, show that the original problem is either

infeasible or its optimal cost is $-\infty$. *Hint:* When the simplex method terminates, it has discovered a feasible direction $\mathbf{d} = (\mathbf{d}_x, \mathbf{d}_y)$ of cost decrease. Show that $\mathbf{d}_y = \mathbf{0}$.

(d) Provide examples to show that both alternatives in part (c) are possible.

Exercise 3.27 *

(a) Suppose that we wish to find a vector $\mathbf{x} \in \Re^n$ that satisfies $\mathbf{A}\mathbf{x} = \mathbf{0}$ and $\mathbf{x} \geq \mathbf{0}$, and such that the number of positive components of $\mathbf{x}$ is maximized. Show that this can be accomplished by solving the linear programming problem

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} y_i \\ \text{subject to} \quad & \mathbf{A}(\mathbf{z} + \mathbf{y}) = \mathbf{0} \\ & y_i \leq 1, \qquad \text{for all } i, \\ & \mathbf{z}, \mathbf{y} \geq \mathbf{0}. \end{aligned}$$

(b) Suppose that we wish to find a vector $\mathbf{x} \in \Re^n$ that satisfies $\mathbf{A}\mathbf{x} = \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$, and such that the number of positive components of $\mathbf{x}$ is maximized. Show how this can be accomplished by solving a single linear programming problem.

Exercise 3.28 Consider a linear programming problem in standard form with a bounded feasible set. Furthermore, suppose that we know the value of a scalar U such that any feasible solution satisfies $x_i \leq U$, for all i. Show that the problem can be transformed into an equivalent one that contains the constraint $\sum_{i=1}^{n} x_i = 1$.

Exercise 3.29 Consider the simplex method, viewed in terms of column geometry. Show that the $m + 1$ basic points $(\mathbf{A}_i, c_i)$, as defined in Section 3.6, are affinely independent.

Exercise 3.30 Consider the simplex method, viewed in terms of column geometry. In the terminology of Section 3.6, show that the vertical distance from the dual plane to a point $(\mathbf{A}_j, c_j)$ is equal to the reduced cost of the variable x_j.

Exercise 3.31 Consider the linear programming problem

$$\begin{aligned} \text{minimize} \quad & x_1 + 3x_2 + 2x_3 + 2x_4 \\ \text{subject to} \quad & 2x_1 + 3x_2 + x_3 + x_4 = b_1 \\ & x_1 + 2x_2 + x_3 + 3x_4 = b_2 \\ & x_1 + x_2 + x_3 + x_4 = 1 \\ & x_1, \ldots, x_4 \geq 0, \end{aligned}$$

where b_1, b_2 are free parameters. Let $P(b_1, b_2)$ be the feasible set. Use the column geometry of linear programming to answer the following questions.

(a) Characterize explicitly (preferably with a picture) the set of all (b_1, b_2) for which $P(b_1, b_2)$ is nonempty.

(b) Characterize explicitly (preferably with a picture) the set of all (b_1, b_2) for which some basic feasible solution is degenerate.

(c) There are four bases in this problem; in the ith basis, all variables except for x_i are basic. For every (b_1, b_2) for which there exists a degenerate basic feasible solution, enumerate all bases that correspond to each degenerate basic feasible solution.

(d) For $i = 1, \ldots, 4$, let $S_i = \{(b_1, b_2) \mid \text{the } i\text{th basis is optimal}\}$. Identify, preferably with a picture, the sets $S_1, \ldots, S_4$.

(e) For which values of (b_1, b_2) is the optimal solution degenerate?

(f) Let $b_1 = 9/5$ and $b_2 = 7/5$. Suppose that we start the simplex method with x_2, x_3, x_4 as the basic variables. Which path will the simplex method follow?

Exercise 3.32* Prove Theorem 3.5.

Exercise 3.33 Consider a polyhedron in standard form, and let $\mathbf{x}$, $\mathbf{y}$ be two different basic feasible solutions. If we are allowed to move from any basic feasible solution to an adjacent one in a single step, show that we can go from $\mathbf{x}$ to $\mathbf{y}$ in a finite number of steps.

3.10 Notes and sources

3.2. The simplex method was pioneered by Dantzig in 1947, who later wrote a comprehensive text on the subject (Dantzig, 1963).

3.3. For more discussion of practical implementations of the simplex method based on products of sparse matrices, instead of $\mathbf{B}^{-1}$, see the books by Gill, Murray, and Wright (1981), Chvátal (1983), Murty (1983), and Luenberger (1984). An excellent introduction to numerical linear algebra is the text by Golub and Van Loan (1983). Example 3.6, which shows the possibility of cycling, is due to Beale (1955).

If we have upper bounds for all or some of the variables, instead of converting the problem to standard form, we can use a suitable adaptation of the simplex method. This is developed in Exercise 3.25 and in the textbooks that we mentioned earlier.

3.4. The lexicographic anticycling rule is due to Dantzig, Orden, and Wolfe (1955). It can be viewed as an outgrowth of a perturbation method developed by Orden and also by Charnes (1952). For an exposition of the perturbation method, see Chvátal (1983) and Murty (1983), as well as Exercise 3.15. The smallest subscript rule is due to Bland (1977). A proof that Bland's rule avoids cycling can also be found in Papadimitriou and Steiglitz (1982), Chvátal (1983), or Murty (1983).

3.6. The column geometry interpretation of the simplex method is due to Dantzig (1963). For further discussion, see Stone and Tovey (1991).

3.7. The example showing that the simplex method can take an exponential number of iterations is due to Klee and Minty (1972). The Hirsch conjecture was made by Hirsch in 1957. The first results on the average case behavior of the simplex method were obtained by Borgwardt (1982) and Smale (1983). Schrijver (1986) contains an overview of the early research in this area, as well as proof of the $n/2$ bound on the number of pivots due to Haimovich (1983).

3.9. The results in Exercises 3.10 and 3.11, which deal with the smallest examples of cycling, are due to Marshall and Suurballe (1969). The matrix inversion lemma [Exercise 3.13(a)] is known as the Sherman-Morrison formula.

Chapter 4

Duality theory

Contents

In this chapter, we start with a linear programming problem, called the primal, and introduce another linear programming problem, called the dual. Duality theory deals with the relation between these two problems and uncovers the deeper structure of linear programming. It is a powerful theoretical tool that has numerous applications, provides new geometric insights, and leads to another algorithm for linear programming (the dual simplex method).

4.1 Motivation

Duality theory can be motivated as an outgrowth of the Lagrange multiplier method, often used in calculus to minimize a function subject to equality constraints. For example, in order to solve the problem

$$\begin{aligned} \text{minimize} \quad & x^2 + y^2 \\ \text{subject to} \quad & x + y = 1, \end{aligned}$$

we introduce a Lagrange multiplier p and form the Lagrangean $L(x, y, p)$ defined by

$$L(x, y, p) = x^2 + y^2 + p(1 - x - y).$$

While keeping p fixed, we minimize the Lagrangean over all x and y, subject to no constraints, which can be done by setting $\partial L/\partial x$ and $\partial L/\partial y$ to zero. The optimal solution to this unconstrained problem is

$$x = y = \frac{p}{2},$$

and depends on p. The constraint $x + y = 1$ gives us the additional relation $p = 1$, and the optimal solution to the original problem is $x = y = 1/2$.

The main idea in the above example is the following. Instead of enforcing the hard constraint $x + y = 1$, we allow it to be violated and associate a Lagrange multiplier, or *price*, p with the amount $1 - x - y$ by which it is violated. This leads to the unconstrained minimization of $x^2 + y^2 + p(1 - x - y)$. When the price is properly chosen ($p = 1$, in our example), the optimal solution to the constrained problem is also optimal for the unconstrained problem. In particular, under that specific value of p, the presence or absence of the hard constraint does not affect the optimal cost.

The situation in linear programming is similar: we associate a price variable with each constraint and start searching for prices under which the presence or absence of the constraints does not affect the optimal cost. It turns out that the right prices can be found by solving a new linear programming problem, called the *dual* of the original. We now motivate the form of the dual problem.

Consider the standard form problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

which we call the *primal* problem, and let $\mathbf{x}^*$ be an optimal solution, assumed to exist. We introduce a *relaxed* problem in which the constraint $\mathbf{A}\mathbf{x} = \mathbf{b}$ is replaced by a penalty $\mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x})$, where $\mathbf{p}$ is a price vector of the same dimension as $\mathbf{b}$. We are then faced with the problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Let $g(\mathbf{p})$ be the optimal cost for the relaxed problem, as a function of the price vector $\mathbf{p}$. The relaxed problem allows for more options than those present in the primal problem, and we expect $g(\mathbf{p})$ to be no larger than the optimal primal cost $\mathbf{c}'\mathbf{x}^*$. Indeed,

$$g(\mathbf{p}) = \min_{\mathbf{x} \geq \mathbf{0}} \Big[\mathbf{c}'\mathbf{x} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}) \Big] \leq \mathbf{c}'\mathbf{x}^* + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}^*) = \mathbf{c}'\mathbf{x}^*,$$

where the last inequality follows from the fact that $\mathbf{x}^*$ is a feasible solution to the primal problem, and satisfies $\mathbf{A}\mathbf{x}^* = \mathbf{b}$. Thus, each $\mathbf{p}$ leads to a lower bound $g(\mathbf{p})$ for the optimal cost $\mathbf{c}'\mathbf{x}^*$. The problem

$$\begin{aligned} \text{maximize} \quad & g(\mathbf{p}) \\ \text{subject to} \quad & \text{no constraints} \end{aligned}$$

can be then interpreted as a search for the tightest possible lower bound of this type, and is known as the *dual* problem. The main result in duality theory asserts that the optimal cost in the dual problem is equal to the optimal cost $\mathbf{c}'\mathbf{x}^*$ in the primal. In other words, when the prices are chosen according to an optimal solution for the dual problem, the option of violating the constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$ is of no value.

Using the definition of $g(\mathbf{p})$, we have

$$\begin{aligned} g(\mathbf{p}) &= \min_{\mathbf{x} \geq \mathbf{0}} \Big[\mathbf{c}'\mathbf{x} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}) \Big] \\ &= \mathbf{p}'\mathbf{b} + \min_{\mathbf{x} \geq \mathbf{0}} (\mathbf{c}' - \mathbf{p}'\mathbf{A})\mathbf{x}. \end{aligned}$$

Note that

$$\min_{\mathbf{x} \geq \mathbf{0}} (\mathbf{c}' - \mathbf{p}'\mathbf{A})\mathbf{x} = \begin{cases} 0, & \text{if } \mathbf{c}' - \mathbf{p}'\mathbf{A} \geq \mathbf{0}', \\ -\infty, & \text{otherwise.} \end{cases}$$

In maximizing $g(\mathbf{p})$, we only need to consider those values of $\mathbf{p}$ for which $g(\mathbf{p})$ is not equal to $-\infty$. We therefore conclude that the dual problem is the same as the linear programming problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} \leq \mathbf{c}'. \end{aligned}$$

In the preceding example, we started with the equality constraint $\mathbf{Ax} = \mathbf{b}$ and we ended up with no constraints on the sign of the price vector $\mathbf{p}$. If the primal problem had instead inequality constraints of the form $\mathbf{Ax} \geq \mathbf{b}$, they could be replaced by $\mathbf{Ax} - \mathbf{s} = \mathbf{b}$, $\mathbf{s} \geq \mathbf{0}$. The equality constraint can be written in the form

$$[\mathbf{A} \mid -\mathbf{I}] \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix} = \mathbf{0},$$

which leads to the dual constraints

$$\mathbf{p}'[\mathbf{A} \mid -\mathbf{I}] \leq [\mathbf{c}' \mid \mathbf{0}'],$$

or, equivalently,

$$\mathbf{p}'\mathbf{A} \leq \mathbf{c}', \quad \mathbf{p} \geq \mathbf{0}.$$

Also, if the vector $\mathbf{x}$ is free rather than sign-constrained, we use the fact

$$\min_{\mathbf{x}} (\mathbf{c}' - \mathbf{p}'\mathbf{A})\mathbf{x} = \begin{cases} 0, & \text{if } \mathbf{c}' - \mathbf{p}'\mathbf{A} = \mathbf{0}', \\ -\infty, & \text{otherwise,} \end{cases}$$

to end up with the constraint $\mathbf{p}'\mathbf{A} = \mathbf{c}'$ in the dual problem. These considerations motivate the general form of the dual problem which we introduce in the next section.

In summary, the construction of the dual of a primal minimization problem can be viewed as follows. We have a vector of parameters (dual variables) $\mathbf{p}$, and for every $\mathbf{p}$ we have a method for obtaining a lower bound on the optimal primal cost. The dual problem is a maximization problem that looks for the tightest such lower bound. For some vectors $\mathbf{p}$, the corresponding lower bound is equal to $-\infty$, and does not carry any useful information. Thus, we only need to maximize over those $\mathbf{p}$ that lead to nontrivial lower bounds, and this is what gives rise to the dual constraints.

4.2 The dual problem

Let $\mathbf{A}$ be a matrix with rows $\mathbf{a}_i'$ and columns $\mathbf{A}_j$. Given a *primal* problem with the structure shown on the left, its *dual* is defined to be the maximization problem shown on the right:

minimize	$\mathbf{c}'\mathbf{x}$		maximize	$\mathbf{p}'\mathbf{b}$	
subject to	$\mathbf{a}_i'\mathbf{x} \geq b_i$,	$i \in M_1$,	subject to	$p_i \geq 0$,	$i \in M_1$,
	$\mathbf{a}_i'\mathbf{x} \leq b_i$,	$i \in M_2$,		$p_i \leq 0$,	$i \in M_2$,
	$\mathbf{a}_i'\mathbf{x} = b_i$,	$i \in M_3$,		p_i free,	$i \in M_3$,
	$x_j \geq 0$,	$j \in N_1$,		$\mathbf{p}'\mathbf{A}_j \leq c_j$,	$j \in N_1$,
	$x_j \leq 0$,	$j \in N_2$,		$\mathbf{p}'\mathbf{A}_j \geq c_j$,	$j \in N_2$,
	x_j free,	$j \in N_3$,		$\mathbf{p}'\mathbf{A}_j = c_j$,	$j \in N_3$.

Notice that for each constraint in the primal (other than the sign constraints), we introduce a variable in the dual problem; for each variable in the primal, we introduce a constraint in the dual. Depending on whether the primal constraint is an equality or inequality constraint, the corresponding dual variable is either free or sign-constrained, respectively. In addition, depending on whether a variable in the primal problem is free or sign-constrained, we have an equality or inequality constraint, respectively, in the dual problem. We summarize these relations in Table 4.1.

PRIMAL	minimize	maximize	DUAL
constraints	$\geq b_i$	≥ 0	**variables**
	$\leq b_i$	≤ 0	
	$= b_i$	free	
variables	≥ 0	$\leq c_j$	**constraints**
	≤ 0	$\geq c_j$	
	free	$= c_j$	

Table 4.1: Relation between primal and dual variables and constraints.

If we start with a maximization problem, we can always convert it into an equivalent minimization problem, and then form its dual according to the rules we have described. However, to avoid confusion, we will adhere to the convention that the primal is a minimization problem, and its dual is a maximization problem. Finally, we will keep referring to the objective function in the dual problem as a "cost" that is being maximized.

A problem and its dual can be stated more compactly, in matrix notation, if a particular form is assumed for the primal. We have, for example, the following pairs of primal and dual problems:

$$\begin{aligned} &\text{minimize} && \mathbf{c}'\mathbf{x} \\ &\text{subject to} && \mathbf{A}\mathbf{x} = \mathbf{b} \\ & && \mathbf{x} \geq \mathbf{0}, \end{aligned} \qquad \begin{aligned} &\text{maximize} && \mathbf{p}'\mathbf{b} \\ &\text{subject to} && \mathbf{p}'\mathbf{A} \leq \mathbf{c}', \end{aligned}$$

and

$$\begin{aligned} &\text{minimize} && \mathbf{c}'\mathbf{x} \\ &\text{subject to} && \mathbf{A}\mathbf{x} \geq \mathbf{b}, \end{aligned} \qquad \begin{aligned} &\text{maximize} && \mathbf{p}'\mathbf{b} \\ &\text{subject to} && \mathbf{p}'\mathbf{A} = \mathbf{c}' \\ & && \mathbf{p} \geq \mathbf{0}. \end{aligned}$$

Example 4.1 Consider the primal problem shown on the left and its dual shown

on the right:

$$\begin{array}{lrlrl} \text{minimize} & x_1 + 2x_2 + 3x_3 & & \text{maximize} & 5p_1 + 6p_2 + 4p_3 \\ \text{subject to} & -x_1 + 3x_2 = 5 & & \text{subject to} & p_1 \text{ free} \\ & 2x_1 - x_2 + 3x_3 \geq 6 & & & p_2 \geq 0 \\ & x_3 \leq 4 & & & p_3 \leq 0 \\ & x_1 \geq 0 & & & -p_1 + 2p_2 \leq 1 \\ & x_2 \leq 0 & & & 3p_1 - p_2 \geq 2 \\ & x_3 \text{ free}, & & & 3p_2 + p_3 = 3. \end{array}$$

We transform the dual into an equivalent minimization problem, rename the variables from p_1, p_2, p_3 to x_1, x_2, x_3, and multiply the three last constraints by -1. The resulting problem is shown on the left. Then, on the right, we show its dual:

$$\begin{array}{lrlrl} \text{minimize} & -5x_1 - 6x_2 - 4x_3 & & \text{maximize} & -p_1 - 2p_2 - 3p_3 \\ \text{subject to} & x_1 \text{ free} & & \text{subject to} & p_1 - 3p_2 = -5 \\ & x_2 \geq 0 & & & -2p_1 + p_2 - 3p_3 \leq -6 \\ & x_3 \leq 0 & & & - p_3 \geq -4 \\ & x_1 - 2x_2 \geq -1 & & & p_1 \geq 0 \\ & -3x_1 + x_2 \leq -2 & & & p_2 \leq 0 \\ & - 3x_2 - x_3 = -3, & & & p_3 \text{ free}. \end{array}$$

We observe that the latter problem is equivalent to the primal problem we started with. (The first three constraints in the latter problem are the same as the first three constraints in the original problem, multiplied by -1. Also, if the maximization in the latter problem is changed to a minimization, by multiplying the objective function by -1, we obtain the cost function in the original problem.)

The first primal problem considered in Example 4.1 had all of the ingredients of a general linear programming problem. This suggests that the conclusion reached at the end of the example should hold in general. Indeed, we have the following result. Its proof needs nothing more than the steps followed in Example 4.1, with abstract symbols replacing specific numbers, and will therefore be omitted.

Theorem 4.1 *If we transform the dual into an equivalent minimization problem and then form its dual, we obtain a problem equivalent to the original problem.*

A compact statement that is often used to describe Theorem 4.1 is that "the dual of the dual is the primal."

Any linear programming problem can be manipulated into one of several equivalent forms, for example, by introducing slack variables or by using the difference of two nonnegative variables to replace a single free variable. Each equivalent form leads to a somewhat different form for the dual problem. Nevertheless, the examples that follow indicate that the duals of equivalent problems are equivalent.

Example 4.2 Consider the primal problem shown on the left and its dual shown on the right:

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \text{ free,} \end{array} \qquad \begin{array}{ll} \text{maximize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p} \geq \mathbf{0} \\ & \mathbf{p}'\mathbf{A} = \mathbf{c}'. \end{array}$$

We transform the primal problem by introducing surplus variables and then obtain its dual:

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} + \mathbf{0}'\mathbf{s} \\ \text{subject to} & \mathbf{A}\mathbf{x} - \mathbf{s} = \mathbf{b} \\ & \mathbf{x} \text{ free} \\ & \mathbf{s} \geq \mathbf{0}, \end{array} \qquad \begin{array}{ll} \text{maximize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p} \text{ free} \\ & \mathbf{p}'\mathbf{A} = \mathbf{c}' \\ & -\mathbf{p} \leq \mathbf{0}. \end{array}$$

Alternatively, if we take the original primal problem and replace $\mathbf{x}$ by sign-constrained variables, we obtain the following pair of problems:

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x}^+ - \mathbf{c}'\mathbf{x}^- \\ \text{subject to} & \mathbf{A}\mathbf{x}^+ - \mathbf{A}\mathbf{x}^- \geq \mathbf{b} \\ & \mathbf{x}^+ \geq \mathbf{0} \\ & \mathbf{x}^- \geq \mathbf{0}, \end{array} \qquad \begin{array}{ll} \text{maximize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p} \geq \mathbf{0} \\ & \mathbf{p}'\mathbf{A} \leq \mathbf{c}' \\ & -\mathbf{p}'\mathbf{A} \leq -\mathbf{c}'. \end{array}$$

Note that we have three equivalent forms of the primal. We observe that the constraint $\mathbf{p} \geq \mathbf{0}$ is equivalent to the constraint $-\mathbf{p} \leq \mathbf{0}$. Furthermore, the constraint $\mathbf{p}'\mathbf{A} = \mathbf{c}'$ is equivalent to the two constraints $\mathbf{p}'\mathbf{A} \leq \mathbf{c}'$ and $-\mathbf{p}'\mathbf{A} \leq -\mathbf{c}'$. Thus, the duals of the three variants of the primal problem are also equivalent.

The next example is in the same spirit and examines the effect of removing redundant equality constraints in a standard form problem.

Example 4.3 Consider a standard form problem, assumed feasible, and its dual:

$$\begin{array}{ll} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \qquad \begin{array}{ll} \text{maximize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p}'\mathbf{A} \leq \mathbf{c}'. \end{array}$$

Let $\mathbf{a}_1', \ldots, \mathbf{a}_m'$ be the rows of $\mathbf{A}$ and suppose that $\mathbf{a}_m = \sum_{i=1}^{m-1} \gamma_i \mathbf{a}_i$ for some scalars $\gamma_1, \ldots, \gamma_{m-1}$. In particular, the last equality constraint is redundant and can be eliminated. By considering an arbitrary feasible solution $\mathbf{x}$, we obtain

$$b_m = \mathbf{a}_m'\mathbf{x} = \sum_{i=1}^{m-1} \gamma_i \mathbf{a}_i'\mathbf{x} = \sum_{i=1}^{m-1} \gamma_i b_i. \tag{4.1}$$

Note that the dual constraints are of the form $\sum_{i=1}^{m} p_i \mathbf{a}_i' \leq \mathbf{c}'$ and can be rewritten as

$$\sum_{i=1}^{m-1} (p_i + \gamma_i p_m)\mathbf{a}_i' \leq \mathbf{c}'.$$

Furthermore, using Eq. (4.1), the dual cost $\sum_{i=1}^{m} p_i b_i$ is equal to

$$\sum_{i=1}^{m-1} (p_i + \gamma_i p_m) b_i.$$

If we now let $q_i = p_i + \gamma_i p_m$, we see that the dual problem is equivalent to

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{m-1} q_i b_i \\ \text{subject to} \quad & \sum_{i=1}^{m-1} q_i \mathbf{a}_i' \leq \mathbf{c}'. \end{aligned}$$

We observe that this is the exact same dual that we would have obtained if we had eliminated the last (and redundant) constraint in the primal problem, before forming the dual.

The conclusions of the preceding two examples are summarized and generalized by the following result.

Theorem 4.2 *Suppose that we have transformed a linear programming problem Π_1 to another linear programming problem Π_2, by a sequence of transformations of the following types:*

(a) *Replace a free variable with the difference of two nonnegative variables.*

(b) *Replace an inequality constraint by an equality constraint involving a nonnegative slack variable.*

(c) *If some row of the matrix $\mathbf{A}$ in a feasible standard form problem is a linear combination of the other rows, eliminate the corresponding equality constraint.*

Then, the duals of Π_1 and Π_2 are equivalent, i.e., they are either both infeasible, or they have the same optimal cost.

The proof of Theorem 4.2 involves a combination of the various steps in Examples 4.2 and 4.3, and is left to the reader.

4.3 The duality theorem

We saw in Section 4.1 that for problems in standard form, the cost $g(\mathbf{p})$ of any dual solution provides a lower bound for the optimal cost. We now show that this property is true in general.

Theorem 4.3 (Weak duality) *If $\mathbf{x}$ is a feasible solution to the primal problem and $\mathbf{p}$ is a feasible solution to the dual problem, then*

$$\mathbf{p}'\mathbf{b} \leq \mathbf{c}'\mathbf{x}.$$

Proof. For any vectors $\mathbf{x}$ and $\mathbf{p}$, we define

$$\begin{aligned} u_i &= p_i(\mathbf{a}_i'\mathbf{x} - b_i), \\ v_j &= (c_j - \mathbf{p}'\mathbf{A}_j)x_j. \end{aligned}$$

Suppose that $\mathbf{x}$ and $\mathbf{p}$ are primal and dual feasible, respectively. The definition of the dual problem requires the sign of p_i to be the same as the sign of $\mathbf{a}_i'\mathbf{x} - b_i$, and the sign of $c_j - \mathbf{p}'\mathbf{A}_j$ to be the same as the sign of x_j. Thus, primal and dual feasibility imply that

$$u_i \geq 0, \qquad \forall\, i,$$

and

$$v_j \geq 0, \qquad \forall\, j.$$

Notice that

$$\sum_i u_i = \mathbf{p}'\mathbf{A}\mathbf{x} - \mathbf{p}'\mathbf{b},$$

and

$$\sum_j v_j = \mathbf{c}'\mathbf{x} - \mathbf{p}'\mathbf{A}\mathbf{x}.$$

We add these two equalities and use the nonnegativity of u_i, v_j, to obtain

$$0 \leq \sum_i u_i + \sum_j v_j = \mathbf{c}'\mathbf{x} - \mathbf{p}'\mathbf{b}. \qquad \square$$

The weak duality theorem is not a deep result, yet it does provide some useful information about the relation between the primal and the dual. We have, for example, the following corollary.

Corollary 4.1

(a) *If the optimal cost in the primal is $-\infty$, then the dual problem must be infeasible.*

(b) *If the optimal cost in the dual is $+\infty$, then the primal problem must be infeasible.*

Proof. Suppose that the optimal cost in the primal problem is $-\infty$ and that the dual problem has a feasible solution $\mathbf{p}$. By weak duality, $\mathbf{p}$ satisfies $\mathbf{p}'\mathbf{b} \leq \mathbf{c}'\mathbf{x}$ for every primal feasible $\mathbf{x}$. Taking the minimum over all primal feasible $\mathbf{x}$, we conclude that $\mathbf{p}'\mathbf{b} \leq -\infty$. This is impossible and shows that the dual cannot have a feasible solution, thus establishing part (a). Part (b) follows by a symmetrical argument. $\square$

Another important corollary of the weak duality theorem is the following.

Corollary 4.2 *Let* $\mathbf{x}$ *and* $\mathbf{p}$ *be feasible solutions to the primal and the dual, respectively, and suppose that* $\mathbf{p}'\mathbf{b} = \mathbf{c}'\mathbf{x}$. *Then,* $\mathbf{x}$ *and* $\mathbf{p}$ *are optimal solutions to the primal and the dual, respectively.*

Proof. Let $\mathbf{x}$ and $\mathbf{p}$ be as in the statement of the corollary. For every primal feasible solution $\mathbf{y}$, the weak duality theorem yields $\mathbf{c}'\mathbf{x} = \mathbf{p}'\mathbf{b} \leq \mathbf{c}'\mathbf{y}$, which proves that $\mathbf{x}$ is optimal. The proof of optimality of $\mathbf{p}$ is similar. □

The next theorem is the central result on linear programming duality.

Theorem 4.4 (Strong duality) *If a linear programming problem has an optimal solution, so does its dual, and the respective optimal costs are equal.*

Proof. Consider the standard form problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Let us assume temporarily that the rows of $\mathbf{A}$ are linearly independent and that there exists an optimal solution. Let us apply the simplex method to this problem. As long as cycling is avoided, e.g., by using the lexicographic pivoting rule, the simplex method terminates with an optimal solution $\mathbf{x}$ and an optimal basis $\mathbf{B}$. Let $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$ be the corresponding vector of basic variables. When the simplex method terminates, the reduced costs must be nonnegative and we obtain

$$\mathbf{c}' - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{0}',$$

where $\mathbf{c}_B'$ is the vector with the costs of the basic variables. Let us define a vector $\mathbf{p}$ by letting $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$. We then have $\mathbf{p}'\mathbf{A} \leq \mathbf{c}'$, which shows that $\mathbf{p}$ is a feasible solution to the dual problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} \leq \mathbf{c}'. \end{aligned}$$

In addition,

$$\mathbf{p}'\mathbf{b} = \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{b} = \mathbf{c}_B'\mathbf{x}_B = \mathbf{c}'\mathbf{x}.$$

It follows that $\mathbf{p}$ is an optimal solution to the dual (cf. Corollary 4.2), and the optimal dual cost is equal to the optimal primal cost.

If we are dealing with a general linear programming problem Π_1 that has an optimal solution, we first transform it into an equivalent standard

form problem Π_2, with the same optimal cost, and in which the rows of the matrix $\mathbf{A}$ are linearly independent. Let D_1 and D_2 be the duals of Π_1 and Π_2, respectively. By Theorem 4.2, the dual problems D_1 and D_2 have the same optimal cost. We have already proved that Π_2 and D_2 have the same optimal cost. It follows that Π_1 and D_1 have the same optimal cost (see Figure 4.1). □

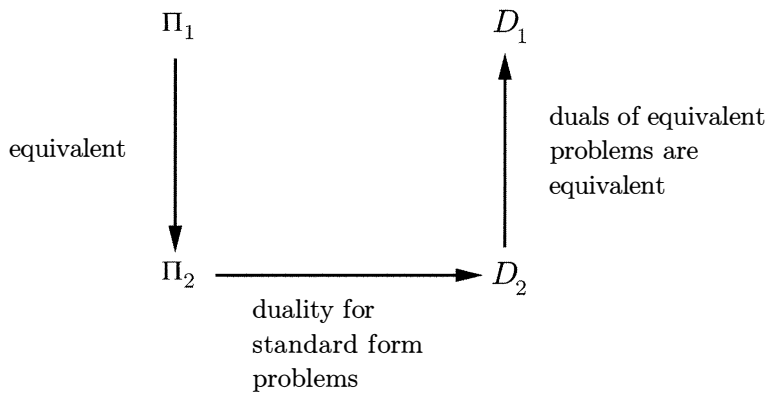


Figure 4.1: Proof of the duality theorem for general linear programming problems.

The preceding proof shows that an optimal solution to the dual problem is obtained as a byproduct of the simplex method as applied to a primal problem in standard form. It is based on the fact that the simplex method is guaranteed to terminate and this, in turn, depends on the existence of pivoting rules that prevent cycling. There is an alternative derivation of the duality theorem, which provides a geometric, algorithm-independent view of the subject, and which is developed in Section 4.7. At this point, we provide an illustration that conveys most of the content of the geometric proof.

Example 4.4 Consider a solid ball constrained to lie in a polyhedron defined by inequality constraints of the form $\mathbf{a}_i'\mathbf{x} \geq b_i$. If left under the influence of gravity, this ball reaches equilibrium at the lowest corner $\mathbf{x}^*$ of the polyhedron; see Figure 4.2. This corner is an optimal solution to the problem

$$\begin{aligned} &\text{minimize} && \mathbf{c}'\mathbf{x} \\ &\text{subject to} && \mathbf{a}_i'\mathbf{x} \geq b_i, \qquad \forall\, i, \end{aligned}$$

where $\mathbf{c}$ is a vertical vector pointing upwards. At equilibrium, gravity is counterbalanced by the forces exerted on the ball by the "walls" of the polyhedron. The latter forces are normal to the walls, that is, they are aligned with the vectors $\mathbf{a}_i$. We conclude that $\mathbf{c} = \sum_i p_i \mathbf{a}_i$, for some nonnegative coefficients p_i; in particular,

the vector $\mathbf{p}$ is a feasible solution to the dual problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} = \mathbf{c}' \\ & \mathbf{p} \geq \mathbf{0}. \end{aligned}$$

Given that forces can only be exerted by the walls that touch the ball, we must have $p_i = 0$, whenever $\mathbf{a}_i'\mathbf{x}^* > b_i$. Consequently, $p_i(b_i - \mathbf{a}_i'\mathbf{x}^*) = 0$ for all i. We therefore have $\mathbf{p}'\mathbf{b} = \sum_i p_i b_i = \sum_i p_i \mathbf{a}_i'\mathbf{x}^* = \mathbf{c}'\mathbf{x}^*$. It follows (Corollary 4.2) that $\mathbf{p}$ is an optimal solution to the dual, and the optimal dual cost is equal to the optimal primal cost.

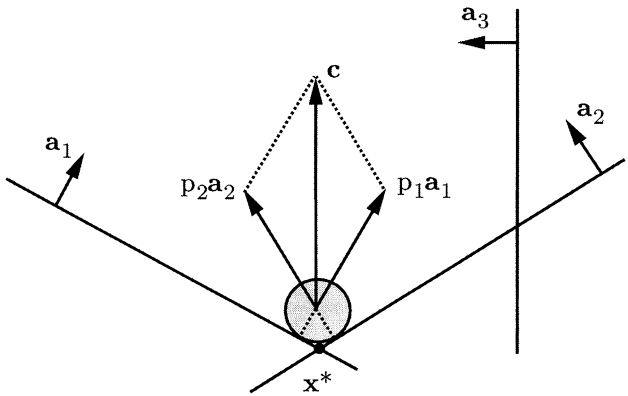


Figure 4.2: A mechanical analogy of the duality theorem.

Recall that in a linear programming problem, exactly one of the following three possibilities will occur:

(a) There is an optimal solution.

(b) The problem is "unbounded"; that is, the optimal cost is $-\infty$ (for minimization problems), or $+\infty$ (for maximization problems).

(c) The problem is infeasible.

This leads to nine possible combinations for the primal and the dual, which are shown in Table 4.2. By the strong duality theorem, if one problem has an optimal solution, so does the other. Furthermore, as discussed earlier, the weak duality theorem implies that if one problem is unbounded, the other must be infeasible. This allows us to mark some of the entries in Table 4.2 as "impossible."

	Finite optimum	**Unbounded**	**Infeasible**
Finite optimum	Possible	Impossible	Impossible
Unbounded	Impossible	Impossible	Possible
Infeasible	Impossible	Possible	Possible

Table 4.2: The different possibilities for the primal and the dual.

The case where both problems are infeasible can indeed occur, as shown by the following example.

Example 4.5 Consider the infeasible primal

$$\begin{array}{lrcl} \text{minimize} & x_1 + 2x_2 & & \\ \text{subject to} & x_1 + x_2 & = & 1 \\ & 2x_1 + 2x_2 & = & 3. \end{array}$$

Its dual is

$$\begin{array}{lrcl} \text{maximize} & p_1 + 3p_2 & & \\ \text{subject to} & p_1 + 2p_2 & = & 1 \\ & p_1 + 2p_2 & = & 2, \end{array}$$

which is also infeasible.

There is another interesting relation between the primal and the dual which is known as Clark's theorem (Clark, 1961). It asserts that unless both problems are infeasible, at least one of them must have an unbounded feasible set (Exercise 4.21).

Complementary slackness

An important relation between primal and dual optimal solutions is provided by the *complementary slackness* conditions, which we present next.

Theorem 4.5 (Complementary slackness) *Let* $\mathbf{x}$ *and* $\mathbf{p}$ *be feasible solutions to the primal and the dual problem, respectively. The vectors* $\mathbf{x}$ *and* $\mathbf{p}$ *are optimal solutions for the two respective problems if and only if:*

$$\begin{aligned} p_i(\mathbf{a}_i'\mathbf{x} - b_i) &= 0, \qquad \forall\, i, \\ (c_j - \mathbf{p}'\mathbf{A}_j)x_j &= 0, \qquad \forall\, j. \end{aligned}$$

Proof. In the proof of Theorem 4.3, we defined $u_i = p_i(\mathbf{a}_i'\mathbf{x} - b_i)$ and $v_j = (c_j - \mathbf{p}'\mathbf{A}_j)x_j$, and noted that for $\mathbf{x}$ primal feasible and $\mathbf{p}$ dual feasible,

we have $u_i \geq 0$ and $v_j \geq 0$ for all i and j. In addition, we showed that

$$\mathbf{c}'\mathbf{x} - \mathbf{p}'\mathbf{b} = \sum_i u_i + \sum_j v_j.$$

By the strong duality theorem, if $\mathbf{x}$ and $\mathbf{p}$ are optimal, then $\mathbf{c}'\mathbf{x} = \mathbf{p}'\mathbf{b}$, which implies that $u_i = v_j = 0$ for all i, j. Conversely, if $u_i = v_j = 0$ for all i, j, then $\mathbf{c}'\mathbf{x} = \mathbf{p}'\mathbf{b}$, and Corollary 4.2 implies that $\mathbf{x}$ and $\mathbf{p}$ are optimal. □

The first complementary slackness condition is automatically satisfied by every feasible solution to a problem in standard form. If the primal problem is not in standard form and has a constraint like $\mathbf{a}_i'\mathbf{x} \geq b_i$, the corresponding complementary slackness condition asserts that the dual variable p_i is zero unless the constraint is active. An intuitive explanation is that a constraint which is not active at an optimal solution can be removed from the problem without affecting the optimal cost, and there is no point in associating a nonzero price with such a constraint. Note also the analogy with Example 4.4, where "forces" were only exerted by the active constraints.

If the primal problem is in standard form and a nondegenerate optimal basic feasible solution is known, the complementary slackness conditions determine a unique solution to the dual problem. We illustrate this fact in the next example.

Example 4.6 Consider a problem in standard form and its dual:

$$\begin{array}{lrcl} \text{minimize} & 13x_1 + 10x_2 + 6x_3 & & \\ \text{subject to} & 5x_1 + x_2 + 3x_3 & = & 8 \\ & 3x_1 + x_2 & = & 3 \\ & x_1, x_2, x_3 \geq 0, & & \end{array} \qquad \begin{array}{lrcl} \text{maximize} & 8p_1 + 3p_2 & & \\ \text{subject to} & 5p_1 + 3p_2 & \leq & 13 \\ & p_1 + p_2 & \leq & 10 \\ & 3p_1 & \leq & 6. \end{array}$$

As will be verified shortly, the vector $\mathbf{x}^* = (1, 0, 1)$ is a nondegenerate optimal solution to the primal problem. Assuming this to be the case, we use the complementary slackness conditions to construct the optimal solution to the dual. The condition $p_i(\mathbf{a}_i'\mathbf{x}^* - b_i) = 0$ is automatically satisfied for each i, since the primal is in standard form. The condition $(c_j - \mathbf{p}'\mathbf{A}_j)x_j^* = 0$ is clearly satisfied for $j = 2$, because $x_2^* = 0$. However, since $x_1^* > 0$ and $x_3^* > 0$, we obtain

$$5p_1 + 3p_2 = 13,$$

and

$$3p_1 = 6,$$

which we can solve to obtain $p_1 = 2$ and $p_2 = 1$. Note that this is a dual feasible solution whose cost is equal to 19, which is the same as the cost of $\mathbf{x}^*$. This verifies that $\mathbf{x}^*$ is indeed an optimal solution as claimed earlier.

We now generalize the above example. Suppose that x_j is a basic variable in a nondegenerate optimal basic feasible solution to a primal

problem in standard form. Then, the complementary slackness condition $(c_j - \mathbf{p}'\mathbf{A}_j)x_j = 0$ yields $\mathbf{p}'\mathbf{A}_j = c_j$ for every such j. Since the basic columns $\mathbf{A}_j$ are linearly independent, we obtain a system of equations for $\mathbf{p}$ which has a unique solution, namely, $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$. A similar conclusion can also be drawn for problems not in standard form (Exercise 4.12). On the other hand, if we are given a degenerate optimal basic feasible solution to the primal, complementary slackness may be of very little help in determining an optimal solution to the dual problem (Exercise 4.17).

We finally mention that if the primal constraints are of the form $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, and the primal problem has an optimal solution, then there exist optimal solutions to the primal and the dual which satisfy *strict complementary slackness*; that is, a variable in one problem is nonzero if and only if the corresponding constraint in the other problem is active (Exercise 4.20). This result has some interesting applications in discrete optimization, but these lie outside the scope of this book.

A geometric view

We now develop a geometric view that allows us to visualize pairs of primal and dual vectors without having to draw the dual feasible set.

We consider the primal problem

$$\begin{aligned} &\text{minimize} && \mathbf{c}'\mathbf{x} \\ &\text{subject to} && \mathbf{a}_i'\mathbf{x} \geq b_i, \qquad i = 1, \ldots, m, \end{aligned}$$

where the dimension of $\mathbf{x}$ is equal to n. We assume that the vectors $\mathbf{a}_i$ span $\Re^n$. The corresponding dual problem is

$$\begin{aligned} &\text{maximize} && \mathbf{p}'\mathbf{b} \\ &\text{subject to} && \sum_{i=1}^{m} p_i\mathbf{a}_i = \mathbf{c} \\ & && \mathbf{p} \geq \mathbf{0}. \end{aligned}$$

Let I be a subset of $\{1, \ldots, m\}$ of cardinality n, such that the vectors $\mathbf{a}_i$, $i \in I$, are linearly independent. The system $\mathbf{a}_i'\mathbf{x} = b_i$, $i \in I$, has a unique solution, denoted by $\mathbf{x}^I$, which is a basic solution to the primal problem (cf. Definition 2.9 in Section 2.2). We assume, that $\mathbf{x}^I$ is nondegenerate, that is, $\mathbf{a}_i'\mathbf{x} \neq b_i$ for $i \notin I$.

Let $\mathbf{p} \in \Re^m$ be a dual vector (not necessarily dual feasible), and let us consider what is required for $\mathbf{x}^I$ and $\mathbf{p}$ to be optimal solutions to the primal and the dual problem, respectively. We need:

(a) $\mathbf{a}_i'\mathbf{x}^I \geq b_i$, for all i, (primal feasibility),

(b) $p_i = 0$, for all $i \notin I$, (complementary slackness),

(c) $\sum_{i=1}^{m} p_i\mathbf{a}_i = \mathbf{c}$, (dual feasibility),

(d) $\mathbf{p} \geq \mathbf{0}$, (dual feasibility).

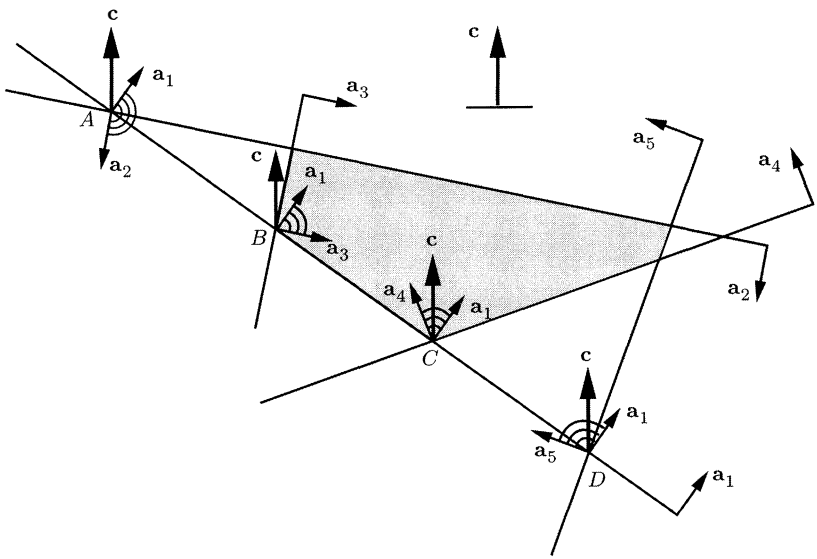


Figure 4.3: Consider a primal problem with two variables and five inequality constraints ($n = 2$, $m = 5$), and suppose that no two of the vectors $\mathbf{a}_i$ are collinear. Every two-element subset I of $\{1, 2, 3, 4, 5\}$ determines basic solutions $\mathbf{x}^I$ and $\mathbf{p}^I$ of the primal and the dual, respectively.
If $I = \{1, 2\}$, $\mathbf{x}^I$ is primal infeasible (point A) and $\mathbf{p}^I$ is dual infeasible, because $\mathbf{c}$ cannot be expressed as a nonnegative linear combination of the vectors $\mathbf{a}_1$ and $\mathbf{a}_2$.
If $I = \{1, 3\}$, $\mathbf{x}^I$ is primal feasible (point B) and $\mathbf{p}^I$ is dual infeasible.
If $I = \{1, 4\}$, $\mathbf{x}^I$ is primal feasible (point C) and $\mathbf{p}^I$ is dual feasible, because $\mathbf{c}$ can be expressed as a nonnegative linear combination of the vectors $\mathbf{a}_1$ and $\mathbf{a}_4$. In particular, $\mathbf{x}^I$ and $\mathbf{p}^I$ are optimal.
If $I = \{1, 5\}$, $\mathbf{x}^I$ is primal infeasible (point D) and $\mathbf{p}^I$ is dual feasible.

Given the complementary slackness condition (b), condition (c) becomes

$$\sum_{i \in I} p_i \mathbf{a}_i = \mathbf{c}.$$

Since the vectors $\mathbf{a}_i$, $i \in I$, are linearly independent, the latter equation has a unique solution that we denote by $\mathbf{p}^I$. In fact, it is readily seen that the vectors $\mathbf{a}_i$, $i \in I$, form a basis for the dual problem (which is in standard form) and $\mathbf{p}^I$ is the associated basic solution. For the vector $\mathbf{p}^I$ to be dual feasible, we also need it to be nonnegative. We conclude that once the complementary slackness condition (b) is enforced, feasibility of

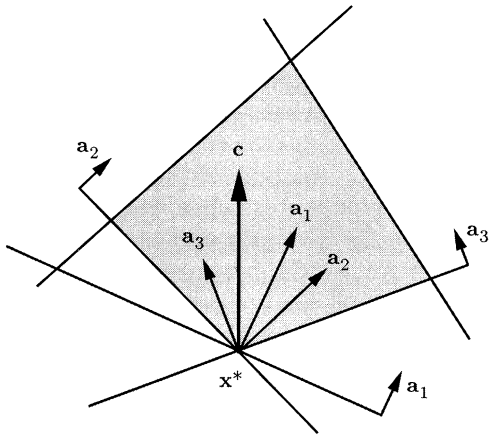


Figure 4.4: The vector $\mathbf{x}^*$ is a degenerate basic feasible solution of the primal. If we choose $I = \{1, 2\}$, the corresponding dual basic solution $\mathbf{p}^I$ is infeasible, because $\mathbf{c}$ is not a nonnegative linear combination of $\mathbf{a}_1$, $\mathbf{a}_2$. On the other hand, if we choose $I = \{1, 3\}$ or $I = \{2, 3\}$, the resulting dual basic solution $\mathbf{p}^I$ is feasible and, therefore, optimal.

the resulting dual vector $\mathbf{p}^I$ is equivalent to $\mathbf{c}$ being a nonnegative linear combination of the vectors $\mathbf{a}_i$, $i \in I$, associated with the active primal constraints. This allows us to visualize dual feasibility without having to draw the dual feasible set; see Figure 4.3.

If $\mathbf{x}^*$ is a degenerate basic solution to the primal, there can be several subsets I such that $\mathbf{x}^I = \mathbf{x}^*$. Using different choices for I, and by solving the system $\sum_{i \in I} p_i \mathbf{a}_i = \mathbf{c}$, we may obtain several dual basic solutions $\mathbf{p}^I$. It may then well be the case that some of them are dual feasible and some are not; see Figure 4.4. Still, if $\mathbf{p}^I$ is dual feasible (i.e., all p_i are nonnegative) and if $\mathbf{x}^*$ is primal feasible, then they are both optimal, because we have been enforcing complementary slackness and Theorem 4.5 applies.

4.4 Optimal dual variables as marginal costs

In this section, we elaborate on the interpretation of the dual variables as prices. This theme will be revisited, in more depth, in Chapter 5.

Consider the standard form problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

We assume that the rows of $\mathbf{A}$ are linearly independent and that there

is a nondegenerate basic feasible solution $\mathbf{x}^*$ which is optimal. Let $\mathbf{B}$ be the corresponding basis matrix and let $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$ be the vector of basic variables, which is positive, by nondegeneracy. Let us now replace $\mathbf{b}$ by $\mathbf{b} + \mathbf{d}$, where $\mathbf{d}$ is a small perturbation vector. Since $\mathbf{B}^{-1}\mathbf{b} > \mathbf{0}$, we also have $\mathbf{B}^{-1}(\mathbf{b} + \mathbf{d}) > \mathbf{0}$, as long as $\mathbf{d}$ is small. This implies that the same basis leads to a basic feasible solution of the perturbed problem as well. Perturbing the right-hand side vector $\mathbf{b}$ has no effect on the reduced costs associated with this basis. By the optimality of $\mathbf{x}^*$ in the original problem, the vector of reduced costs $\mathbf{c}' - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A}$ is nonnegative and this establishes that the same basis is optimal for the perturbed problem as well. Thus, the optimal cost in the perturbed problem is

$$\mathbf{c}_B'\mathbf{B}^{-1}(\mathbf{b} + \mathbf{d}) = \mathbf{p}'(\mathbf{b} + \mathbf{d}),$$

where $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$ is an optimal solution to the dual problem. Therefore, a small change of $\mathbf{d}$ in the right-hand side vector $\mathbf{b}$ results in a change of $\mathbf{p}'\mathbf{d}$ in the optimal cost. We conclude that each component p_i of the optimal dual vector can be interpreted as the *marginal cost* (or *shadow price*) per unit increase of the ith requirement b_i.

We conclude with yet another interpretation of duality, for standard form problems. In order to develop some concrete intuition, we phrase our discussion in terms of the diet problem (Example 1.3 in Section 1.1). We interpret each vector $\mathbf{A}_j$ as the nutritional content of the jth available food, and view $\mathbf{b}$ as the nutritional content of an ideal food that we wish to synthesize. Let us interpret p_i as the "fair" price per unit of the ith nutrient. A unit of the jth food has a value of c_j at the food market, but it also has a value of $\mathbf{p}'\mathbf{A}_j$ if priced at the nutrient market. Complementary slackness asserts that every food which is used (at a nonzero level) to synthesize the ideal food, should be consistently priced at the two markets. Thus, duality is concerned with two alternative ways of cost accounting. The value of the ideal food, as computed in the food market, is $\mathbf{c}'\mathbf{x}^*$, where $\mathbf{x}^*$ is an optimal solution to the primal problem; the value of the ideal food, as computed in the nutrient market, is $\mathbf{p}'\mathbf{b}$. The duality relation $\mathbf{c}'\mathbf{x}^* = \mathbf{p}'\mathbf{b}$ states that when prices are chosen appropriately, the two accounting methods should give the same results.

4.5 Standard form problems and the dual simplex method

In this section, we concentrate on the case where the primal problem is in standard form. We develop the *dual simplex method*, which is an alternative to the simplex method of Chapter 3. We also comment on the relation between the basic feasible solutions to the primal and the dual, including a discussion of dual degeneracy.

In the proof of the strong duality theorem, we considered the simplex method applied to a primal problem in standard form and defined a dual vector $\mathbf{p}$ by letting $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$. We then noted that the primal optimality condition $\mathbf{c}' - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{0}'$ is the same as the dual feasibility condition $\mathbf{p}'\mathbf{A} \leq \mathbf{c}'$. We can thus think of the simplex method as an algorithm that maintains primal feasibility and works towards dual feasibility. A method with this property is generally called a *primal* algorithm. An alternative is to start with a dual feasible solution and work towards primal feasibility. A method of this type is called a *dual* algorithm. In this section, we present a dual simplex method, implemented in terms of the full tableau. We argue that it does indeed solve the dual problem, and we show that it moves from one basic feasible solution of the dual problem to another. An alternative implementation that only keeps track of the matrix $\mathbf{B}^{-1}$, instead of the entire tableau, is called a *revised dual simplex* method (Exercise 4.23).

The dual simplex method

Let us consider a problem in standard form, under the usual assumption that the rows of the matrix $\mathbf{A}$ are linearly independent. Let $\mathbf{B}$ be a basis matrix, consisting of m linearly independent columns of $\mathbf{A}$, and consider the corresponding tableau

$-\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{b}$	$\bar{\mathbf{c}}'$
$\mathbf{B}^{-1}\mathbf{b}$	$\mathbf{B}^{-1}\mathbf{A}$

or, in more detail,

$-\mathbf{c}_B'\mathbf{x}_B$	$\bar{c}_1$	$\cdots$	$\bar{c}_n$
$x_{B(1)}$	|		|
$\vdots$	$\mathbf{B}^{-1}\mathbf{A}_1$	$\cdots$	$\mathbf{B}^{-1}\mathbf{A}_n$
$x_{B(m)}$	|		|

We do not require $\mathbf{B}^{-1}\mathbf{b}$ to be nonnegative, which means that we have a basic, but not necessarily feasible solution to the primal problem. However, we assume that $\bar{\mathbf{c}} \geq \mathbf{0}$; equivalently, the vector $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$ satisfies $\mathbf{p}'\mathbf{A} \leq \mathbf{c}'$, and we have a feasible solution to the dual problem. The cost of this dual feasible solution is $\mathbf{p}'\mathbf{b} = \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{b} = \mathbf{c}_B'\mathbf{x}_B$, which is the negative of the entry at the upper left corner of the tableau. If the inequality $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$ happens to hold, we also have a primal feasible solution with the same cost, and optimal solutions to both problems have been found. If the inequality $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$ fails to hold, we perform a change of basis in a manner we describe next.

We find some ℓ such that $x_{B(\ell)} < 0$ and consider the ℓth row of the tableau, called the *pivot row*; this row is of the form $(x_{B(\ell)}, v_1, \ldots, v_n)$, where v_i is the ℓth component of $\mathbf{B}^{-1}\mathbf{A}_i$. For each i with $v_i < 0$ (if such i exist), we form the ratio $\bar{c}_i/|v_i|$ and let j be an index for which this ratio is smallest; that is, $v_j < 0$ and

$$\frac{\bar{c}_j}{|v_j|} = \min_{\{i|v_i<0\}} \frac{\bar{c}_i}{|v_i|}. \tag{4.2}$$

(We call the corresponding entry v_j the *pivot element*. Note that x_j must be a nonbasic variable, since the jth column in the tableau contains the negative element v_j.) We then perform a change of basis: column $\mathbf{A}_j$ enters the basis and column $\mathbf{A}_{B(\ell)}$ exits. This change of basis (or *pivot*) is effected exactly as in the primal simplex method: we add to each row of the tableau a multiple of the pivot row so that all entries in the pivot column are set to zero, with the exception of the pivot element which is set to 1. In particular, in order to set the reduced cost in the pivot column to zero, we multiply the pivot row by $\bar{c}_j/|v_j|$ and add it to the zeroth row. For every i, the new value of $\bar{c}_i$ is equal to

$$\bar{c}_i + v_i \frac{\bar{c}_j}{|v_j|},$$

which is nonnegative because of the way that j was selected [cf. Eq. (4.2)]. We conclude that the reduced costs in the new tableau will also be nonnegative and dual feasibility has been maintained.

Example 4.7 Consider the tableau

		x_1	x_2	x_3	x_4	x_5
	0	2	6	10	0	0
$x_4 =$	2	−2	4	1	1	0
$x_5 =$	−1	4	−2*	−3	0	1

Since $x_{B(2)} < 0$, we choose the second row to be the pivot row. Negative entries of the pivot row are found in the second and third column. We compare the corresponding ratios $6/|-2|$ and $10/|-3|$. The smallest ratio is $6/|-2|$ and, therefore, the second column enters the basis. (The pivot element is indicated by an asterisk.) We multiply the pivot row by 3 and add it to the zeroth row. We multiply the pivot row by 2 and add it to the first row. We then divide the pivot row by −2. The new tableau is

		x_1	x_2	x_3	x_4	x_5
	−3	14	0	1	0	3
$x_4 =$	0	6	0	−5	1	2
$x_2 =$	1/2	−2	1	3/2	0	−1/2

The cost has increased to 3. Furthermore, we now have $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$, and an optimal solution has been found.

Note that the pivot element v_j is always chosen to be negative, whereas the corresponding reduced cost $\bar{c}_j$ is nonnegative. Let us temporarily assume that $\bar{c}_j$ is in fact positive. Then, in order to replace $\bar{c}_j$ by zero, we need to add a positive multiple of the pivot row to the zeroth row. Since $x_{B(\ell)}$ is negative, this has the effect of adding a negative quantity to the upper left corner. Equivalently, the dual cost increases. Thus, as long as the reduced cost of every nonbasic variable is nonzero, the dual cost increases with each basis change, and no basis will ever be repeated in the course of the algorithm. It follows that the algorithm must eventually terminate and this can happen in one of two ways:

(a) We have $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$ and an optimal solution.

(b) All of the entries $v_1, \ldots, v_n$ in the pivot row are nonnegative and we are therefore unable to locate a pivot element. In full analogy with the primal simplex method, this implies that the optimal dual cost is equal to $+\infty$ and the primal problem is infeasible; the proof is left as an exercise (Exercise 4.22).

We now provide a summary of the algorithm.

An iteration of the dual simplex method

1. A typical iteration starts with the tableau associated with a basis matrix $\mathbf{B}$ and with all reduced costs nonnegative.
2. Examine the components of the vector $\mathbf{B}^{-1}\mathbf{b}$ in the zeroth column of the tableau. If they are all nonnegative, we have an optimal basic feasible solution and the algorithm terminates; else, choose some ℓ such that $x_{B(\ell)} < 0$.
3. Consider the ℓth row of the tableau, with elements $x_{B(\ell)}, v_1, \ldots, v_n$ (the pivot row). If $v_i \geq 0$ for all i, then the optimal dual cost is $+\infty$ and the algorithm terminates.
4. For each i such that $v_i < 0$, compute the ratio $\bar{c}_i/|v_i|$ and let j be the index of a column that corresponds to the smallest ratio. The column $\mathbf{A}_{B(\ell)}$ exits the basis and the column $\mathbf{A}_j$ takes its place.
5. Add to each row of the tableau a multiple of the ℓth row (the pivot row) so that v_j (the pivot element) becomes 1 and all other entries of the pivot column become 0.

Let us now consider the possibility that the reduced cost $\bar{c}_j$ in the pivot column is zero. In this case, the zeroth row of the tableau does not change and the dual cost $\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{b}$ remains the same. The proof of termina-

tion given earlier does not apply and the algorithm can cycle. This can be avoided by employing a suitable anticycling rule, such as the following.

Lexicographic pivoting rule for the dual simplex method

1. Choose any row ℓ such that $x_{B(\ell)} < 0$, to be the pivot row.
2. Determine the index j of the entering column as follows. For each column with $v_i < 0$, divide all entries by $|v_i|$, and then choose the lexicographically smallest column. If there is a tie between several lexicographically smallest columns, choose the one with the smallest index.

If the dual simplex method is initialized so that every column of the tableau [that is, each vector $(\bar{c}_j, \mathbf{B}^{-1}\mathbf{A}_j)$] is lexicographically positive, and if the above lexicographic pivoting rule is used, the method terminates in a finite number of steps. The proof is similar to the proof of the corresponding result for the primal simplex method (Theorem 3.4) and is left as an exercise (Exercise 4.24).

When should we use the dual simplex method

At this point, it is natural to ask when the dual simplex method should be used. One such case arises when a basic feasible solution of the dual problem is readily available. Suppose, for example, that we already have an optimal basis for some linear programming problem, and that we wish to solve the same problem for a different choice of the right-hand side vector $\mathbf{b}$. The optimal basis for the original problem may be primal infeasible under the new value of $\mathbf{b}$. On the other hand, a change in $\mathbf{b}$ does not affect the reduced costs and we still have a dual feasible solution. Thus, instead of solving the new problem from scratch, it may be preferable to apply the dual simplex algorithm starting from the optimal basis for the original problem. This idea will be considered in more detail in Chapter 5.

The geometry of the dual simplex method

Our development of the dual simplex method was based entirely on tableau manipulations and algebraic arguments. We now present an alternative viewpoint based on geometric considerations.

We continue assuming that we are dealing with a problem in standard form and that the matrix $\mathbf{A}$ has linearly independent rows. Let $\mathbf{B}$ be a basis matrix with columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$. This basis matrix determines a basic solution to the primal problem with $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}$. The same basis can also be used to determine a dual vector $\mathbf{p}$ by means of the equations

$$\mathbf{p}'\mathbf{A}_{B(i)} = c_{B(i)}, \qquad i = 1, \ldots, m.$$

These are m equations in m unknowns; since the columns $\mathbf{A}_{B(1)}, \ldots, \mathbf{A}_{B(m)}$ are linearly independent, there is a unique solution $\mathbf{p}$. For such a vector $\mathbf{p}$, the number of linearly independent active dual constraints is equal to the dimension of the dual vector, and it follows that we have a basic solution to the dual problem. In matrix notation, the dual basic solution $\mathbf{p}$ satisfies $\mathbf{p}'\mathbf{B} = \mathbf{c}'_B$, or $\mathbf{p}' = \mathbf{c}'_B\mathbf{B}^{-1}$, which was referred to as the vector of simplex multipliers in Chapter 3. If $\mathbf{p}$ is also dual feasible, that is, if $\mathbf{p}'\mathbf{A} \leq \mathbf{c}'$, then $\mathbf{p}$ is a basic feasible solution of the dual problem.

To summarize, a basis matrix $\mathbf{B}$ is associated with a basic solution to the primal problem and also with a basic solution to the dual. A basic solution to the primal (respectively, dual) which is primal (respectively, dual) feasible, is a basic feasible solution to the primal (respectively, dual).

We now have a geometric interpretation of the dual simplex method: at every iteration, we have a basic feasible solution to the dual problem. The basic feasible solutions obtained at any two consecutive iterations have $m-1$ linearly independent active constraints in common (the reduced costs of the $m-1$ variables that are common to both bases are zero); thus, consecutive basic feasible solutions are either adjacent or they coincide.

Example 4.8 Consider the following standard form problem and its dual:

$$\begin{array}{lll} \text{minimize} & x_1 + x_2 & \\ \text{subject to} & x_1 + 2x_2 - x_3 = 2 & \\ & x_1 - x_4 = 1 & \\ & x_1, x_2, x_3, x_4 \geq 0, & \end{array} \qquad \begin{array}{ll} \text{maximize} & 2p_1 + p_2 \\ \text{subject to} & p_1 + p_2 \leq 1 \\ & 2p_1 \leq 1 \\ & p_1, p_2 \geq 0. \end{array}$$

The feasible set of the primal problem is 4-dimensional. If we eliminate the variables x_3 and x_4, we obtain the equivalent problem

$$\begin{array}{ll} \text{minimize} & x_1 + x_2 \\ \text{subject to} & x_1 + 2x_2 \geq 2 \\ & x_1 \geq 1 \\ & x_1, x_2 \geq 0. \end{array}$$

The feasible sets of the equivalent primal problem and of the dual are shown in Figures 4.5(a) and 4.5(b), respectively.

There is a total of five different bases in the standard form primal problem and five different basic solutions. These correspond to the points A, B, C, D, and E in Figure 4.5(a). The same five bases also lead to five basic solutions to the dual problem, which are points A, B, C, D, and E in Figure 4.5(b).

For example, if we choose the columns $\mathbf{A}_3$ and $\mathbf{A}_4$ to be the basic columns, we have the infeasible primal basic solution $\mathbf{x} = (0, 0, -2, -1)$ (point A). The corresponding dual basic solution is obtained by letting $\mathbf{p}'\mathbf{A}_3 = c_3 = 0$ and $\mathbf{p}'\mathbf{A}_4 = c_4 = 0$, which yields $\mathbf{p} = (0, 0)$. This is a basic feasible solution of the dual problem and can be used to start the dual simplex method. The associated initial tableau is

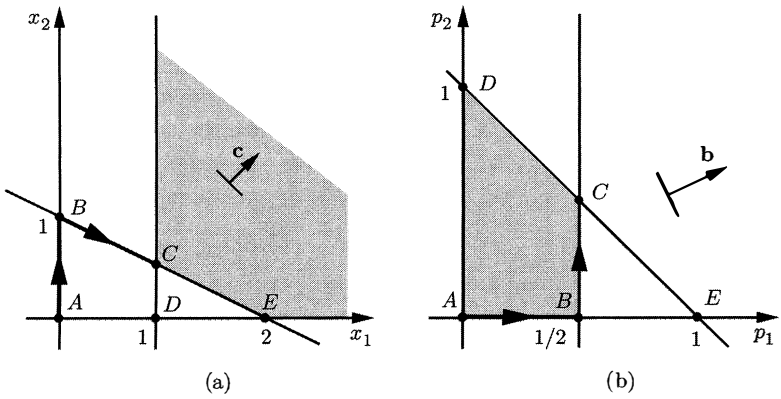


Figure 4.5: The feasible sets in Example 4.8.

		x_1	x_2	x_3	x_4
	0	1	1	0	0
$x_3 =$	−2	−1	−2*	1	0
$x_4 =$	−1	−1	0	0	1

We carry out two iterations of the dual simplex method to obtain the following two tableaux:

		x_1	x_2	x_3	x_4
	−1	1/2	0	1/2	0
$x_2 =$	1	1/2	1	−1/2	0
$x_4 =$	−1	−1*	0	0	1

		x_1	x_2	x_3	x_4
	−3/2	0	0	1/2	1/2
$x_2 =$	1/2	0	1	−1/2	1/2
$x_1 =$	1	1	0	0	−1

This sequence of tableaux corresponds to the path $A - B - C$ in either figure. In the primal space, the path traces a sequence of infeasible basic solutions until, at

optimality, it becomes feasible. In the dual space, the algorithm behaves exactly like the primal simplex method: it moves through a sequence of (dual) basic feasible solutions, while at each step improving the cost function.

Having observed that the dual simplex method moves from one basic feasible solution of the dual to an adjacent one, it may be tempting to say that the dual simplex method is simply the primal simplex method applied to the dual. This is a somewhat ambiguous statement, however, because the dual problem is not in standard form. If we were to convert it to standard form and then apply the primal simplex method, the resulting method is not necessarily identical to the dual simplex method (Exercise 4.25). A more accurate statement is to simply say that the dual simplex method is a variant of the simplex method tailored to problems defined exclusively in terms of linear inequality constraints.

Duality and degeneracy

Let us keep assuming that we are dealing with a standard form problem in which the rows of the matrix $\mathbf{A}$ are linearly independent. Any basis matrix $\mathbf{B}$ leads to an associated dual basic solution given by $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$. At this basic solution, the dual constraint $\mathbf{p}'\mathbf{A}_j = c_j$ is active if and only if $\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A}_j = c_j$, that is, if and only if the reduced cost $\bar{c}_j$ is zero. Since $\mathbf{p}$ is m-dimensional, dual degeneracy amounts to having more than m reduced costs that are zero. Given that the reduced costs of the m basic variables must be zero, dual degeneracy is obtained whenever there exists a nonbasic variable whose reduced cost is zero.

The example that follows deals with the relation between basic solutions to the primal and the dual in the face of degeneracy.

Example 4.9 Consider the following standard from problem and its dual:

$$\begin{array}{lll} \text{minimize} & 3x_1 + x_2 & \\ \text{subject to} & x_1 + x_2 - x_3 = 2 & \\ & 2x_1 - x_2 - x_4 = 0 & \\ & x_1, x_2, x_3, x_4 \geq 0, & \end{array} \qquad \begin{array}{lll} \text{maximize} & 2p_1 & \\ \text{subject to} & p_1 + 2p_2 \leq 3 & \\ & p_1 - p_2 \leq 1 & \\ & p_1, p_2 \geq 0. & \end{array}$$

We eliminate x_3 and x_4 to obtain the equivalent primal problem

$$\begin{array}{ll} \text{minimize} & 3x_1 + x_2 \\ \text{subject to} & x_1 + x_2 \geq 2 \\ & 2x_1 - x_2 \geq 0 \\ & x_1, x_2 \geq 0. \end{array}$$

The feasible set of the equivalent primal and of the dual is shown in Figures 4.6(a) and 4.6(b), respectively.

There is a total of six different bases in the standard form primal problem, but only four different basic solutions [points A, B, C, D in Figure 4.6(a)]. In the dual problem, however, the six bases lead to six distinct basic solutions [points A, A', A'', B, C, D in Figure 4.6(b)].

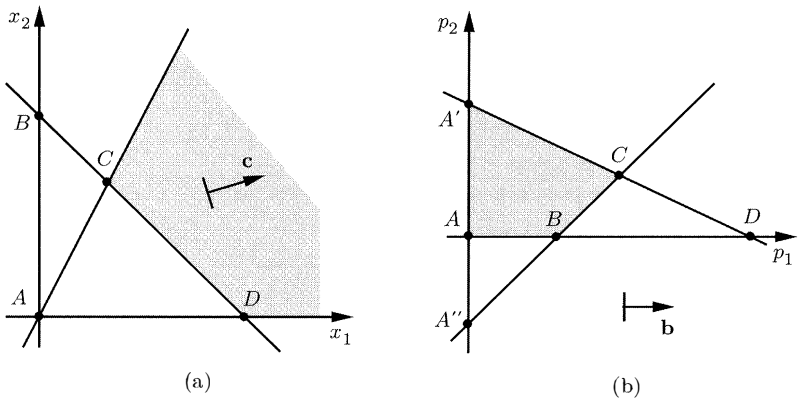


Figure 4.6: The feasible sets in Example 4.9.

For example, if we let columns $\mathbf{A}_3$ and $\mathbf{A}_4$ be basic, the primal basic solution has $x_1 = x_2 = 0$ and the corresponding dual basic solution is $(p_1, p_2) = (0, 0)$. Note that this is a basic feasible solution of the dual problem. If we let columns $\mathbf{A}_1$ and $\mathbf{A}_3$ be basic, the primal basic solution has again $x_1 = x_2 = 0$. For the dual problem, however, the equations $\mathbf{p}'\mathbf{A}_1 = c_1$ and $\mathbf{p}'\mathbf{A}_3 = c_3$ yield $(p_1, p_2) = (0, 3/2)$, which is a basic feasible solution of the dual, namely, point A' in Figure 4.6(b). Finally, if we let columns $\mathbf{A}_2$ and $\mathbf{A}_3$ be basic, we still have the same primal solution. For the dual problem, the equations $\mathbf{p}'\mathbf{A}_2 = c_1$ and $\mathbf{p}'\mathbf{A}_3 = c_3$ yield $(p_1, p_2) = (0, -1)$, which is an infeasible basic solution to the dual, namely, point A'' in Figure 4.6(b).

Example 4.9 has established that different bases may lead to the same basic solution for the primal problem, but to different basic solutions for the dual. Furthermore, out of the different basic solutions to the dual problem, it may be that some are feasible and some are infeasible.

We conclude with a summary of some properties of bases and basic solutions, for standard form problems, that were discussed in this section.

(a) Every basis determines a basic solution to the primal, but also a corresponding basic solution to the dual, namely, $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$.

(b) This dual basic solution is feasible if and only if all of the reduced costs are nonnegative.

(c) Under this dual basic solution, the reduced costs that are equal to zero correspond to active constraints in the dual problem.

(d) This dual basic solution is degenerate if and only if some nonbasic variable has zero reduced cost.

4.6 Farkas' lemma and linear inequalities

Suppose that we wish to determine whether a given system of linear inequalities is infeasible. In this section, we approach this question using duality theory, and we show that infeasibility of a given system of linear inequalities is equivalent to the feasibility of another, related, system of linear inequalities. Intuitively, the latter system of linear inequalities can be interpreted as a search for a *certificate of infeasibility* for the former system.

To be more specific, consider a set of standard form constraints $\mathbf{Ax} = \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$. Suppose that there exists some vector $\mathbf{p}$ such that $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$ and $\mathbf{p}'\mathbf{b} < 0$. Then, for any $\mathbf{x} \geq \mathbf{0}$, we have $\mathbf{p}'\mathbf{Ax} \geq \mathbf{0}$ and since $\mathbf{p}'\mathbf{b} < 0$, it follows that $\mathbf{p}'\mathbf{Ax} \neq \mathbf{p}'\mathbf{b}$. We conclude that $\mathbf{Ax} \neq \mathbf{b}$, for all $\mathbf{x} \geq \mathbf{0}$. This argument shows that if we can find a vector $\mathbf{p}$ satisfying $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$ and $\mathbf{p}'\mathbf{b} < 0$, the standard form constraints cannot have any feasible solution, and such a vector $\mathbf{p}$ is a certificate of infeasibility. Farkas' lemma below states that whenever a standard form problem is infeasible, such a certificate of infeasibility $\mathbf{p}$ is guaranteed to exist.

Theorem 4.6 (Farkas' lemma) *Let $\mathbf{A}$ be a matrix of dimensions $m \times n$ and let $\mathbf{b}$ be a vector in $\Re^m$. Then, exactly one of the following two alternatives holds:*

(a) *There exists some $\mathbf{x} \geq \mathbf{0}$ such that $\mathbf{Ax} = \mathbf{b}$.*

(b) *There exists some vector $\mathbf{p}$ such that $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$ and $\mathbf{p}'\mathbf{b} < 0$.*

Proof. One direction is easy. If there exists some $\mathbf{x} \geq \mathbf{0}$ satisfying $\mathbf{Ax} = \mathbf{b}$, and if $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$, then $\mathbf{p}'\mathbf{b} = \mathbf{p}'\mathbf{Ax} \geq 0$, which shows that the second alternative cannot hold.

Let us now assume that there exists no vector $\mathbf{x} \geq \mathbf{0}$ satisfying $\mathbf{Ax} = \mathbf{b}$. Consider the pair of problems

$$\begin{array}{lll} \text{maximize} & \mathbf{0}'\mathbf{x} & \\ \text{subject to} & \mathbf{Ax} = \mathbf{b} & \\ & \mathbf{x} \geq \mathbf{0}, & \end{array} \qquad \begin{array}{ll} \text{minimize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p}'\mathbf{A} \geq \mathbf{0}', \end{array}$$

and note that the first is the dual of the second. The maximization problem is infeasible, which implies that the minimization problem is either unbounded (the optimal cost is $-\infty$) or infeasible. Since $\mathbf{p} = \mathbf{0}$ is a feasible solution to the minimization problem, it follows that the minimization problem is unbounded. Therefore, there exists some $\mathbf{p}$ which is feasible, that is, $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$, and whose cost is negative, that is, $\mathbf{p}'\mathbf{b} < 0$. □

We now provide a geometric illustration of Farkas' lemma (see Figure 4.7). Let $\mathbf{A}_1, \ldots, \mathbf{A}_n$ be the columns of the matrix $\mathbf{A}$ and note that $\mathbf{Ax} = \sum_{i=1}^{n} \mathbf{A}_i x_i$. Therefore, the existence of a vector $\mathbf{x} \geq \mathbf{0}$ satisfying

$\mathbf{Ax} = \mathbf{b}$ is the same as requiring that $\mathbf{b}$ lies in the set of all nonnegative linear combinations of the vectors $\mathbf{A}_1, \ldots, \mathbf{A}_n$, which is the shaded region in Figure 4.7. If $\mathbf{b}$ does not belong to the shaded region (in which case the first alternative in Farkas' lemma does not hold), we expect intuitively that we can find a vector $\mathbf{p}$ and an associated hyperplane $\{\mathbf{z} \mid \mathbf{p}'\mathbf{z} = 0\}$ such that $\mathbf{b}$ lies on one side of the hyperplane while the shaded region lies on the other side. We then have $\mathbf{p}'\mathbf{b} < 0$ and $\mathbf{p}'\mathbf{A}_i \geq 0$ for all i, or, equivalently, $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$, and the second alternative holds.

Farkas' lemma predates the development of linear programming, but duality theory leads to a simple proof. A different proof, based on the geometric argument we just gave, is provided in the next section. Finally, there is an equivalent statement of Farkas' lemma which is sometimes more convenient.

Corollary 4.3 *Let $\mathbf{A}_1, \ldots, \mathbf{A}_n$ and $\mathbf{b}$ be given vectors and suppose that any vector $\mathbf{p}$ that satisfies $\mathbf{p}'\mathbf{A}_i \geq 0$, $i = 1, \ldots, n$, must also satisfy $\mathbf{p}'\mathbf{b} \geq 0$. Then, $\mathbf{b}$ can be expressed as a nonnegative linear combination of the vectors $\mathbf{A}_1, \ldots, \mathbf{A}_n$.*

Our next result is of a similar character.

Theorem 4.7 *Suppose that the system of linear inequalities $\mathbf{Ax} \leq \mathbf{b}$ has at least one solution, and let d be some scalar. Then, the following are equivalent:*

(a) *Every feasible solution to the system $\mathbf{Ax} \leq \mathbf{b}$ satisfies $\mathbf{c}'\mathbf{x} \leq d$.*

(b) *There exists some $\mathbf{p} \geq \mathbf{0}$ such that $\mathbf{p}'\mathbf{A} = \mathbf{c}'$ and $\mathbf{p}'\mathbf{b} \leq d$.*

Proof. Consider the following pair of problems

$$\begin{array}{lll} \text{maximize} & \mathbf{c}'\mathbf{x} & \\ \text{subject to} & \mathbf{Ax} \leq \mathbf{b}, & \end{array} \qquad \begin{array}{lll} \text{minimize} & \mathbf{p}'\mathbf{b} & \\ \text{subject to} & \mathbf{p}'\mathbf{A} = \mathbf{c}' & \\ & \mathbf{p} \geq \mathbf{0}, & \end{array}$$

and note that the first is the dual of the second. If the system $\mathbf{Ax} \leq \mathbf{b}$ has a feasible solution and if every feasible solution satisfies $\mathbf{c}'\mathbf{x} \leq d$, then the first problem has an optimal solution and the optimal cost is bounded above by d. By the strong duality theorem, the second problem also has an optimal solution $\mathbf{p}$ whose cost is bounded above by d. This optimal solution satisfies $\mathbf{p}'\mathbf{A} = \mathbf{c}'$, $\mathbf{p} \geq \mathbf{0}$, and $\mathbf{p}'\mathbf{b} \leq d$.

Conversely, if some $\mathbf{p}$ satisfies $\mathbf{p}'\mathbf{A} = \mathbf{c}'$, $\mathbf{p} \geq \mathbf{0}$, and $\mathbf{p}'\mathbf{b} \leq d$, then the weak duality theorem asserts that every feasible solution to the first problem must also satisfy $\mathbf{c}'\mathbf{x} \leq d$. □

Results such as Theorems 4.6 and 4.7 are often called *theorems of the*

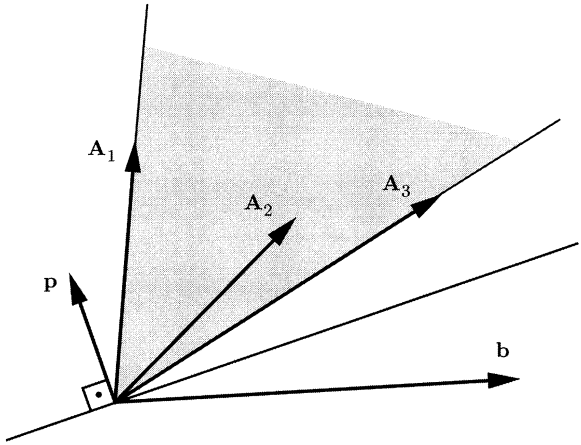


Figure 4.7: If the vector **b** does not belong to the set of all nonnegative linear combinations of $\mathbf{A}_1, \ldots, \mathbf{A}_n$, then we can find a hyperplane $\{\mathbf{z} \mid \mathbf{p}'\mathbf{z} = 0\}$ that separates it from that set.

alternative. There are several more results of this type; see, for example, Exercises 4.26, 4.27, and 4.28.

Applications of Farkas' lemma to asset pricing

Consider a market that operates for a single period, and in which n different assets are traded. Depending on the events during that single period, there are m possible states of nature at the end of the period. If we invest one dollar in some asset i and the state of nature turns out to be s, we receive a payoff of r_{si}. Thus, each asset i is described by a payoff vector $(r_{1i}, \ldots, r_{mi})$. The following $m \times n$ payoff matrix gives the payoffs of each of the n assets for each of the m states of nature:

$$\mathbf{R} = \begin{bmatrix} r_{11} & \cdots & r_{1n} \\ \vdots & \ddots & \vdots \\ r_{m1} & \cdots & r_{mn} \end{bmatrix}.$$

Let x_i be the amount held of asset i. A portfolio of assets is then a vector $\mathbf{x} = (x_1, \ldots, x_n)$. The components of a portfolio $\mathbf{x}$ can be either positive or negative. A positive value of x_i indicates that one has bought x_i units of asset i and is thus entitled to receive $r_{si}x_i$ if state s materializes. A negative value of x_i indicates a "short" position in asset i: this amounts to selling $|x_i|$ units of asset i at the beginning of the period, with a promise to buy them back at the end. Hence, one must pay out $r_{si}|x_i|$ if state s occurs, which is the same as receiving a payoff of $r_{si}x_i$.

The wealth in state s that results from a portfolio $\mathbf{x}$ is given by

$$w_s = \sum_{i=1}^{n} r_{si} x_i.$$

We introduce the vector $\mathbf{w} = (w_1, \ldots, w_m)$, and we obtain

$$\mathbf{w} = \mathbf{R}\mathbf{x}.$$

Let p_i be the price of asset i in the beginning of the period, and let $\mathbf{p} = (p_1, \ldots, p_n)$ be the vector of asset prices. Then, the cost of acquiring a portfolio $\mathbf{x}$ is given by $\mathbf{p}'\mathbf{x}$.

The central problem in asset pricing is to determine what the prices p_i should be. In order to address this question, we introduce the *absence of arbitrage* condition, which underlies much of finance theory: asset prices should always be such that no investor can get a guaranteed nonnegative payoff out of a negative investment. In other words, any portfolio that pays off nonnegative amounts in every state of nature, must be valuable to investors, so it must have nonnegative cost. Mathematically, the absence of arbitrage condition can be expressed as follows:

$$\text{if } \mathbf{R}\mathbf{x} \geq \mathbf{0}, \text{ then we must have } \mathbf{p}'\mathbf{x} \geq 0.$$

Given a particular set of assets, as described by the payoff matrix $\mathbf{R}$, only certain prices $\mathbf{p}$ are consistent with the absence of arbitrage. What characterizes such prices? What restrictions does the assumption of no arbitrage impose on asset prices? The answer is provided by Farkas' lemma.

Theorem 4.8 *The absence of arbitrage condition holds if and only if there exists a nonnegative vector* $\mathbf{q} = (q_1, \ldots, q_m)$, *such that the price of each asset* i *is given by*

$$p_i = \sum_{s=1}^{m} q_s r_{si}.$$

Proof. The absence of arbitrage condition states that there exists no vector $\mathbf{x}$ such that $\mathbf{x}'\mathbf{R}' \geq \mathbf{0}'$ and $\mathbf{x}'\mathbf{p} < 0$. This is of the same form as condition (b) in the statement of Farkas' lemma (Theorem 4.6). (Note that here $\mathbf{p}$ plays the role of $\mathbf{b}$, and $\mathbf{R}'$ plays the role of $\mathbf{A}$.) Therefore, by Farkas' lemma, the absence of arbitrage condition holds if and only if there exists some nonnegative vector $\mathbf{q}$ such that $\mathbf{R}'\mathbf{q} = \mathbf{p}$, which is the same as the condition in the theorem's statement. □

Theorem 4.8 asserts that whenever the market works efficiently enough to eliminate the possibility of arbitrage, there must exist "state prices" q_s

that can be used to value the existing assets. Intuitively, it establishes a nonnegative price q_s for an elementary asset that pays one dollar if the state of nature is s, and nothing otherwise. It then requires that every asset must be consistently priced, its total value being the sum of the values of the elementary assets from which it is composed. There is an alternative interpretation of the variables q_s as being (unnormalized) probabilities of the different states s, which, however, we will not pursue. In general, the state price vector $\mathbf{q}$ will not be unique, unless the number of assets equals or exceeds the number of states.

The no arbitrage condition is very simple, and yet very powerful. It is the key element behind many important results in financial economics, but these lie beyond the scope of this text. (See, however, Exercise 4.33 for an application in options pricing.)

4.7 From separating hyperplanes to duality*

Let us review the path followed in our development of duality theory. We started from the fact that the simplex method, in conjunction with an anticycling rule, is guaranteed to terminate. We then exploited the termination conditions of the simplex method to derive the strong duality theorem. We finally used the duality theorem to derive Farkas' lemma, which we interpreted in terms of a hyperplane that separates $\mathbf{b}$ from the columns of $\mathbf{A}$. In this section, we show that the reverse line of argument is also possible. We start from first principles and prove a general result on separating hyperplanes. We then establish Farkas' lemma, and conclude by showing that the duality theorem follows from Farkas' lemma. This line of argument is more elegant and fundamental because instead of relying on the rather complicated development of the simplex method, it only involves a small number of basic geometric concepts. Furthermore, it can be naturally generalized to nonlinear optimization problems.

Closed sets and Weierstrass' theorem

Before we proceed any further, we need to develop some background material. A set $S \subset \Re^n$ is called *closed* if it has the following property: if $\mathbf{x}^1, \mathbf{x}^2, \ldots$ is a sequence of elements of S that converges to some $\mathbf{x} \in \Re^n$, then $\mathbf{x} \in S$. In other words, S contains the limit of any sequence of elements of S. Intuitively, the set S contains its boundary.

Theorem 4.9 *Every polyhedron is closed.*

Proof. Consider the polyhedron $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$. Suppose that $\mathbf{x}^1, \mathbf{x}^2, \ldots$ is a sequence of elements of P that converges to some $\mathbf{x}^*$. We have

to show that $\mathbf{x}^* \in P$. For each k, we have $\mathbf{x}^k \in P$ and, therefore, $\mathbf{A}\mathbf{x}^k \geq \mathbf{b}$. Taking the limit, we obtain $\mathbf{A}\mathbf{x}^* = \mathbf{A}\big(\lim_{k\to\infty} \mathbf{x}^k\big) = \lim_{k\to\infty} \big(\mathbf{A}\mathbf{x}^k\big) \geq \mathbf{b}$, and $\mathbf{x}^*$ belongs to P. □

The following is a fundamental result from real analysis that provides us with conditions for the existence of an optimal solution to an optimization problem. The proof lies beyond the scope of this book and is omitted.

Theorem 4.10 (Weierstrass' theorem) *If $f : \Re^n \mapsto \Re$ is a continuous function, and if S is a nonempty, closed, and bounded subset of $\Re^n$, then there exists some $\mathbf{x}^* \in S$ such that $f(\mathbf{x}^*) \leq f(\mathbf{x})$ for all $\mathbf{x} \in S$. Similarly, there exists some $\mathbf{y}^* \in S$ such that $f(\mathbf{y}^*) \geq f(\mathbf{x})$ for all $\mathbf{x} \in S$.*

Weierstrass' theorem is not valid if the set S is not closed. Consider, for example, the set $S = \{x \in \Re \mid x > 0\}$. This set is not closed because we can form a sequence of elements of S that converge to zero, but $x = 0$ does not belong to S. We then observe that the cost function $f(x) = x$ is not minimized at any point in S; for every $x > 0$, there exists another positive number with smaller cost, and no feasible x can be optimal. Ultimately, the reason that S is not closed is that the feasible set was defined by means of *strict* inequalities. The definition of polyhedra and linear programming problems does not allow for strict inequalities in order to avoid situations of this type.

The separating hyperplane theorem

The result that follows is "geometrically obvious" but nevertheless extremely important in the study of convex sets and functions. It states that if we are given a closed and nonempty convex set S and a point $\mathbf{x}^* \notin S$, then we can find a hyperplane, called a *separating hyperplane*, such that S and $\mathbf{x}^*$ lie in different halfspaces (Figure 4.8).

Theorem 4.11 (Separating hyperplane theorem) *Let S be a nonempty closed convex subset of $\Re^n$ and let $\mathbf{x}^* \in \Re^n$ be a vector that does not belong to S. Then, there exists some vector $\mathbf{c} \in \Re^n$ such that $\mathbf{c}'\mathbf{x}^* < \mathbf{c}'\mathbf{x}$ for all $\mathbf{x} \in S$.*

Proof. Let $\|\cdot\|$ be the Euclidean norm defined by $\|\mathbf{x}\| = (\mathbf{x}'\mathbf{x})^{1/2}$. Let us fix some element $\mathbf{w}$ of S, and let

$$B = \big\{\mathbf{x} \mid \|\mathbf{x} - \mathbf{x}^*\| \leq \|\mathbf{w} - \mathbf{x}^*\|\big\},$$

and $D = S \cap B$ [Figure 4.9(a)]. The set D is nonempty, because $\mathbf{w} \in D$.

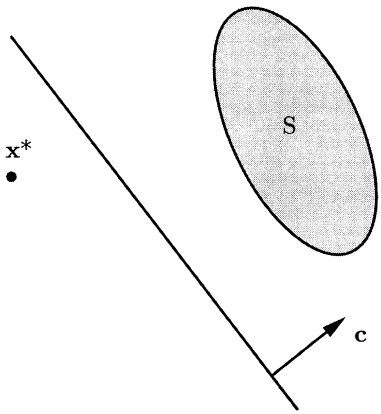


Figure 4.8: A hyperplane that separates the point $\mathbf{x}^*$ from the convex set S.

Furthermore, D is the intersection of the closed set S with the closed set B and is also closed. Finally, D is a bounded set because B is bounded. Consider the quantity $\|\mathbf{x} - \mathbf{x}^*\|$, where $\mathbf{x}$ ranges over the set D. This is a continuous function of $\mathbf{x}$. Since D is nonempty, closed, and bounded, Weierstrass' theorem implies that there exists some $\mathbf{y} \in D$ such that

$$\|\mathbf{y} - \mathbf{x}^*\| \leq \|\mathbf{x} - \mathbf{x}^*\|, \qquad \forall\ \mathbf{x} \in D.$$

For any $\mathbf{x} \in S$ that does not belong to D, we have $\|\mathbf{x} - \mathbf{x}^*\| > \|\mathbf{w} - \mathbf{x}^*\| \geq \|\mathbf{y} - \mathbf{x}^*\|$. We conclude that $\mathbf{y}$ minimizes $\|\mathbf{x} - \mathbf{x}^*\|$ over all $\mathbf{x} \in S$.

We have so far established that there exists an element $\mathbf{y}$ of S which is closest to $\mathbf{x}^*$. We now show that the vector $\mathbf{c} = \mathbf{y} - \mathbf{x}^*$ has the desired property [see Figure 4.9(b)].

Let $\mathbf{x} \in S$. For any λ satisfying $0 < \lambda \leq 1$, we have $\mathbf{y} + \lambda(\mathbf{x} - \mathbf{y}) \in S$, because S is convex. Since $\mathbf{y}$ minimizes $\|\mathbf{x} - \mathbf{x}^*\|$ over all $\mathbf{x} \in S$, we obtain

$$\begin{aligned} \|\mathbf{y} - \mathbf{x}^*\|^2 &\leq \|\mathbf{y} + \lambda(\mathbf{x} - \mathbf{y}) - \mathbf{x}^*\|^2 \\ &= \|\mathbf{y} - \mathbf{x}^*\|^2 + 2\lambda(\mathbf{y} - \mathbf{x}^*)'(\mathbf{x} - \mathbf{y}) + \lambda^2\|\mathbf{x} - \mathbf{y}\|^2, \end{aligned}$$

which yields

$$2\lambda(\mathbf{y} - \mathbf{x}^*)'(\mathbf{x} - \mathbf{y}) + \lambda^2\|\mathbf{x} - \mathbf{y}\|^2 \geq 0.$$

We divide by λ and then take the limit as λ decreases to zero. We obtain

$$(\mathbf{y} - \mathbf{x}^*)'(\mathbf{x} - \mathbf{y}) \geq 0.$$

[This inequality states that the angle θ in Figure 4.9(b) is no larger than 90 degrees.] Thus,

$$(\mathbf{y} - \mathbf{x}^*)'\mathbf{x} \geq (\mathbf{y} - \mathbf{x}^*)'\mathbf{y}$$

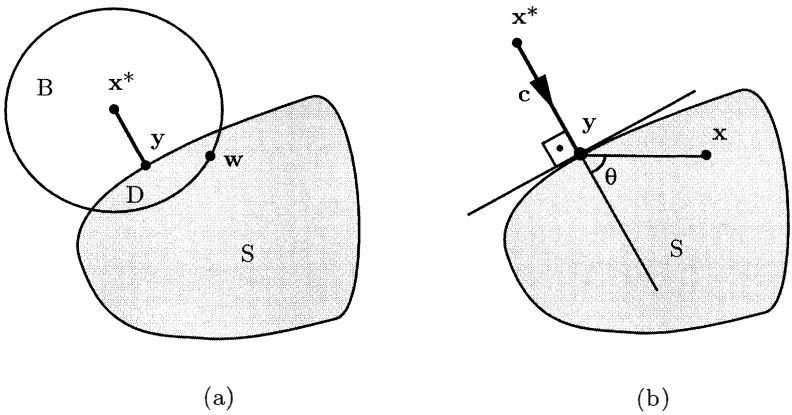


Figure 4.9: Illustration of the proof of the separating hyperplane theorem.

$$\begin{aligned} &= (\mathbf{y} - \mathbf{x}^*)'\mathbf{x}^* + (\mathbf{y} - \mathbf{x}^*)'(\mathbf{y} - \mathbf{x}^*) \\ &> (\mathbf{y} - \mathbf{x}^*)'\mathbf{x}^*. \end{aligned}$$

Setting $\mathbf{c} = \mathbf{y} - \mathbf{x}^*$ proves the theorem. □

Farkas' lemma revisited

We now show that Farkas' lemma is a consequence of the separating hyperplane theorem.

We will only be concerned with the difficult half of Farkas' lemma. In particular, we will prove that if the system $\mathbf{Ax} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, does not have a solution, then there exists a vector $\mathbf{p}$ such that $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$ and $\mathbf{p}'\mathbf{b} < 0$.

Let

$$\begin{aligned} S &= \{\mathbf{Ax} \mid \mathbf{x} \geq \mathbf{0}\} \\ &= \{\mathbf{y} \mid \text{there exists } \mathbf{x} \text{ such that } \mathbf{y} = \mathbf{Ax},\ \mathbf{x} \geq \mathbf{0}\}, \end{aligned}$$

and suppose that the vector $\mathbf{b}$ does not belong to S. The set S is clearly convex; it is also nonempty because $\mathbf{0} \in S$. Finally, the set S is closed; this may seem obvious, but is not easy to prove. For one possible proof, note that S is the projection of the polyhedron $\{(\mathbf{x}, \mathbf{y}) \mid \mathbf{y} = \mathbf{Ax},\ \mathbf{x} \geq \mathbf{0}\}$ onto the $\mathbf{y}$ coordinates, is itself a polyhedron (see Section 2.8), and is therefore closed. An alternative proof is outlined in Exercise 4.37.

We now invoke the separating hyperplane theorem to separate $\mathbf{b}$ from S and conclude that there exists a vector $\mathbf{p}$ such that $\mathbf{p}'\mathbf{b} < \mathbf{p}'\mathbf{y}$ for every

$\mathbf{y} \in S$. Since $\mathbf{0} \in S$, we must have $\mathbf{p}'\mathbf{b} < 0$. Furthermore, for every column $\mathbf{A}_i$ of $\mathbf{A}$ and every $\lambda > 0$, we have $\lambda\mathbf{A}_i \in S$ and $\mathbf{p}'\mathbf{b} < \lambda\mathbf{p}'\mathbf{A}_i$. We divide both sides of the latter inequality by λ and then take the limit as λ tends to infinity, to conclude that $\mathbf{p}'\mathbf{A}_i \geq 0$. Since this is true for every i, we obtain $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$ and the proof is complete.

The duality theorem revisited

We will now derive the duality theorem as a corollary of Farkas' lemma. We only provide the proof for the case where the primal constraints are of the form $\mathbf{A}\mathbf{x} \geq \mathbf{b}$. The proof for the general case can be constructed along the same lines at the expense of more notation (Exercise 4.38). We also note that the proof given here is very similar to the line of argument used in the heuristic explanation of the duality theorem in Example 4.4.

We consider the following pair of primal and dual problems

$$\begin{array}{lll} \text{minimize} & \mathbf{c}'\mathbf{x} & \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b}, & \end{array} \qquad \begin{array}{lll} \text{maximize} & \mathbf{p}'\mathbf{b} & \\ \text{subject to} & \mathbf{p}'\mathbf{A} = \mathbf{c}' & \\ & \mathbf{p} \geq \mathbf{0}, & \end{array}$$

and we assume that the primal has an optimal solution $\mathbf{x}^*$. We will show that the dual problem also has a feasible solution with the same cost. Once this is done, the strong duality theorem follows from weak duality (cf. Corollary 4.2).

Let $I = \{i \mid \mathbf{a}_i'\mathbf{x}^* = b_i\}$ be the set of indices of the constraints that are active at $\mathbf{x}^*$. We will first show that any vector $\mathbf{d}$ that satisfies $\mathbf{a}_i'\mathbf{d} \geq 0$ for every $i \in I$, must also satisfy $\mathbf{c}'\mathbf{d} \geq 0$. Consider such a vector $\mathbf{d}$ and let ϵ be a positive scalar. We then have $\mathbf{a}_i'(\mathbf{x}^* + \epsilon\mathbf{d}) \geq \mathbf{a}_i\mathbf{x}^* = b_i$ for all $i \in I$. In addition, if $i \notin I$ and if ϵ is sufficiently small, the inequality $\mathbf{a}_i'\mathbf{x}^* > b_i$ implies that $\mathbf{a}_i'(\mathbf{x}^* + \epsilon\mathbf{d}) > b_i$. We conclude that when ϵ is sufficiently small, $\mathbf{x}^* + \epsilon\mathbf{d}$ is a feasible solution. By the optimality of $\mathbf{x}^*$, we obtain $\mathbf{c}'\mathbf{d} \geq 0$, which establishes our claim. By Farkas' lemma (cf. Corollary 4.3), $\mathbf{c}$ can be expressed as a nonnegative linear combination of the vectors $\mathbf{a}_i$, $i \in I$, and there exist nonnegative scalars p_i, $i \in I$, such that

$$\mathbf{c} = \sum_{i \in I} p_i \mathbf{a}_i. \tag{4.3}$$

For $i \notin I$, we define $p_i = 0$. We then have $\mathbf{p} \geq \mathbf{0}$ and Eq. (4.3) shows that the vector $\mathbf{p}$ satisfies the dual constraint $\mathbf{p}'\mathbf{A} = \mathbf{c}'$. In addition,

$$\mathbf{p}'\mathbf{b} = \sum_{i \in I} p_i b_i = \sum_{i \in I} p_i \mathbf{a}_i'\mathbf{x}^* = \mathbf{c}'\mathbf{x}^*,$$

which shows that the cost of this dual feasible solution $\mathbf{p}$ is the same as the optimal primal cost. The duality theorem now follows from Corollary 4.2.

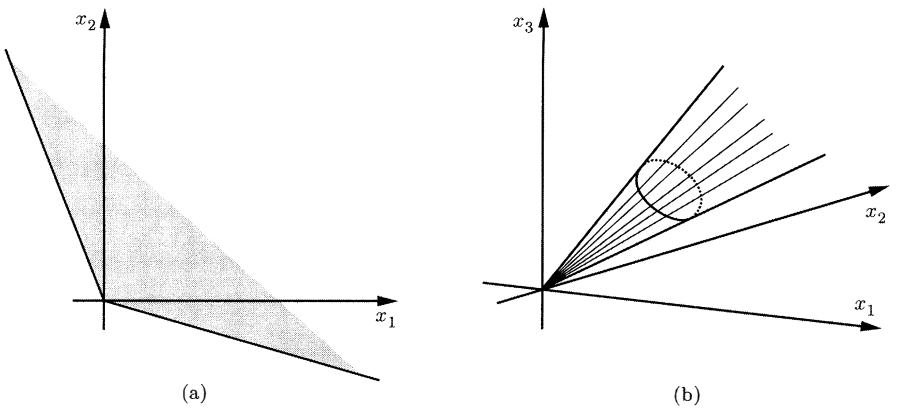


Figure 4.10: Examples of cones.

In conclusion, we have accomplished the goals that were set out in the beginning of this section. We proved the separating hyperplane theorem, which is a very intuitive and seemingly simple result, but with many important ramifications in optimization and other areas in mathematics. We used the separating hyperplane theorem to establish Farkas' lemma, and finally showed that the strong duality theorem is an easy consequence of Farkas' lemma.

4.8 Cones and extreme rays

We have seen in Chapter 2, that if the optimal cost in a linear programming problem is finite, then our search for an optimal solution can be restricted to finitely many points, namely, the basic feasible solutions, assuming one exists. In this section, we wish to develop a similar result for the case where the optimal cost is $-\infty$. In particular, we will show that the optimal cost is $-\infty$ if and only if there exists a cost reducing direction along which we can move without ever leaving the feasible set. Furthermore, our search for such a direction can be restricted to a finite set of suitably defined "extreme rays."

Cones

The first step in our development is to introduce the concept of a cone.

Definition 4.1 *A set $C \subset \Re^n$ is a* **cone** *if $\lambda \mathbf{x} \in C$ for all $\lambda \geq 0$ and all $\mathbf{x} \in C$.*

Notice that if C is a nonempty cone, then $\mathbf{0} \in C$. To this see, consider an arbitrary element $\mathbf{x}$ of C and set $\lambda = 0$ in the definition of a cone; see also Figure 4.10. A polyhedron of the form $P = \{\mathbf{x} \in \Re^n \mid \mathbf{Ax} \geq \mathbf{0}\}$ is easily seen to be a nonempty cone and is called a *polyhedral cone.*

Let $\mathbf{x}$ be a nonzero element of a polyhedral cone C. We then have $3\mathbf{x}/2 \in C$ and $\mathbf{x}/2 \in C$. Since $\mathbf{x}$ is the average of $3\mathbf{x}/2$ and $\mathbf{x}/2$, it is not an extreme point and, therefore, the only possible extreme point is the zero vector. If the zero vector is indeed an extreme point, we say that the cone is *pointed.* Whether this will be the case or not is determined by the criteria provided by our next result.

Theorem 4.12 *Let $C \subset \Re^n$ be the polyhedral cone defined by the constraints $\mathbf{a}_i'\mathbf{x} \geq 0$, $i = 1, \ldots, m$. Then, the following are equivalent:*

(a) *The zero vector is an extreme point of C.*

(b) *The cone C does not contain a line.*

(c) *There exist n vectors out of the family $\mathbf{a}_1, \ldots, \mathbf{a}_m$, which are linearly independent.*

Proof. This result is a special case of Theorem 2.6 in Section 2.5. □

Rays and recession cones

Consider a nonempty polyhedron

$$P = \{\mathbf{x} \in \Re^n \mid \mathbf{Ax} \geq \mathbf{b}\},$$

and let us fix some $\mathbf{y} \in P$. We define the *recession cone at* $\mathbf{y}$ as the set of all directions $\mathbf{d}$ along which we can move indefinitely away from $\mathbf{y}$, without leaving the set P. More formally, the recession cone is defined as the set

$$\{\mathbf{d} \in \Re^n \mid \mathbf{A}(\mathbf{y} + \lambda\mathbf{d}) \geq \mathbf{b}, \text{ for all } \lambda \geq 0\}.$$

It is easily seen that this set is the same as

$$\{\mathbf{d} \in \Re^n \mid \mathbf{Ad} \geq \mathbf{0}\},$$

and is a polyhedral cone. This shows that the recession cone is independent of the starting point $\mathbf{y}$; see Figure 4.11. The nonzero elements of the recession cone are called the *rays* of the polyhedron P.

For the case of a nonempty polyhedron $P = \{\mathbf{x} \in \Re^n \mid \mathbf{Ax} = \mathbf{b}, \ \mathbf{x} \geq \mathbf{0}\}$ in standard form, the recession cone is seen to be the set of all vectors $\mathbf{d}$ that satisfy

$$\mathbf{Ad} = \mathbf{0}, \qquad \mathbf{d} \geq \mathbf{0}.$$

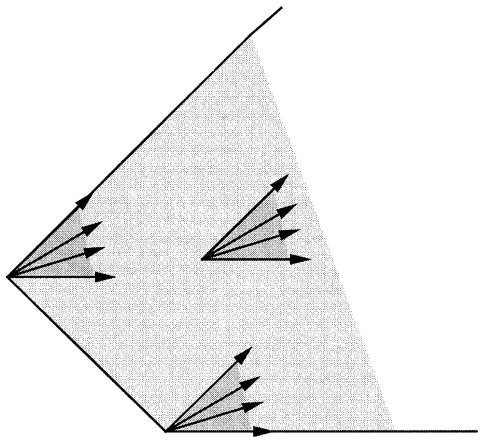


Figure 4.11: The recession cone at different elements of a polyhedron.

Extreme rays

We now define the extreme rays of a polyhedron. Intuitively, these are the directions associated with "edges" of the polyhedron that extend to infinity; see Figure 4.12 for an illustration.

Definition 4.2

(a) *A nonzero element* $\mathbf{x}$ *of a polyhedral cone* $C \subset \Re^n$ *is called an* **extreme ray** *if there are* $n-1$ *linearly independent constraints that are active at* $\mathbf{x}$.

(b) *An extreme ray of the recession cone associated with a nonempty polyhedron* P *is also called an* **extreme ray** *of* P.

Note that a positive multiple of an extreme ray is also an extreme ray. We say that two extreme rays are *equivalent* if one is a positive multiple of the other. Note that for this to happen, they must correspond to the same $n-1$ linearly independent active constraints. Any $n-1$ linearly independent constraints define a line and can lead to at most two nonequivalent extreme rays (one being the negative of the other). Given that there is a finite number of ways that we can choose $n-1$ constraints to become active, and as long as we do not distinguish between equivalent extreme rays, we conclude that the number of extreme rays of a polyhedron is finite. A finite collection of extreme rays will be said to be a *complete set of extreme rays* if it contains exactly one representative from each equivalence class.

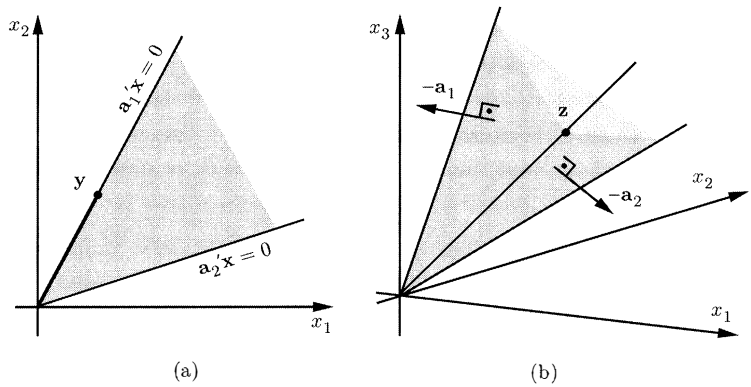


Figure 4.12: Extreme rays of polyhedral cones. (a) The vector $\mathbf{y}$ is an extreme ray because $n = 2$ and the constraint $\mathbf{a}_1'\mathbf{x} = 0$ is active at $\mathbf{y}$. (b) A polyhedral cone defined by three linearly independent constraints of the form $\mathbf{a}_i'\mathbf{x} \geq 0$. The vector $\mathbf{z}$ is an extreme ray because $n = 3$ and the two linearly independent constraints $\mathbf{a}_1'\mathbf{x} \geq 0$ and $\mathbf{a}_2'\mathbf{x} \geq 0$ are active at $\mathbf{z}$.

The definition of extreme rays mimics the definition of basic feasible solutions. An alternative and equivalent definition, resembling the definition of extreme points of polyhedra, is explored in Exercise 4.39.

Characterization of unbounded linear programming problems

We now derive conditions under which the optimal cost in a linear programming problem is equal to $-\infty$, first for the case where the feasible set is a cone, and then for the general case.

Theorem 4.13 *Consider the problem of minimizing* $\mathbf{c}'\mathbf{x}$ *over a pointed polyhedral cone* $C = \{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \geq 0,\ i = 1, \ldots, m\}$. *The optimal cost is equal to* $-\infty$ *if and only if some extreme ray* $\mathbf{d}$ *of* C *satisfies* $\mathbf{c}'\mathbf{d} < 0$.

Proof. One direction of the result is trivial because if some extreme ray has negative cost, then the cost becomes arbitrarily negative by moving along this ray.

For the converse, suppose that the optimal cost is $-\infty$. In particular, there exists some $\mathbf{x} \in C$ whose cost is negative and, by suitably scaling $\mathbf{x}$,

we can assume that $\mathbf{c}'\mathbf{x} = -1$. In particular, the polyhedron

$$P = \{\mathbf{x} \in \Re^n \mid \mathbf{a}_1'\mathbf{x} \geq 0, \ldots, \mathbf{a}_m'\mathbf{x} \geq 0, \mathbf{c}'\mathbf{x} = -1\}$$

is nonempty. Since C is pointed, the vectors $\mathbf{a}_1, \ldots, \mathbf{a}_m$ span $\Re^n$ and this implies that P has at least one extreme point; let $\mathbf{d}$ be one of them. At $\mathbf{d}$, we have n linearly independent active constraints, which means that $n-1$ linearly independent constraints of the form $\mathbf{a}_i'\mathbf{x} \geq 0$ must be active. It follows that $\mathbf{d}$ is an extreme ray of C. □

By exploiting duality, Theorem 4.13 leads to a criterion for unboundedness in general linear programming problems. Interestingly enough, this criterion does not involve the right-hand side vector $\mathbf{b}$.

Theorem 4.14 *Consider the problem of minimizing* $\mathbf{c}'\mathbf{x}$ *subject to* $\mathbf{A}\mathbf{x} \geq \mathbf{b}$, *and assume that the feasible set has at least one extreme point. The optimal cost is equal to* $-\infty$ *if and only if some extreme ray* $\mathbf{d}$ *of the feasible set satisfies* $\mathbf{c}'\mathbf{d} < 0$.

Proof. One direction of the result is trivial because if an extreme ray has negative cost, then the cost becomes arbitrarily negative by starting at a feasible solution and moving along the direction of this ray.

For the proof of the reverse direction, we consider the dual problem:

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} = \mathbf{c}' \\ & \mathbf{p} \geq \mathbf{0}. \end{aligned}$$

If the primal problem is unbounded, the dual problem is infeasible. Then, the related problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{0} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} = \mathbf{c}' \\ & \mathbf{p} \geq \mathbf{0}, \end{aligned}$$

is also infeasible. This implies that the associated primal problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{0}, \end{aligned}$$

is either unbounded or infeasible. Since $\mathbf{x} = \mathbf{0}$ is one feasible solution, it must be unbounded. Since the primal feasible set has at least one extreme point, the rows of $\mathbf{A}$ span $\Re^n$, where n is the dimension of $\mathbf{x}$. It follows that the recession cone $\{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq \mathbf{0}\}$ is pointed and, by Theorem 4.13, there exists an extreme ray $\mathbf{d}$ of the recession cone satisfying $\mathbf{c}'\mathbf{d} < 0$. By definition, this is an extreme ray of the feasible set. □

The unboundedness criterion in the simplex method

We end this section by pointing out that if we have a standard form problem in which the optimal cost is $-\infty$, the simplex method provides us at termination with an extreme ray.

Indeed, consider what happens when the simplex method terminates with an indication that the optimal cost is $-\infty$. At that point, we have a basis matrix $\mathbf{B}$, a nonbasic variable x_j with negative reduced cost, and the jth column $\mathbf{B}^{-1}\mathbf{A}_j$ of the tableau has no positive elements. Consider the jth basic direction $\mathbf{d}$, which is the vector that satisfies $\mathbf{d}_B = -\mathbf{B}^{-1}\mathbf{A}_j$, $d_j = 1$, and $d_i = 0$ for every nonbasic index i other than j. Then, the vector $\mathbf{d}$ satisfies $\mathbf{Ad} = \mathbf{0}$ and $\mathbf{d} \geq \mathbf{0}$, and belongs to the recession cone. It is also a direction of cost decrease, since the reduced cost $\overline{c}_j$ of the entering variable is negative.

Out of the constraints defining the recession cone, the jth basic direction $\mathbf{d}$ satisfies $n-1$ linearly independent such constraints with equality: these are the constraints $\mathbf{Ad} = \mathbf{0}$ (m of them) and the constraints $d_i = 0$ for i nonbasic and different than j ($n - m - 1$ of them). We conclude that $\mathbf{d}$ is an extreme ray.

4.9 Representation of polyhedra

In this section, we establish one of the fundamental results of linear programming theory. In particular, we show that any element of a polyhedron that has at least one extreme point can be represented as a convex combination of extreme points plus a nonnegative linear combination of extreme rays. A precise statement is given by our next result. A generalization to the case of general polyhedra is developed in Exercise 4.47.

Theorem 4.15 (Resolution theorem) *Let*

$$P = \left\{ \mathbf{x} \in \Re^n \mid \mathbf{Ax} \geq \mathbf{b} \right\}$$

be a nonempty polyhedron with at least one extreme point. Let $\mathbf{x}^1, \ldots, \mathbf{x}^k$ *be the extreme points, and let* $\mathbf{w}^1, \ldots, \mathbf{w}^r$ *be a complete set of extreme rays of* P. *Let*

$$Q = \left\{ \sum_{i=1}^{k} \lambda_i \mathbf{x}^i + \sum_{j=1}^{r} \theta_j \mathbf{w}^j \;\middle|\; \lambda_i \geq 0,\ \theta_j \geq 0,\ \sum_{i=1}^{k} \lambda_i = 1 \right\}.$$

Then, $Q = P$.

Proof. We first prove that $Q \subset P$. Let

$$\mathbf{x} = \sum_{i=1}^{k} \lambda_i \mathbf{x}^i + \sum_{j=1}^{r} \theta_j \mathbf{w}^j$$

be an element of Q, where the coefficients λ_i and θ_j are nonnegative, and $\sum_{i=1}^{k} \lambda_i = 1$. The vector $\mathbf{y} = \sum_{i=1}^{k} \lambda_i \mathbf{x}^i$ is a convex combination of elements of P. It therefore belongs to P and satisfies $\mathbf{A}\mathbf{y} \geq \mathbf{b}$. We also have $\mathbf{A}\mathbf{w}^j \geq \mathbf{0}$ for every j, which implies that the vector $\mathbf{z} = \sum_{j=1}^{r} \theta_j \mathbf{w}^j$ satisfies $\mathbf{A}\mathbf{z} \geq \mathbf{0}$. It then follows that the vector $\mathbf{x} = \mathbf{y} + \mathbf{z}$ satisfies $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and belongs to P.

For the reverse inclusion, we assume that P is not a subset of Q and we will derive a contradiction. Let $\mathbf{z}$ be an element of P that does not belong to Q. Consider the linear programming problem

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{k} 0\lambda_i + \sum_{j=1}^{r} 0\theta_j \\
\text{subject to} \quad & \sum_{i=1}^{k} \lambda_i \mathbf{x}^i + \sum_{j=1}^{r} \theta_j \mathbf{w}^j = \mathbf{z} \\
& \sum_{i=1}^{k} \lambda_i = 1 \\
& \lambda_i \geq 0, \qquad i = 1, \ldots, k, \\
& \theta_j \geq 0, \qquad j = 1, \ldots, r,
\end{aligned} \tag{4.4}$$

which is infeasible because $\mathbf{z} \notin Q$. This problem is the dual of the problem

$$\begin{aligned}
\text{minimize} \quad & \mathbf{p}'\mathbf{z} + q \\
\text{subject to} \quad & \mathbf{p}'\mathbf{x}^i + q \geq 0, \qquad i = 1, \ldots, k, \\
& \mathbf{p}'\mathbf{w}^j \geq 0, \qquad j = 1, \ldots, r.
\end{aligned} \tag{4.5}$$

Because the latter problem has a feasible solution, namely, $\mathbf{p} = \mathbf{0}$ and $q = 0$, the optimal cost is $-\infty$, and there exists a feasible solution $(\mathbf{p}, q)$ whose cost $\mathbf{p}'\mathbf{z} + q$ is negative. On the other hand, $\mathbf{p}'\mathbf{x}^i + q \geq 0$ for all i and this implies that $\mathbf{p}'\mathbf{z} < \mathbf{p}'\mathbf{x}^i$ for all i. We also have $\mathbf{p}'\mathbf{w}^j \geq 0$ for all j. [1]

Having fixed $\mathbf{p}$ as above, we now consider the linear programming problem

$$\begin{aligned}
\text{minimize} \quad & \mathbf{p}'\mathbf{x} \\
\text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b}.
\end{aligned}$$

If the optimal cost is finite, there exists an extreme point $\mathbf{x}^i$ which is optimal. Since $\mathbf{z}$ is a feasible solution, we obtain $\mathbf{p}'\mathbf{x}^i \leq \mathbf{p}'\mathbf{z}$, which is a

[1] For an intuitive view of this proof, the purpose of this paragraph was to construct a hyperplane that separates $\mathbf{z}$ from Q.

contradiction. If the optimal cost is $-\infty$, Theorem 4.14 implies that there exists an extreme ray $\mathbf{w}^j$ such that $\mathbf{p}'\mathbf{w}^j < 0$, which is again a contradiction. □

Example 4.10 Consider the unbounded polyhedron defined by the constraints

$$\begin{aligned} x_1 - x_2 &\geq -2 \\ x_1 + x_2 &\geq 1 \\ x_1, x_2 &\geq 0 \end{aligned}$$

(see Figure 4.13). This polyhedron has three extreme points, namely, $\mathbf{x}^1 = (0, 2)$, $\mathbf{x}^2 = (0, 1)$, and $\mathbf{x}^3 = (1, 0)$. The recession cone C is described by the inequalities $d_1 - d_2 \geq 0$, $d_1 + d_2 \geq 0$, and $d_1, d_2 \geq 0$. We conclude that $C = \{(d_1, d_2) \mid 0 \leq d_2 \leq d_1\}$. This cone has two extreme rays, namely, $\mathbf{w}^1 = (1, 1)$ and $\mathbf{w}^2 = (1, 0)$. The vector $\mathbf{y} = (2, 2)$ is an element of the polyhedron and can be represented as

$$\mathbf{y} = \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \mathbf{x}^2 + \mathbf{w}^1 + \mathbf{w}^2.$$

However, this representation is not unique; for example, we also have

$$\mathbf{y} = \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 1 \\ 0 \end{bmatrix} + \frac{3}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{2}\mathbf{x}^2 + \frac{1}{2}\mathbf{x}^3 + \frac{3}{2}\mathbf{w}^1.$$

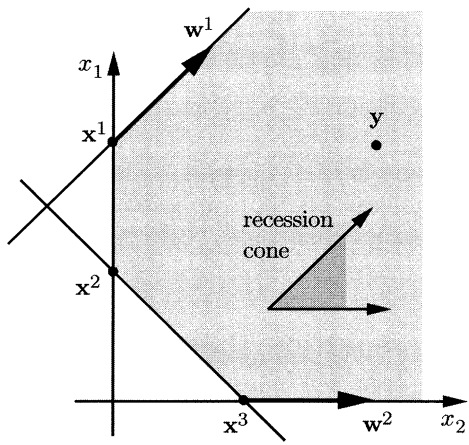


Figure 4.13: The polyhedron of Example 4.10.

We note that the set Q in Theorem 4.15 is the image of the polyhedron

$$H = \left\{ (\lambda_1, \ldots, \lambda_k, \theta_1, \ldots, \theta_r) \;\middle|\; \sum_{i=1}^{k} \lambda_i = 1,\ \lambda_i \geq 0,\ \theta_j \geq 0 \right\},$$

under the linear mapping

$$(\lambda_1, \ldots, \lambda_k, \theta_1, \ldots, \theta_r) \mapsto \sum_{i=1}^{k} \lambda_i \mathbf{x}^i + \sum_{j=1}^{r} \theta_j \mathbf{w}^j.$$

Thus, one corollary of the resolution theorem is that every polyhedron is the image, under a linear mapping, of a polyhedron H with this particular structure.

We now specialize Theorem 4.15 to the case of bounded polyhedra, to recover a result that was also proved in Section 2.7, using a different line of argument.

Corollary 4.4 *A nonempty bounded polyhedron is the convex hull of its extreme points.*

Proof. Let $P = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ be a nonempty bounded polyhedron. If $\mathbf{d}$ is a nonzero element of the cone $C = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq \mathbf{0}\}$ and $\mathbf{x}$ is an element of P, we have $\mathbf{x} + \lambda\mathbf{d} \in P$ for all $\lambda \geq 0$, contradicting the boundedness of P. We conclude that C consists of only the zero vector and does not have any extreme rays. The result then follows from Theorem 4.15. □

There is another corollary of Theorem 4.15 that deals with cones, and which is proved by noting that a cone can have no extreme points other than the zero vector.

Corollary 4.5 *Assume that the cone $C = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} \geq \mathbf{0}\}$ is pointed. Then, every element of C can be expressed as a nonnegative linear combination of the extreme rays of C.*

Converse to the resolution theorem

Let us say that a set Q is *finitely generated* if it is specified in the form

$$Q = \left\{ \sum_{i=1}^{k} \lambda_i \mathbf{x}^i + \sum_{j=1}^{r} \theta_j \mathbf{w}^j \;\middle|\; \lambda_i \geq 0,\ \theta_j \geq 0,\ \sum_{i=1}^{k} \lambda_i = 1 \right\}, \tag{4.6}$$

where $\mathbf{x}^1, \ldots, \mathbf{x}^k$ and $\mathbf{w}^1, \ldots, \mathbf{w}^r$ are some given elements of $\Re^n$. The resolution theorem states that a polyhedron with at least one extreme point is a finitely generated set (this is also true for general polyhedra; see Exercise 4.47). We now discuss a converse result, which states that every finitely generated set is a polyhedron.

As observed earlier, a finitely generated set Q can be viewed as the image of the polyhedron

$$H = \left\{ (\lambda_1, \ldots, \lambda_k, \theta_1, \ldots, \theta_r) \;\middle|\; \sum_{i=1}^{k} \lambda_i = 1,\ \lambda_i \geq 0,\ \theta_j \geq 0 \right\}$$

under a certain linear mapping. Thus, the results of Section 2.8 apply and establish that a finitely generated set is indeed a polyhedron. We record this result and also present a proof based on duality.

Theorem 4.16 *A finitely generated set is a polyhedron. In particular, the convex hull of finitely many vectors is a (bounded) polyhedron.*

Proof. Consider the linear programming problem (4.4) that was used in the proof of Theorem 4.15. A given vector $\mathbf{z}$ belongs to a finitely generated set Q of the form (4.6) if and only if the problem (4.4) has a feasible solution. Using duality, this is the case if and only if problem (4.5) has finite optimal cost. We convert problem (4.5) to standard form by introducing nonnegative variables $\mathbf{p}^+, \mathbf{p}^-, q^+, q^-$, such that $\mathbf{p} = \mathbf{p}^+ - \mathbf{p}^-$, and $q = q^+ - q^-$, as well as surplus variables. Since standard form polyhedra contain no lines, Theorem 4.13 shows that the optimal cost in the standard form problem is finite if and only if

$$(\mathbf{p}^+)'\mathbf{z} - (\mathbf{p}^-)'\mathbf{z} + q^+ - q^- \geq 0,$$

for each one of its finitely many extreme rays. Hence, $\mathbf{z} \in Q$ if and only if $\mathbf{z}$ satisfies a finite collection of linear inequalities. This shows that Q is a polyhedron. □

In conclusion, we have two ways of representing a polyhedron:

(a) in terms of a finite set of linear constraints;

(b) as a finitely generated set, in terms of its extreme points and extreme rays.

These two descriptions are mathematically equivalent, but can be quite different from a practical viewpoint. For example, we may be able to describe a polyhedron in terms of a small number of linear constraints. If on the other hand, this polyhedron has many extreme points, a description as a finitely generated set can be much more complicated. Furthermore, passing from one type of description to the other is, in general, a complicated computational task.

4.10 General linear programming duality*

In the definition of the dual problem (Section 4.2), we associated a dual variable p_i with each constraint of the form $\mathbf{a}_i'\mathbf{x} = b_i$, $\mathbf{a}_i'\mathbf{x} \geq b_i$, or $\mathbf{a}_i'\mathbf{x} \leq b_i$.

However, no dual variables were associated with constraints of the form $x_i \geq 0$ or $x_i \leq 0$. In the same spirit, and in a more general approach to linear programming duality, we can choose arbitrarily which constraints will be associated with price variables and which ones will not. In this section, we develop a general duality theorem that covers such a situation.

Consider the primal problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \in P, \end{aligned}$$

where P is the polyhedron

$$P = \{\mathbf{x} \mid \mathbf{D}\mathbf{x} \geq \mathbf{d}\}.$$

We associate a dual vector $\mathbf{p}$ with the constraint $\mathbf{A}\mathbf{x} \geq \mathbf{b}$. The constraint $\mathbf{x} \in P$ is a generalization of constraints of the form $x_i \geq 0$ or $x_i \leq 0$ and dual variables are not associated with it.

As in Section 4.1, we define the *dual objective* $g(\mathbf{p})$ by

$$g(\mathbf{p}) = \min_{\mathbf{x} \in P} \Big[\mathbf{c}'\mathbf{x} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x})\Big]. \tag{4.7}$$

The *dual problem* is then defined as

$$\begin{aligned} \text{maximize} \quad & g(\mathbf{p}) \\ \text{subject to} \quad & \mathbf{p} \geq \mathbf{0}. \end{aligned}$$

We first provide a generalization of the weak duality theorem.

Theorem 4.17 (Weak duality) *If* $\mathbf{x}$ *is primal feasible* ($\mathbf{A}\mathbf{x} \geq \mathbf{b}$ *and* $\mathbf{x} \in P$), *and* $\mathbf{p}$ *is dual feasible* ($\mathbf{p} \geq \mathbf{0}$), *then* $g(\mathbf{p}) \leq \mathbf{c}'\mathbf{x}$.

Proof. If $\mathbf{x}$ and $\mathbf{p}$ are primal and dual feasible, respectively, then $\mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}) \leq 0$, which implies that

$$\begin{aligned} g(\mathbf{p}) &= \min_{\mathbf{y} \in P} \Big[\mathbf{c}'\mathbf{y} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{y})\Big] \\ &\leq \mathbf{c}'\mathbf{x} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}) \\ &\leq \mathbf{c}'\mathbf{x}. \end{aligned}$$

□

We also have the following generalization of the strong duality theorem.

Theorem 4.18 (Strong duality) *If the primal problem has an optimal solution, so does the dual, and the respective optimal costs are equal.*

Proof. Since $P = \{\mathbf{x} \mid \mathbf{D}\mathbf{x} \geq \mathbf{d}\}$, the primal problem is of the form

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{D}\mathbf{x} \geq \mathbf{d}, \end{aligned}$$

and we assume that it has an optimal solution. Its dual, which is

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} + \mathbf{q}'\mathbf{d} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} + \mathbf{q}'\mathbf{D} = \mathbf{c}' \\ & \mathbf{p} \geq \mathbf{0} \\ & \mathbf{q} \geq \mathbf{0}, \end{aligned} \tag{4.8}$$

must then have the same optimal cost. For any fixed $\mathbf{p}$, the vector $\mathbf{q}$ should be chosen optimally in the problem (4.8). Thus, the dual problem (4.8) can also be written as

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} + f(\mathbf{p}) \\ \text{subject to} \quad & \mathbf{p} \geq \mathbf{0}, \end{aligned}$$

where $f(\mathbf{p})$ is the optimal cost in the problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{q}'\mathbf{d} \\ \text{subject to} \quad & \mathbf{q}'\mathbf{D} = \mathbf{c}' - \mathbf{p}'\mathbf{A} \\ & \mathbf{q} \geq \mathbf{0}. \end{aligned} \tag{4.9}$$

[If the latter problem is infeasible, we set $f(\mathbf{p}) = -\infty$.] Using the strong duality theorem for problem (4.9), we obtain

$$f(\mathbf{p}) = \min_{\mathbf{D}\mathbf{x} \geq \mathbf{d}} (\mathbf{c}'\mathbf{x} - \mathbf{p}'\mathbf{A}\mathbf{x}).$$

We conclude that the dual problem (4.8) has the same optimal cost as the problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} + \min_{\mathbf{D}\mathbf{x} \geq \mathbf{d}} (\mathbf{c}'\mathbf{x} - \mathbf{p}'\mathbf{A}\mathbf{x}) \\ \text{subject to} \quad & \mathbf{p} \geq \mathbf{0}. \end{aligned}$$

By comparing with Eq. (4.7), we see that this is the same as maximizing $g(\mathbf{p})$ over all $\mathbf{p} \geq \mathbf{0}$. □

The idea of selectively assigning dual variables to some of the constraints is often used in order to treat "simpler" constraints differently than more "complex" ones, and has numerous applications in large scale optimization. (Applications to integer programming are discussed in Section 11.4.) Finally, let us point out that the approach in this section extends to certain nonlinear optimization problems. For example, if we replace the

linear cost function $\mathbf{c}'\mathbf{x}$ by a general convex function $c(\mathbf{x})$, and the polyhedron P by a general convex set, we can again define the dual objective according to the formula

$$g(\mathbf{p}) = \min_{\mathbf{x} \in P} \Big[c(\mathbf{x}) + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}) \Big].$$

It turns out that the strong duality theorem remains valid for such nonlinear problems, under suitable technical conditions, but this lies beyond the scope of this book.

4.11 Summary

We summarize here the main ideas that have been developed in this chapter.

Given a (primal) linear programming problem, we can associate with it another (dual) linear programming problem, by following a set of mechanical rules. The definition of the dual problem is consistent, in the sense that the duals of equivalent primal problems are themselves equivalent.

Each dual variable is associated with a particular primal constraint and can be viewed as a penalty for violating that constraint. By replacing the primal constraints with penalty terms, we increase the set of available options, and this allows us to construct primal solutions whose cost is less than the optimal cost. In particular, every dual feasible vector leads to a lower bound on the optimal cost of the primal problem (this is the essence of the weak duality theorem). The maximization in the dual problem is then a search for the tightest such lower bound. The strong duality theorem asserts that the tightest such lower bound is equal to the optimal primal cost.

An optimal dual variable can also be interpreted as a marginal cost, that is, as the rate of change of the optimal primal cost when we perform a small perturbation of the right-hand side vector $\mathbf{b}$, assuming nondegeneracy.

A useful relation between optimal primal and dual solutions is provided by the complementary slackness conditions. Intuitively, these conditions require that any constraint that is inactive at an optimal solution carries a zero price, which is compatible with the interpretation of prices as marginal costs.

We saw that every basis matrix in a standard form problem determines not only a primal basic solution, but also a basic dual solution. This observation is at the heart of the dual simplex method. This method is similar to the primal simplex method in that it generates a sequence of primal basic solutions, together with an associated sequence of dual basic solutions. It is different, however, in that the dual basic solutions are dual feasible, with ever improving costs, while the primal basic solutions are infeasible (except for the last one). We developed the dual simplex method by simply describing its mechanics and by providing an algebraic justification.

Nevertheless, the dual simplex method also has a geometric interpretation. It keeps moving from one dual basic feasible solution to an adjacent one and, in this respect, it is similar to the primal simplex method applied to the dual problem.

All of duality theory can be developed by exploiting the termination conditions of the simplex method, and this was our initial approach to the subject. We also pursued an alternative line of development that proceeded from first principles and used geometric arguments. This is a more direct and more general approach, but requires more abstract reasoning.

Duality theory provided us with some powerful tools based on which we were able to enhance our geometric understanding of polyhedra. We derived a few theorems of the alternative (like Farkas' lemma), which are surprisingly powerful and have applications in a wide variety of contexts. In fact, Farkas' lemma can be viewed as the core of linear programming duality theory. Another major result that we derived is the resolution theorem, which allows us to express any element of a nonempty polyhedron with at least one extreme point as a convex combination of its extreme points plus a nonnegative linear combination of its extreme rays; in other words, every polyhedron is "finitely generated." The converse is also true, and every finitely generated set is a polyhedron (can be represented in terms of linear inequality constraints). Results of this type play a key role in confirming our intuitive geometric understanding of polyhedra and linear programming. They allow us to develop alternative views of certain situations and lead to deeper understanding. Many such results have an "obvious" geometric content and are often taken for granted. Nevertheless, as we have seen, rigorous proofs can be quite elaborate.

4.12 Exercises

Exercise 4.1 Consider the linear programming problem:

$$\begin{array}{lll} \text{minimize} & x_1 - x_2 & \\ \text{subject to} & 2x_1 + 3x_2 - x_3 + x_4 \leq 0 \\ & 3x_1 + x_2 + 4x_3 - 2x_4 \geq 3 \\ & -x_1 - x_2 + 2x_3 + x_4 = 6 \\ & x_1 \leq 0 \\ & x_2, x_3 \geq 0. \end{array}$$

Write down the corresponding dual problem.

Exercise 4.2 Consider the primal problem

$$\begin{array}{lll} \text{minimize} & \mathbf{c}'\mathbf{x} & \\ \text{subject to} & \mathbf{A}\mathbf{x} & \geq \mathbf{b} \\ & \mathbf{x} & \geq \mathbf{0}. \end{array}$$

Form the dual problem and convert it into an equivalent minimization problem. Derive a set of conditions on the matrix $\mathbf{A}$ and the vectors $\mathbf{b}$, $\mathbf{c}$, under which the

dual is identical to the primal, and construct an example in which these conditions are satisfied.

Exercise 4.3 The purpose of this exercise is to show that solving linear programming problems is no harder than solving systems of linear inequalities.

Suppose that we are given a subroutine which, given a system of linear inequality constraints, either produces a solution or decides that no solution exists. Construct a simple algorithm that uses a single call to this subroutine and which finds an optimal solution to any linear programming problem that has an optimal solution.

Exercise 4.4 Let $\mathbf{A}$ be a symmetric square matrix. Consider the linear programming problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} \geq \mathbf{c} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Prove that if $\mathbf{x}^*$ satisfies $\mathbf{A}\mathbf{x}^* = \mathbf{c}$ and $\mathbf{x}^* \geq \mathbf{0}$, then $\mathbf{x}^*$ is an optimal solution.

Exercise 4.5 Consider a linear programming problem in standard form and assume that the rows of $\mathbf{A}$ are linearly independent. For each one of the following statements, provide either a proof or a counterexample.

(a) Let $\mathbf{x}^*$ be a basic feasible solution. Suppose that for every basis corresponding to $\mathbf{x}^*$, the associated basic solution to the dual is infeasible. Then, the optimal cost must be strictly less that $\mathbf{c}'\mathbf{x}^*$.

(b) The dual of the auxiliary primal problem considered in Phase I of the simplex method is always feasible.

(c) Let p_i be the dual variable associated with the ith equality constraint in the primal. Eliminating the ith primal equality constraint is equivalent to introducing the additional constraint $p_i = 0$ in the dual problem.

(d) If the unboundedness criterion in the primal simplex algorithm is satisfied, then the dual problem is infeasible.

Exercise 4.6* (Duality in Chebychev approximation) Let $\mathbf{A}$ be an $m \times n$ matrix and let $\mathbf{b}$ be a vector in $\Re^m$. We consider the problem of minimizing $\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_\infty$ over all $\mathbf{x} \in \Re^n$. Here $\|\cdot\|_\infty$ is the vector norm defined by $\|\mathbf{y}\|_\infty = \max_i |y_i|$. Let v be the value of the optimal cost.

(a) Let $\mathbf{p}$ be any vector in $\Re^m$ that satisfies $\sum_{i=1}^m |p_i| = 1$ and $\mathbf{p}'\mathbf{A} = \mathbf{0}'$. Show that $\mathbf{p}'\mathbf{b} \leq v$.

(b) In order to obtain the best possible lower bound of the form considered in part (a), we form the linear programming problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} = \mathbf{0}' \\ & \sum_{i=1}^m |p_i| \leq 1. \end{aligned}$$

Show that the optimal cost in this problem is equal to v.

Exercise 4.7 (Duality in piecewise linear convex optimization) Consider the problem of minimizing $\max_{i=1,\ldots,m}(\mathbf{a}_i'\mathbf{x} - b_i)$ over all $\mathbf{x} \in \Re^n$. Let v be the value of the optimal cost, assumed finite. Let $\mathbf{A}$ be the matrix with rows $\mathbf{a}_1, \ldots, \mathbf{a}_m$, and let $\mathbf{b}$ be the vector with components $b_1, \ldots, b_m$.

(a) Consider any vector $\mathbf{p} \in \Re^m$ that satisfies $\mathbf{p}'\mathbf{A} = \mathbf{0}'$, $\mathbf{p} \geq \mathbf{0}$, and $\sum_{i=1}^m p_i = 1$. Show that $-\mathbf{p}'\mathbf{b} \leq v$.

(b) In order to obtain the best possible lower bound of the form considered in part (a), we form the linear programming problem

$$\begin{aligned} \text{maximize} \quad & -\mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} = \mathbf{0}' \\ & \mathbf{p}'\mathbf{e} = 1 \\ & \mathbf{p} \geq \mathbf{0}, \end{aligned}$$

where $\mathbf{e}$ is the vector with all components equal to 1. Show that the optimal cost in this problem is equal to v.

Exercise 4.8 Consider the linear programming problem of minimizing $\mathbf{c}'\mathbf{x}$ subject to $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$. Let $\mathbf{x}^*$ be an optimal solution, assumed to exist, and let $\mathbf{p}^*$ be an optimal solution to the dual.

(a) Let $\tilde{\mathbf{x}}$ be an optimal solution to the primal, when $\mathbf{c}$ is replaced by some $\tilde{\mathbf{c}}$. Show that $(\tilde{\mathbf{c}} - \mathbf{c})'(\tilde{\mathbf{x}} - \mathbf{x}^*) \leq 0$.

(b) Let the cost vector be fixed at $\mathbf{c}$, but suppose that we now change $\mathbf{b}$ to $\tilde{\mathbf{b}}$, and let $\tilde{\mathbf{x}}$ be a corresponding optimal solution to the primal. Prove that $(\mathbf{p}^*)'(\tilde{\mathbf{b}} - \mathbf{b}) \leq \mathbf{c}'(\tilde{\mathbf{x}} - \mathbf{x}^*)$.

Exercise 4.9 (Back-propagation of dual variables in a multiperiod problem) A company makes a product that can be either sold or stored to meet future demand. Let $t = 1, \ldots, T$ denote the periods of the planning horizon. Let b_t be the production volume during period t, which is assumed to be known in advance. During each period t, a quantity x_t of the product is sold, at a unit price of d_t. Furthermore, a quantity y_t can be sent to long-term storage, at a unit transportation cost of c. Alternatively, a quantity w_t can be retrieved from storage, at zero cost. We assume that when the product is prepared for long-term storage, it is partly damaged, and only a fraction f of the total survives. Demand is assumed to be unlimited. The main question is whether it is profitable to store some of the production, in anticipation of higher prices in the future. This leads us to the following problem, where z_t stands for the amount kept in long-term storage, at the end of period t:

$$\begin{aligned} \text{maximize} \quad & \sum_{t=1}^{T} \alpha^{t-1}(d_t x_t - c y_t) + \alpha^T d_{T+1} z_T \\ \text{subject to} \quad & x_t + y_t - w_t = b_t, && t = 1, \ldots, T, \\ & z_t + w_t - z_{t-1} - f y_t = 0, && t = 1, \ldots, T, \\ & z_0 = 0, \\ & x_t, y_t, w_t, z_t \geq 0. \end{aligned}$$

Here, d_{T+1} is the salvage prive for whatever inventory is left at the end of period T. Furthermore, α is a discount factor, with $0 < \alpha < 1$, reflecting the fact that future revenues are valued less than current ones.

(a) Let p_t and q_t be dual variables associated with the first and second equality constraint, respectively. Write down the dual problem.

(b) Assume that $0 < f < 1$, $b_t \geq 0$, and $c \geq 0$. Show that the following formulae provide an optimal solution to the dual problem:

$$\begin{aligned} q_T &= \alpha^T d_{T+1}, \\ p_T &= \max\left\{\alpha^{T-1} d_T,\ f q_T - \alpha^{T-1} c\right\}, \\ q_t &= \max\left\{q_{t+1}, \alpha^{t-1} d_t\right\}, && t = 1, \ldots, T-1, \\ p_t &= \max\left\{\alpha^{t-1} d_t,\ f q_t - \alpha^{t-1} c\right\}, && t = 1, \ldots, T-1. \end{aligned}$$

(c) Explain how the result in part (b) can be used to compute an optimal solution to the original problem. Primal and dual nondegeneracy can be assumed.

Exercise 4.10 (Saddle points of the Lagrangean) Consider the standard form problem of minimizing $\mathbf{c}'\mathbf{x}$ subject to $\mathbf{A}\mathbf{x} = \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$. We define the *Lagrangean* by

$$L(\mathbf{x}, \mathbf{p}) = \mathbf{c}'\mathbf{x} + \mathbf{p}'(\mathbf{b} - \mathbf{A}\mathbf{x}).$$

Consider the following "game": player 1 chooses some $\mathbf{x} \geq \mathbf{0}$, and player 2 chooses some $\mathbf{p}$; then, player 1 pays to player 2 the amount $L(\mathbf{x}, \mathbf{p})$. Player 1 would like to minimize $L(\mathbf{x}, \mathbf{p})$, while player 2 would like to maximize it.

A pair $(\mathbf{x}^*, \mathbf{p}^*)$, with $\mathbf{x}^* \geq \mathbf{0}$, is called an *equilibrium* point (or a *saddle point*, or a *Nash equilibrium*) if

$$L(\mathbf{x}^*, \mathbf{p}) \leq L(\mathbf{x}^*, \mathbf{p}^*) \leq L(\mathbf{x}, \mathbf{p}^*), \qquad \forall\ \mathbf{x} \geq \mathbf{0},\ \forall\ \mathbf{p}.$$

(Thus, we have an equilibrium if no player is able to improve her performance by unilaterally modifying her choice.)

Show that a pair $(\mathbf{x}^*, \mathbf{p}^*)$ is an equilibrium if and only if $\mathbf{x}^*$ and $\mathbf{p}^*$ are optimal solutions to the standard form problem under consideration and its dual, respectively.

Exercise 4.11 Consider a linear programming problem in standard form which is infeasible, but which becomes feasible and has finite optimal cost when the last equality constraint is omitted. Show that the dual of the original (infeasible) problem is feasible and the optimal cost is infinite.

Exercise 4.12* (Degeneracy and uniqueness – I) Consider a general linear programming problem and suppose that we have a nondegenerate basic feasible solution to the primal. Show that the complementary slackness conditions lead to a system of equations for the dual vector that has a unique solution.

Exercise 4.13* (Degeneracy and uniqueness – II) Consider the following pair of problems that are duals of each other:

$$\begin{array}{lll} \text{minimize} & \mathbf{c}'\mathbf{x} & \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} & \\ & \mathbf{x} \geq \mathbf{0}, & \end{array} \qquad \begin{array}{ll} \text{maximize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p}'\mathbf{A} \leq \mathbf{c}'. \end{array}$$

(a) Prove that if one problem has a nondegenerate and unique optimal solution, so does the other.

(b) Suppose that we have a nondegenerate optimal basis for the primal and that the reduced cost for one of the basic variables is zero. What does the result of part (a) imply? Is it true that there must exist another optimal basis?

Exercise 4.14 (Degeneracy and uniqueness – III) Give an example in which the primal problem has a degenerate optimal basic feasible solution, but the dual has a unique optimal solution. (The example need not be in standard form.)

Exercise 4.15 (Degeneracy and uniqueness – IV) Consider the problem

$$\begin{aligned} \text{minimize} \quad & x_2 \\ \text{subject to} \quad & x_2 = 1 \\ & x_1 \geq 0 \\ & x_2 \geq 0. \end{aligned}$$

Write down its dual. For both the primal and the dual problem determine whether they have unique optimal solutions and whether they have nondegenerate optimal solutions. Is this example in agreement with the statement that nondegeneracy of an optimal basic feasible solution in one problem implies uniqueness of optimal solutions for the other? Explain.

Exercise 4.16 Give an example of a pair (primal and dual) of linear programming problems, both of which have multiple optimal solutions.

Exercise 4.17 This exercise is meant to demonstrate that knowledge of a primal optimal solution does not necessarily contain information that can be exploited to determine a dual optimal solution. In particular, determining an optimal solution to the dual is as hard as solving a system of linear inequalities, even if an optimal solution to the primal is available.

Consider the problem of minimizing $\mathbf{c}'\mathbf{x}$ subject to $\mathbf{A}\mathbf{x} \geq \mathbf{0}$, and suppose that we are told that the zero vector is optimal. Let the dimensions of $\mathbf{A}$ be $m \times n$, and suppose that we have an algorithm that determines a dual optimal solution and whose running time $O\big((m+n)^k\big)$, for some constant k. (Note that if $\mathbf{x} = \mathbf{0}$ is not an optimal primal solution, the dual has no feasible solution, and we assume that in this case our algorithm exits with an error message.) Assuming the availability of the above algorithm, construct a new algorithm that takes as input a system of m linear inequalities in n variables, runs for $O\big((m+n)^k\big)$ time, and either finds a feasible solution or determines that no feasible solution exists.

Exercise 4.18 Consider a problem in standard form. Suppose that the matrix $\mathbf{A}$ has dimensions $m \times n$ and its rows are linearly independent. Suppose that all basic solutions to the primal and to the dual are nondegenerate. Let $\mathbf{x}$ be a feasible solution to the primal and let $\mathbf{p}$ be a dual vector (not necessarily feasible), such that the pair $(\mathbf{x}, \mathbf{p})$ satisfies complementary slackness.

(a) Show that there exist m columns of $\mathbf{A}$ that are linearly independent and such that the corresponding components of $\mathbf{x}$ are all positive.

(b) Show that $\mathbf{x}$ and $\mathbf{p}$ are basic solutions to the primal and the dual, respectively.

(c) Show that the result of part (a) is false if the nondegeneracy assumption is removed.

Exercise 4.19 Let $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ be a nonempty polyhedron, and let m be the dimension of the vector $\mathbf{b}$. We call x_j a *null variable* if $x_j = 0$ whenever $\mathbf{x} \in P$.

(a) Suppose that there exists some $\mathbf{p} \in \Re^m$ for which $\mathbf{p}'\mathbf{A} \geq \mathbf{0}'$, $\mathbf{p}'\mathbf{b} = 0$, and such that the jth component of $\mathbf{p}'\mathbf{A}$ is positive. Prove that x_j is a null variable.

(b) Prove the converse of (a): if x_j is a null variable, then there exists some $\mathbf{p} \in \Re^m$ with the properties stated in part (a).

(c) If x_j is not a null variable, then by definition, there exists some $\mathbf{y} \in P$ for which $y_j > 0$. Use the results in parts (a) and (b) to prove that there exist $\mathbf{x} \in P$ and $\mathbf{p} \in \Re^m$ such that:

$$\mathbf{p}'\mathbf{A} \geq \mathbf{0}', \qquad \mathbf{p}'\mathbf{b} = 0, \qquad \mathbf{x} + \mathbf{A}'\mathbf{p} > \mathbf{0}.$$

Exercise 4.20* (Strict complementary slackness)

(a) Consider the following linear programming problem and its dual

$$\begin{array}{lll} \text{minimize} & \mathbf{c}'\mathbf{x} & \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} & \\ & \mathbf{x} \geq \mathbf{0}, & \end{array} \qquad \begin{array}{lll} \text{maximize} & \mathbf{p}'\mathbf{b} & \\ \text{subject to} & \mathbf{p}'\mathbf{A} \leq \mathbf{c}', & \end{array}$$

and assume that both problems have an optimal solution. Fix some j. Suppose that every optimal solution to the primal satisfies $x_j = 0$. Show that there exists an optimal solution $\mathbf{p}$ to the dual such that $\mathbf{p}'\mathbf{A}_j < c_j$. (Here, $\mathbf{A}_j$ is the jth column of $\mathbf{A}$.) *Hint:* Let d be the optimal cost. Consider the problem of minimizing $-x_j$ subject to $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, and $-\mathbf{c}'\mathbf{x} \geq -d$, and form its dual.

(b) Show that there exist optimal solutions $\mathbf{x}$ and $\mathbf{p}$ to the primal and to the dual, respectively, such that for every j we have either $x_j > 0$ or $\mathbf{p}'\mathbf{A}_j < c_j$. *Hint:* Use part (a) for each j, and then take the average of the vectors obtained.

(c) Consider now the following linear programming problem and its dual:

$$\begin{array}{lll} \text{minimize} & \mathbf{c}'\mathbf{x} & \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b} & \\ & \mathbf{x} \geq \mathbf{0}, & \end{array} \qquad \begin{array}{lll} \text{maximize} & \mathbf{p}'\mathbf{b} & \\ \text{subject to} & \mathbf{p}'\mathbf{A} \leq \mathbf{c}' & \\ & \mathbf{p} \geq \mathbf{0}. & \end{array}$$

Assume that both problems have an optimal solution. Show that there exist optimal solutions to the primal and to the dual, respectively, that satisfy *strict complementary slackness*, that is:

(i) For every j we have either $x_j > 0$ or $\mathbf{p}'\mathbf{A}_j < c_j$.

(ii) For every i, we have either $\mathbf{a}_i'\mathbf{x} > b_i$ or $p_i > 0$. (Here, $\mathbf{a}_i'$ is the ith row of $\mathbf{A}$.) *Hint:* Convert the primal to standard form and apply part (b).

(d) Consider the linear programming problem

$$\begin{array}{rl} \text{minimize} & 5x_1 + 5x_2 \\ \text{subject to} & x_1 + x_2 \geq 2 \\ & 2x_1 - x_2 \geq 0 \\ & x_1, x_2 \geq 0. \end{array}$$

Does the optimal primal solution $(2/3, 4/3)$, together with the corresponding dual optimal solution, satisfy strict complementary slackness? Determine all primal and dual optimal solutions and identify the set of *all* strictly complementary pairs.

Exercise 4.21* (Clark's theorem) Consider the following pair of linear programming problems:

$$\begin{array}{rl} \text{minimize} & \mathbf{c}'\mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{array} \qquad \begin{array}{rl} \text{maximize} & \mathbf{p}'\mathbf{b} \\ \text{subject to} & \mathbf{p}'\mathbf{A} \leq \mathbf{c}' \\ & \mathbf{p} \geq \mathbf{0}. \end{array}$$

Suppose that at least one of these two problems has a feasible solution. Prove that the set of feasible solutions to at least one of the two problems is unbounded. *Hint:* Interpret boundedness of a set in terms of the finiteness of the optimal cost of some linear programming problem.

Exercise 4.22 Consider the dual simplex method applied to a standard form problem with linearly independent rows. Suppose that we have a basis which is primal infeasible, but dual feasible, and let i be such that $x_{B(i)} < 0$. Suppose that all entries in the ith row in the tableau (other than $x_{B(i)}$) are nonnegative. Show that the optimal dual cost is $+\infty$.

Exercise 4.23 Describe in detail the mechanics of a revised dual simplex method that works in terms of the inverse basis matrix $\mathbf{B}^{-1}$ instead of the full simplex tableau.

Exercise 4.24 Consider the lexicographic pivoting rule for the dual simplex method and suppose that the algorithm is initialized with each column of the tableau being lexicographically positive. Prove that the dual simplex method does not cycle.

Exercise 4.25 This exercise shows that if we bring the dual problem into standard form and then apply the primal simplex method, the resulting algorithm is not identical to the dual simplex method.

Consider the following standard form problem and its dual.

$$\begin{array}{rl} \text{minimize} & x_1 + x_2 \\ \text{subject to} & x_1 = 1 \\ & x_2 = 1 \\ & x_1, x_2 \geq 0 \end{array} \qquad \begin{array}{rl} \text{maximize} & p_1 + p_2 \\ \text{subject to} & p_1 \leq 1 \\ & p_2 \leq 1. \end{array}$$

Here, there is only one possible basis and the dual simplex method must terminate immediately. Show that if the dual problem is converted into standard form and the primal simplex method is applied to it, one or more changes of basis may be required.

Exercise 4.26 Let $\mathbf{A}$ be a given matrix. Show that exactly one of the following alternatives must hold.

(a) There exists some $\mathbf{x} \neq \mathbf{0}$ such that $\mathbf{Ax} = \mathbf{0}$, $\mathbf{x} \geq \mathbf{0}$.

(b) There exists some $\mathbf{p}$ such that $\mathbf{p}'\mathbf{A} > \mathbf{0}'$.

Exercise 4.27 Let $\mathbf{A}$ be a given matrix. Show that the following two statements are equivalent.

(a) Every vector such that $\mathbf{Ax} \geq \mathbf{0}$ and $\mathbf{x} \geq \mathbf{0}$ must satisfy $x_1 = 0$.

(b) There exists some $\mathbf{p}$ such that $\mathbf{p}'\mathbf{A} \leq \mathbf{0}$, $\mathbf{p} \geq \mathbf{0}$, and $\mathbf{p}'\mathbf{A}_1 < 0$, where $\mathbf{A}_1$ is the first column of $\mathbf{A}$.

Exercise 4.28 Let $\mathbf{a}$ and $\mathbf{a}_1, \ldots, \mathbf{a}_m$ be given vectors in $\Re^n$. Prove that the following two statements are equivalent:

(a) For all $\mathbf{x} \geq \mathbf{0}$, we have $\mathbf{a}'\mathbf{x} \leq \max_i \mathbf{a}_i'\mathbf{x}$.

(b) There exist nonnegative coefficients λ_i that sum to 1 and such that $\mathbf{a} \leq \sum_{i=1}^m \lambda_i \mathbf{a}_i$.

Exercise 4.29 (Inconsistent systems of linear inequalities) Let $\mathbf{a}_1, \ldots, \mathbf{a}_m$ be some vectors in $\Re^n$, with $m > n+1$. Suppose that the system of inequalities $\mathbf{a}_i'\mathbf{x} \geq b_i$, $i = 1, \ldots, m$, does not have any solutions. Show that we can choose $n+1$ of these inequalities, so that the resulting system of inequalities has no solutions.

Exercise 4.30 (Helly's theorem)

(a) Let $\mathcal{F}$ be a finite family of polyhedra in $\Re^n$ such that every $n+1$ polyhedra in $\mathcal{F}$ have a point in common. Prove that all polyhedra in $\mathcal{F}$ have a point in common. *Hint:* Use the result in Exercise 4.29.

(b) For $n = 2$, part (a) asserts that the polyhedra $P_1, P_2, \ldots, P_K$ $(K \geq 3)$ in the plane have a point in common if and only if every three of them have a point in common. Is the result still true with "three" replaced by "two"?

Exercise 4.31 (Unit eigenvectors of stochastic matrices) We say that an $n \times n$ matrix $\mathbf{P}$, with entries p_{ij}, is *stochastic* if all of its entries are nonnegative and

$$\sum_{j=1}^{n} p_{ij} = 1, \qquad \forall\, i,$$

that is, the sum of the entries of each row is equal to 1.

Use duality to show that if $\mathbf{P}$ is a stochastic matrix, then the system of equations

$$\mathbf{p}'\mathbf{P} = \mathbf{p}', \qquad \mathbf{p} \geq \mathbf{0},$$

has a nonzero solution. (Note that the vector $\mathbf{p}$ can be normalized so that its components sum to one. Then, the result in this exercise establishes that every finite state Markov chain has an invariant probability distribution.)

Exercise 4.32 * (Leontief systems and Samuelson's substitution theorem) A *Leontief matrix* is an $m \times n$ matrix $\mathbf{A}$ in which every column has at most one positive element. For an interpretation, each column $\mathbf{A}_j$ corresponds to a production process. If a_{ij} is negative, $|a_{ij}|$ represents the amount of goods of type i consumed by the process. If a_{ij} is positive, it represents the amount of goods of type i produced by the process. If x_j is the intensity with which process j is used, then $\mathbf{Ax}$ represents the net output of the different goods. The matrix $\mathbf{A}$ is called *productive* if there exists some $\mathbf{x} \geq \mathbf{0}$ such that $\mathbf{Ax} > \mathbf{0}$.

(a) Let $\mathbf{A}$ be a square productive Leontief matrix ($m = n$). Show that every vector $\mathbf{z}$ that satisfies $\mathbf{Az} \geq \mathbf{0}$ must be nonnegative. *Hint:* If $\mathbf{z}$ satisfies $\mathbf{Az} \geq \mathbf{0}$ but has a negative component, consider the smallest nonnegative θ such that some component of $\mathbf{x} + \theta\mathbf{z}$ becomes zero, and derive a contradiction.

(b) Show that every square productive Leontief matrix is invertible and that all entries of the inverse matrix are nonnegative. *Hint:* Use the result in part (a).

(c) We now consider the general case where $n \geq m$, and we introduce a constraint of the form $\mathbf{e}'\mathbf{x} \leq 1$, where $\mathbf{e} = (1, \ldots, 1)$. (Such a constraint could capture, for example, a bottleneck due to the finiteness of the labor force.) An "output" vector $\mathbf{y} \in \Re^m$ is said to be *achievable* if $\mathbf{y} \geq \mathbf{0}$ and there exists some $\mathbf{x} \geq \mathbf{0}$ such that $\mathbf{Ax} = \mathbf{y}$ and $\mathbf{e}'\mathbf{y} \leq 1$. An achievable vector $\mathbf{y}$ is said to be *efficient* if there exists no achievable vector $\mathbf{z}$ such that $\mathbf{z} \geq \mathbf{y}$ and $\mathbf{z} \neq \mathbf{y}$. (Intuitively, an output vector $\mathbf{y}$ which is not efficient can be improved upon and is therefore uninteresting.) Suppose that $\mathbf{A}$ is productive. Show that there exists a positive efficient vector $\mathbf{y}$. *Hint:* Given a positive achievable vector $\mathbf{y}^*$, consider maximizing $\sum_i y_i$ over all achievable vectors $\mathbf{y}$ that are larger than $\mathbf{y}^*$.

(d) Suppose that $\mathbf{A}$ is productive. Show that there exists a set of m production processes that are capable of generating all possible efficient output vectors $\mathbf{y}$. That is, there exist indices $B(1), \ldots, B(m)$, such that every efficient output vector $\mathbf{y}$ can be expressed in the form $\mathbf{y} = \sum_{i=1}^{m} \mathbf{A}_{B(i)} x_{B(i)}$, for some nonnegative coefficients $x_{B(i)}$ whose sum is bounded by 1. *Hint:* Consider the problem of minimizing $\mathbf{e}'\mathbf{x}$ subject to $\mathbf{Ax} = \mathbf{y}$, $\mathbf{x} \geq \mathbf{0}$, and show that we can use the same optimal basis for all efficient vectors $\mathbf{y}$.

Exercise 4.33 (Options pricing) Consider a market that operates for a single period, and which involves three assets: a stock, a bond, and an option. Let S be the price of the stock, in the beginning of the period. Its price $\overline{S}$ at the end of the period is random and is assumed to be equal to either Su, with probability β, or Sd, with probability $1 - \beta$. Here u and d are scalars that satisfy $d < 1 < u$. Bonds are assumed riskless. Investing one dollar in a bond results in a payoff of r, at the end of the period. (Here, r is a scalar greater than 1.) Finally, the option gives us the right to purchase, at the end of the period, one stock at a fixed price of K. If the realized price $\overline{S}$ of the stock is greater than K, we exercise the option and then immediately sell the stock in the stock market, for a payoff of $\overline{S} - K$. If on the other hand we have $\overline{S} < K$, there is no advantage in exercising the option, and we receive zero payoff. Thus, the value of the option at the end of the period is equal to $\max\{0, \overline{S} - K\}$. Since the option is itself an asset, it

should have a value in the beginning of the time period. Show that under the absence of arbitrage condition, the value of the option must be equal to

$$\gamma \max\{0, Su - K\} + \delta \max\{0, Sd - K\},$$

where γ and δ are a solution to the following system of linear equations:

$$\begin{aligned} u\gamma + d\delta &= 1 \\ \gamma + \delta &= \frac{1}{r}. \end{aligned}$$

Hint: Write down the payoff matrix $\mathbf{R}$ and use Theorem 4.8.

Exercise 4.34 (Finding separating hyperplanes) Consider a polyhedron P that has at least one extreme point.

(a) Suppose that we are given the extreme points $\mathbf{x}^i$ and a complete set of extreme rays $\mathbf{w}^j$ of P. Create a linear programming problem whose solution provides us with a separating hyperplane that separates P from the origin, or allows us to conclude that none exists.

(b) Suppose now that P is given to us in the form $P = \{\mathbf{x} \mid \mathbf{a}_i'\mathbf{x} \geq b_i,\ i = 1, \ldots, m\}$. Suppose that $\mathbf{0} \notin P$. Explain how a separating hyperplane can be found.

Exercise 4.35 (Separation of disjoint polyhedra) Consider two nonempty polyhedra $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ and $Q = \{\mathbf{x} \in \Re^n \mid \mathbf{D}\mathbf{x} \leq \mathbf{d}\}$. We are interested in finding out whether the two polyhedra have a point in common.

(a) Devise a linear programming problem such that: if $P \cap Q$ is nonempty, it returns a point in $P \cap Q$; if $P \cap Q$ is empty, the linear programming problem is infeasible.

(b) Suppose that $P \cap Q$ is empty. Use the dual of the problem you have constructed in part (a) to show that there exists a vector $\mathbf{c}$ such that $\mathbf{c}'\mathbf{x} < \mathbf{c}'\mathbf{y}$ for all $\mathbf{x} \in P$ and $\mathbf{y} \in Q$.

Exercise 4.36 (Containment of polyhedra)

(a) Let P and Q be two polyhedra in $\Re^n$ described in terms of linear inequality constraints. Devise an algorithm that decides whether P is a subset of Q.

(b) Repeat part (a) if the polyhedra are described in terms of their extreme points and extreme rays.

Exercise 4.37 (Closedness of finitely generated cones) Let $\mathbf{A}_1, \ldots, \mathbf{A}_n$ be given vectors in $\Re^m$. Consider the cone $C = \left\{ \sum_{i=1}^n \mathbf{A}_i x_i \mid x_i \geq 0 \right\}$ and let $\mathbf{y}^k$, $k = 1, 2, \ldots$, be a sequence of elements of C that converges to some $\mathbf{y}$. Show that $\mathbf{y} \in C$ (and hence C is closed), using the following argument. With $\mathbf{y}$ fixed as above, consider the problem of minimizing $\|\mathbf{y} - \sum_{i=1}^n \mathbf{A}_i x_i\|_\infty$, subject to the constraints $x_1, \ldots, x_n \geq 0$. Here $\|\cdot\|_\infty$ stands for the maximum norm, defined by $\|\mathbf{x}\|_\infty = \max_i |x_i|$. Explain why the above minimization problem has an optimal solution, find the value of the optimal cost, and prove that $\mathbf{y} \in C$.

Exercise 4.38 (From Farkas' lemma to duality) Use Farkas' lemma to prove the duality theorem for a linear programming problem involving constraints of the form $\mathbf{a}_i'\mathbf{x} = b_i$, $\mathbf{a}_i'\mathbf{x} \geq b_i$, and nonnegativity constraints for some of the variables x_j. *Hint:* Start by deriving the form of the set of feasible directions at an optimal solution.

Exercise 4.39 (Extreme rays of cones) Let us define a nonzero element $\mathbf{d}$ of a pointed polyhedral cone C to be an *extreme ray* if it has the following property: if there exist vectors $\mathbf{f} \in C$ and $\mathbf{g} \in C$ and some $\lambda \in (0,1)$ satisfying $\mathbf{d} = \mathbf{f} + \mathbf{g}$, then both $\mathbf{f}$ and $\mathbf{g}$ are scalar multiples of $\mathbf{d}$. Prove that this definition of extreme rays is equivalent to Definition 4.2.

Exercise 4.40 (Extreme rays of a cone are extreme points of its sections) Consider the cone $C = \{\mathbf{x} \in \Re^n \mid \mathbf{a}_i'\mathbf{x} \geq 0,\ i = 1, \ldots, m\}$ and assume that the first n constraint vectors $\mathbf{a}_1, \ldots, \mathbf{a}_n$ are linearly independent. For any nonnegative scalar r, we define the polyhedron P_r by

$$P_r = \left\{ \mathbf{x} \in C \;\middle|\; \sum_{i=1}^{n} \mathbf{a}_i'\mathbf{x} = r \right\}.$$

(a) Show that the polyhedron P_r is bounded for every $r \geq 0$.

(b) Let $r > 0$. Show that a vector $\mathbf{x} \in P_r$ is an extreme point of P_r if and only if $\mathbf{x}$ is an extreme ray of the cone C.

Exercise 4.41 (Carathéodory's theorem) Show that every element $\mathbf{x}$ of a bounded polyhedron $P \subset \Re^n$ can be expressed as a convex combination of at most $n+1$ extreme points of P. *Hint:* Consider an extreme point of the set of all possible representations of $\mathbf{x}$.

Exercise 4.42 (Problems with side constraints) Consider the linear programming problem of minimizing $\mathbf{c}'\mathbf{x}$ over a bounded polyhedron $P \subset \Re^n$ and subject to additional constraints $\mathbf{a}_i'\mathbf{x} = b_i$, $i = 1, \ldots, L$. Assume that the problem has a feasible solution. Show that there exists an optimal solution which is a convex combination of $L+1$ extreme points of P. *Hint:* Use the resolution theorem to represent P.

Exercise 4.43

(a) Consider the minimization of $c_1x_1 + c_2x_2$ subject to the constraints

$$x_2 - 3 \leq x_1 \leq 2x_2 + 2, \qquad x_1, x_2 \geq 0.$$

Find necessary and sufficient conditions on (c_1, c_2) for the optimal cost to be finite.

(b) For a general feasible linear programming problem, consider the set of all cost vectors for which the optimal cost is finite. Is it a polyhedron? Prove your answer.

Exercise 4.44

(a) Let $P = \{(x_1, x_2) \mid x_1 - x_2 = 0,\ x_1 + x_2 = 0\}$. What are the extreme points and the extreme rays of P?

(b) Let $P = \{(x_1, x_2) \mid 4x_1 + 2x_2 \geq 8,\ 2x_1 + x_2 \leq 8\}$. What are the extreme points and the extreme rays of P?

(c) For the polyhedron of part (b), is it possible to express each one of its elements as a convex combination of its extreme points plus a nonnegative linear combination of its extreme rays? Is this compatible with the resolution theorem?

Exercise 4.45 Let P be a polyhedron with at least one extreme point. Is it possible to express an arbitrary element of P as a convex combination of its extreme points plus a nonnegative multiple of a single extreme ray?

Exercise 4.46 (Resolution theorem for polyhedral cones) Let C be a nonempty polyhedral cone.

(a) Show that C can be expressed as the union of a finite number $C_1, \ldots, C_k$ of pointed polyhedral cones. *Hint:* Intersect with orthants.

(b) Show that an extreme ray of C must be an extreme ray of one of the cones $C_1, \ldots, C_k$.

(c) Show that there exists a finite number of elements $\mathbf{w}^1, \ldots, \mathbf{w}^r$ of C such that

$$C = \left\{ \sum_{i=1}^{r} \theta_i \mathbf{w}^i \;\middle|\; \theta_1, \ldots, \theta_r \geq 0 \right\}.$$

Exercise 4.47 (Resolution theorem for general polyhedra) Let P be a polyhedron. Show that there exist vectors $\mathbf{x}^1, \ldots, \mathbf{x}^k$ and $\mathbf{w}^1, \ldots, \mathbf{w}^r$ such that

$$P = \left\{ \sum_{i=1}^{k} \lambda_i \mathbf{x}^i + \sum_{j=1}^{r} \theta_j \mathbf{w}^j \;\middle|\; \lambda_i \geq 0,\ \theta_j \geq 0,\ \sum_{i=1}^{k} \lambda_i = 1 \right\}.$$

Hint: Generalize the steps in the preceding exercise.

Exercise 4.48 * (Polar, finitely generated, and polyhedral cones) For any cone C, we define its *polar* $C^\perp$ by

$$C^\perp = \{\mathbf{p} \mid \mathbf{p}'\mathbf{x} \leq 0, \text{ for all } \mathbf{x} \in C\}.$$

(a) Let F be a finitely generated cone, of the form

$$F = \left\{ \sum_{i=1}^{r} \theta_i \mathbf{w}^i \;\middle|\; \theta_1, \ldots, \theta_r \geq 0 \right\}.$$

Show that $F^\perp = \{\mathbf{p} \mid \mathbf{p}'\mathbf{w}^i \leq 0,\ i = 1, \ldots, r\}$, which is a polyhedral cone.

(b) Show that the polar of $F^\perp$ is F and conclude that the polar of a polyhedral cone is finitely generated. *Hint:* Use Farkas' lemma.

(c) Show that a finitely generated pointed cone F is a polyhedron. *Hint:* Consider the polar of the polar.

(d) **(Polar cone theorem)** Let C be a closed, nonempty, and convex cone. Show that $(C^{\perp})^{\perp} = C$. *Hint:* Mimic the derivation of Farkas' lemma using the separating hyperplane theorem (Section 4.7).

(e) Is the polar cone theorem true when C is the empty set?

Exercise 4.49 Consider a polyhedron, and let $\mathbf{x}$, $\mathbf{y}$ be two basic feasible solutions. If we are only allowed to make moves from any basic feasible solution to an adjacent one, show that we can go from $\mathbf{x}$ to $\mathbf{y}$ in a finite number of steps. *Hint:* Generalize the simplex method to nonstandard form problems: starting from a nonoptimal basic feasible solution, move along an extreme ray of the cone of feasible directions.

Exercise 4.50 We are interested in the problem of deciding whether a polyhedron

$$Q = \left\{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b},\ \mathbf{D}\mathbf{x} \geq \mathbf{d},\ \mathbf{x} \geq \mathbf{0}\right\}$$

is nonempty. We assume that the polyhedron $P = \{\mathbf{x} \in \Re^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$ is nonempty and bounded. For any vector $\mathbf{p}$, of the same dimension as $\mathbf{d}$, we define

$$g(\mathbf{p}) = -\mathbf{p}'\mathbf{d} + \max_{\mathbf{x} \in P} \mathbf{p}'\mathbf{D}\mathbf{x}.$$

(a) Show that if Q is nonempty, then $g(\mathbf{p}) \geq 0$ for all $\mathbf{p} \geq \mathbf{0}$.

(b) Show that if Q is empty, then there exists some $\mathbf{p} \geq \mathbf{0}$, such that $g(\mathbf{p}) < 0$.

(c) If Q is empty, what is the minimum of $g(\mathbf{p})$ over all $\mathbf{p} \geq \mathbf{0}$?

4.13 Notes and sources

4.3. The duality theorem is due to von Neumann (1947), and Gale, Kuhn, and Tucker (1951).

4.6. Farkas' lemma is due to Farkas (1894) and Minkowski (1896). See Schrijver (1986) for a comprehensive presentation of related results. The connection between duality theory and arbitrage was developed by Ross (1976, 1978).

4.7. Weierstrass' Theorem and its proof can be found in most texts on real analysis; see, for example, Rudin (1976). While the simplex method is only relevant to linear programming problems with a finite number of variables, the approach based on the separating hyperplane theorem leads to a generalization of duality theory that covers more general convex optimization problems, as well as infinite-dimensional linear programming problems, that is, linear programming problems with infinitely many variables and constraints; see, e.g., Luenberger (1969) and Rockafellar (1970).

4.9. The resolution theorem and its converse are usually attributed to Farkas, Minkowski, and Weyl.

4.10. For extensions of duality theory to problems involving general convex functions and constraint sets, see Rockafellar (1970) and Bertsekas (1995b).

4.12 Exercises 4.6 and 4.7 are adapted from Boyd and Vandenberghe (1995). The result on strict complementary slackness (Exercise 4.20) was proved by Tucker (1956). The result in Exercise 4.21 is due to Clark (1961). The result in Exercise 4.30 is due to Helly (1923). Input-output macroeconomic models of the form considered in Exercise 4.32, have been introduced by Leontief, who was awarded the 1973 Nobel prize in economics. The result in Exercise 4.41 is due to Carathéodory (1907).

Chapter 5

Sensitivity analysis

Contents

Consider the standard form problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

and its dual

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} \leq \mathbf{c}'. \end{aligned}$$

In this chapter, we study the dependence of the optimal cost and the optimal solution on the coefficient matrix $\mathbf{A}$, the requirement vector $\mathbf{b}$, and the cost vector $\mathbf{c}$. This is an important issue in practice because we often have incomplete knowledge of the problem data and we may wish to predict the effects of certain parameter changes.

In the first section of this chapter, we develop conditions under which the optimal basis remains the same despite a change in the problem data, and we examine the consequences on the optimal cost. We also discuss how to obtain an optimal solution if we add or delete some constraints. In subsequent sections, we allow larger changes in the problem data, resulting in a new optimal basis, and we develop a global perspective of the dependence of the optimal cost on the vectors $\mathbf{b}$ and $\mathbf{c}$. The chapter ends with a brief discussion of parametric programming, which is an extension of the simplex method tailored to the case where there is a single scalar unknown parameter.

Many of the results in this chapter can be extended to cover general linear programming problems. Nevertheless, and in order to simplify the presentation, our standing assumption throughout this chapter will be that we are dealing with a standard form problem and that the rows of the $m \times n$ matrix $\mathbf{A}$ are linearly independent.

5.1 Local sensitivity analysis

In this section, we develop a methodology for performing sensitivity analysis. We consider a linear programming problem, and we assume that we already have an optimal basis $\mathbf{B}$ and the associated optimal solution $\mathbf{x}^*$. We then assume that some entry of $\mathbf{A}$, $\mathbf{b}$, or $\mathbf{c}$ has been changed, or that a new constraint is added, or that a new variable is added. We first look for conditions under which the current basis is still optimal. If these conditions are violated, we look for an algorithm that finds a new optimal solution without having to solve the new problem from scratch. We will see that the simplex method can be quite useful in this respect.

Having assumed that $\mathbf{B}$ is an optimal basis for the original problem, the following two conditions are satisfied:

$$\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}, \qquad \text{(feasibility)}$$

$$\mathbf{c}' - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{0}', \qquad \text{(optimality)}.$$

When the problem is changed, we check to see how these conditions are affected. By insisting that both conditions (feasibility and optimality) hold for the modified problem, we obtain the conditions under which the basis matrix $\mathbf{B}$ remains optimal for the modified problem. In what follows, we apply this approach to several examples.

A new variable is added

Suppose that we introduce a new variable x_{n+1}, together with a corresponding column $\mathbf{A}_{n+1}$, and obtain the new problem

$$\begin{array}{lll} \text{minimize} & \mathbf{c}'\mathbf{x} + c_{n+1}x_{n+1} & \\ \text{subject to} & \mathbf{A}\mathbf{x} + \mathbf{A}_{n+1}x_{n+1} = \mathbf{b} & \\ & \mathbf{x} \geq \mathbf{0}. & \end{array}$$

We wish to determine whether the current basis $\mathbf{B}$ is still optimal.

We note that $(\mathbf{x}, x_{n+1}) = (\mathbf{x}^*, 0)$ is a basic feasible solution to the new problem associated with the basis $\mathbf{B}$, and we only need to examine the optimality conditions. For the basis $\mathbf{B}$ to remain optimal, it is necessary and sufficient that the reduced cost of x_{n+1} be nonnegative, that is,

$$\bar{c}_{n+1} = c_{n+1} - \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A}_{n+1} \geq 0.$$

If this condition is satisfied, $(\mathbf{x}^*, 0)$ is an optimal solution to the new problem. If, however, $\bar{c}_{n+1} < 0$, then $(\mathbf{x}^*, 0)$ is not necessarily optimal. In order to find an optimal solution, we add a column to the simplex tableau, associated with the new variable, and apply the primal simplex algorithm starting from the current basis $\mathbf{B}$. Typically, an optimal solution to the new problem is obtained with a small number of iterations, and this approach is usually much faster than solving the new problem from scratch.

Example 5.1 Consider the problem

$$\begin{array}{lrcrcrcl} \text{minimize} & -5x_1 & - & x_2 & + & 12x_3 & & \\ \text{subject to} & 3x_1 & + & 2x_2 & + & x_3 & & = 10 \\ & 5x_1 & + & 3x_2 & & & + x_4 & = 16 \\ & x_1, \ldots, x_4 \geq 0. & & & & & & \end{array}$$

An optimal solution to this problem is given by $\mathbf{x} = (2, 2, 0, 0)$ and the corresponding simplex tableau is given by

		x_1	x_2	x_3	x_4
	12	0	0	2	7
$x_1 =$	2	1	0	−3	2
$x_2 =$	2	0	1	5	−3

Note that $\mathbf{B}^{-1}$ is given by the last two columns of the tableau.

Let us now introduce a variable x_5 and consider the new problem

$$\begin{array}{lrcrcrcrcl} \text{minimize} & -5x_1 & - & x_2 & + & 12x_3 & & - & x_5 & \\ \text{subject to} & 3x_1 & + & 2x_2 & + & x_3 & & + & x_5 & = 10 \\ & 5x_1 & + & 3x_2 & & & + & x_4 & + x_5 & = 16 \\ & x_1, \ldots, x_5 \geq 0. & & & & & & & & \end{array}$$

We have $\mathbf{A}_5 = (1, 1)$ and

$$\bar{c}_5 = c_5 - \mathbf{c}_B' \mathbf{B}^{-1} \mathbf{A}_5 = -1 - \begin{bmatrix} -5 & -1 \end{bmatrix} \begin{bmatrix} -3 & 2 \\ 5 & -3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = -4.$$

Since $\bar{c}_5$ is negative, introducing the new variable to the basis can be beneficial. We observe that $\mathbf{B}^{-1}\mathbf{A}_5 = (-1, 2)$ and augment the tableau by introducing a column associated with x_5:

		x_1	x_2	x_3	x_4	x_5
	12	0	0	2	7	−4
$x_1 =$	2	1	0	−3	2	−1
$x_2 =$	2	0	1	5	−3	2

We then bring x_5 into the basis; x_2 exits and we obtain the following tableau, which happens to be optimal:

		x_1	x_2	x_3	x_4	x_5
	16	0	2	12	1	0
$x_1 =$	3	1	0.5	−0.5	0.5	0
$x_5 =$	1	0	0.5	2.5	−1.5	1

An optimal solution is given by $\mathbf{x} = (3, 0, 0, 0, 1)$.

A new inequality constraint is added

Let us now introduce a new constraint $\mathbf{a}_{m+1}'\mathbf{x} \geq b_{m+1}$, where $\mathbf{a}_{m+1}$ and b_{m+1} are given. If the optimal solution $\mathbf{x}^*$ to the original problem satisfies this constraint, then $\mathbf{x}^*$ is an optimal solution to the new problem as well. If the new constraint is violated, we introduce a nonnegative slack variable x_{n+1}, and rewrite the new constraint in the form $\mathbf{a}_{m+1}'\mathbf{x} - x_{n+1} = b_{m+1}$. We obtain a problem in standard form, in which the matrix $\mathbf{A}$ is replaced by

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{a}_{m+1}' & -1 \end{bmatrix}.$$

Let $\mathbf{B}$ be an optimal basis for the original problem. We form a basis for the new problem by selecting the original basic variables together with x_{n+1}. The new basis matrix $\overline{\mathbf{B}}$ is of the form

$$\overline{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & \mathbf{0} \\ \mathbf{a}' & -1 \end{bmatrix},$$

where the row vector $\mathbf{a}'$ contains those components of $\mathbf{a}'_{m+1}$ associated with the original basic columns. (The determinant of this matrix is the negative of the determinant of $\mathbf{B}$, hence nonzero, and we therefore have a true basis matrix.) The basic solution associated with this basis is $(\mathbf{x}^*, \mathbf{a}'_{m+1}\mathbf{x}^* - b_{m+1})$, and is infeasible because of our assumption that $\mathbf{x}^*$ violates the new constraint. Note that the new inverse basis matrix is readily available because

$$\overline{\mathbf{B}}^{-1} = \begin{bmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{a}'\mathbf{B}^{-1} & -1 \end{bmatrix}.$$

(To see this, note that the product $\overline{\mathbf{B}}^{-1}\overline{\mathbf{B}}$ is equal to the identity matrix.)

Let $\mathbf{c}_B$ be the m-dimensional vector with the costs of the basic variables in the original problem. Then, the vector of reduced costs associated with the basis $\overline{\mathbf{B}}$ for the new problem, is given by

$$\begin{bmatrix} \mathbf{c}' & 0 \end{bmatrix} - \begin{bmatrix} \mathbf{c}'_B & 0 \end{bmatrix} \begin{bmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{a}'\mathbf{B}^{-1} & -1 \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{a}'_{m+1} & -1 \end{bmatrix} = \begin{bmatrix} \mathbf{c}' - \mathbf{c}'_B\mathbf{B}^{-1}\mathbf{A} & 0 \end{bmatrix},$$

and is nonnegative due to the optimality of $\mathbf{B}$ for the original problem. Hence, $\overline{\mathbf{B}}$ is a dual feasible basis and we are in a position to apply the dual simplex method to the new problem. Note that an initial simplex tableau for the new problem is readily constructed. For example, we have

$$\overline{\mathbf{B}}^{-1} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{a}'_{m+1} & -1 \end{bmatrix} = \begin{bmatrix} \mathbf{B}^{-1}\mathbf{A} & \mathbf{0} \\ \mathbf{a}'\mathbf{B}^{-1}\mathbf{A} - \mathbf{a}'_{m+1} & 1 \end{bmatrix},$$

where $\mathbf{B}^{-1}\mathbf{A}$ is available from the final simplex tableau for the original problem.

Example 5.2 Consider again the problem in Example 5.1:

$$\begin{array}{lrcrcrcl} \text{minimize} & -5x_1 & - & x_2 & + & 12x_3 & & \\ \text{subject to} & 3x_1 & + & 2x_2 & + & x_3 & = & 10 \\ & 5x_1 & + & 3x_2 & & & + x_4 = & 16 \\ & x_1, \ldots, x_4 & \geq & 0, & & & & \end{array}$$

and recall the optimal simplex tableau:

		x_1	x_2	x_3	x_4
	12	0	0	2	7
$x_1 =$	2	1	0	-3	2
$x_2 =$	2	0	1	5	-3

We introduce the additional constraint $x_1 + x_2 \geq 5$, which is violated by the optimal solution $\mathbf{x}^* = (2, 2, 0, 0)$. We have $\mathbf{a}_{m+1} = (1, 1, 0, 0)$, $b_{m+1} = 5$, and $\mathbf{a}'_{m+1}\mathbf{x}^* < b_{m+1}$. We form the standard form problem

$$\begin{array}{lrcrcrcrcl} \text{minimize} & -5x_1 & - & x_2 & + & 12x_3 & & & & \\ \text{subject to} & 3x_1 & + & 2x_2 & + & x_3 & & & = & 10 \\ & 5x_1 & + & 3x_2 & & & + & x_4 & = & 16 \\ & x_1 & + & x_2 & & & & - \ x_5 & = & 5 \\ & x_1, \ldots, x_5 \geq 0. & & & & & & & & \end{array}$$

Let $\mathbf{a}$ consist of the components of $\mathbf{a}_{m+1}$ associated with the basic variables. We then have $\mathbf{a} = (1, 1)$ and

$$\mathbf{a}'\mathbf{B}^{-1}\mathbf{A} - \mathbf{a}'_{m+1} = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & 1 & 5 & -3 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 2 & -1 \end{bmatrix}.$$

The tableau for the new problem is of the form

		x_1	x_2	x_3	x_4	x_5
	12	0	0	2	7	0
$x_1 =$	2	1	0	−3	2	0
$x_2 =$	2	0	1	5	−3	0
$x_5 =$	−1	0	0	2	−1	1

We now have all the information necessary to apply the dual simplex method to the new problem.

Our discussion has been focused on the case where an inequality constraint is added to the primal problem. Suppose now that we introduce a new constraint $\mathbf{p}'\mathbf{A}_{n+1} \leq c_{n+1}$ in the dual. This is equivalent to introducing a new variable in the primal, and we are back to the case that was considered in the preceding subsection.

A new equality constraint is added

We now consider the case where the new constraint is of the form $\mathbf{a}'_{m+1}\mathbf{x} = b_{m+1}$, and we assume that this new constraint is violated by the optimal solution $\mathbf{x}^*$ to the original problem. The dual of the new problem is

$$\begin{array}{ll} \text{maximize} & \mathbf{p}'\mathbf{b} + p_{m+1}b_{m+1} \\ \text{subject to} & [\mathbf{p}' \ \ p_{m+1}] \begin{bmatrix} \mathbf{A} \\ \mathbf{a}'_{m+1} \end{bmatrix} \leq \mathbf{c}', \end{array}$$

where p_{m+1} is a dual variable associated with the new constraint. Let $\mathbf{p}^*$ be an optimal basic feasible solution to the original dual problem. Then, $(\mathbf{p}^*, 0)$ is a feasible solution to the new dual problem.

Let m be the dimension of $\mathbf{p}$, which is the same as the original number of constraints. Since $\mathbf{p}^*$ is a basic feasible solution to the original dual problem, m of the constraints in $(\mathbf{p}^*)'\mathbf{A} \leq \mathbf{c}'$ are linearly independent and active. However, there is no guarantee that at $(\mathbf{p}^*, 0)$ we will have $m+1$ linearly independent active constraints of the new dual problem. In particular, $(\mathbf{p}^*, 0)$ need not be a basic feasible solution to the new dual problem and may not provide a convenient starting point for the dual simplex method on the new problem. While it may be possible to obtain a dual basic feasible solution by setting p_{m+1} to a suitably chosen nonzero value, we present here an alternative approach.

Let us assume, without loss of generality, that $\mathbf{a}'_{m+1}\mathbf{x}^* > b_{m+1}$. We introduce the auxiliary primal problem

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}'\mathbf{x} + Mx_{n+1} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{a}'_{m+1}\mathbf{x} - x_{n+1} = b_{m+1} \\ & \mathbf{x} \geq \mathbf{0},\ x_{n+1} \geq 0, \end{aligned}$$

where M is a large positive constant. A primal feasible basis for the auxiliary problem is obtained by picking the basic variables of the optimal solution to the original problem, together with the variable x_{n+1}. The resulting basis matrix is the same as the matrix $\overline{\mathbf{B}}$ of the preceding subsection. There is a difference, however. In the preceding subsection, $\overline{\mathbf{B}}$ was a dual feasible basis, whereas here $\overline{\mathbf{B}}$ is a primal feasible basis. For this reason, the primal simplex method can now be used to solve the auxiliary problem to optimality.

Suppose that an optimal solution to the auxiliary problem satisfies $x_{n+1} = 0$; this will be the case if the new problem is feasible and the coefficient M is large enough. Then, the additional constraint $\mathbf{a}'_{m+1}\mathbf{x} = b_{m+1}$ has been satisfied and we have an optimal solution to the new problem.

Changes in the requirement vector b

Suppose that some component b_i of the requirement vector $\mathbf{b}$ is changed to $b_i + \delta$. Equivalently, the vector $\mathbf{b}$ is changed to $\mathbf{b} + \delta\mathbf{e}_i$, where $\mathbf{e}_i$ is the ith unit vector. We wish to determine the range of values of δ under which the current basis remains optimal. Note that the optimality conditions are not affected by the change in $\mathbf{b}$. We therefore need to examine only the feasibility condition

$$\mathbf{B}^{-1}\big(\mathbf{b} + \delta\mathbf{e}_i\big) \geq \mathbf{0}. \tag{5.1}$$

Let $\mathbf{g} = (\beta_{1i}, \beta_{2i}, \ldots, \beta_{mi})$ be the ith column of $\mathbf{B}^{-1}$. Equation (5.1) becomes

$$\mathbf{x}_B + \delta\mathbf{g} \geq \mathbf{0},$$

or,

$$x_{B(j)} + \delta\beta_{ji} \geq 0, \qquad j = 1, \ldots, m.$$

Equivalently,

$$\max_{\{j|\beta_{ji}>0\}} \left(-\frac{x_{B(j)}}{\beta_{ji}}\right) \leq \delta \leq \min_{\{j|\beta_{ji}<0\}} \left(-\frac{x_{B(j)}}{\beta_{ji}}\right).$$

For δ in this range, the optimal cost, as a function of δ, is given by $\mathbf{c}_B'\mathbf{B}^{-1}(\mathbf{b}+\delta\mathbf{e}_i) = \mathbf{p}'\mathbf{b} + \delta p_i$, where $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$ is the (optimal) dual solution associated with the current basis $\mathbf{B}$.

If δ is outside the allowed range, the current solution satisfies the optimality (or dual feasibility) conditions, but is primal infeasible. In that case, we can apply the dual simplex algorithm starting from the current basis.

Example 5.3 Consider the optimal tableau

		x_1	x_2	x_3	x_4
	12	0	0	2	7
$x_1 =$	2	1	0	−3	2
$x_2 =$	2	0	1	5	−3

from Example 5.1.

Let us contemplate adding δ to b_1. We look at the first column of $\mathbf{B}^{-1}$ which is $(-3,5)$. The basic variables under the same basis are $x_1 = 2-3\delta$ and $2+5\delta$. This basis will remain feasible as long as $2-3\delta \geq 0$ and $2+5\delta \geq 0$, that is, if $-2/5 \leq \delta \leq 2/3$. The rate of change of the optimal cost per unit change of δ is given by $\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{e}_1 = (-5,-1)'(-3,5) = 10$.

If δ is increased beyond $2/3$, then x_1 becomes negative. At this point, we can perform an iteration of the dual simplex method to remove x_1 from the basis, and x_3 enters the basis.

Changes in the cost vector c

Suppose now that some cost coefficient c_j becomes $c_j + \delta$. The primal feasibility condition is not affected. We therefore need to focus on the optimality condition

$$\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A} \leq \mathbf{c}'.$$

If c_j is the cost coefficient of a nonbasic variable x_j, then $\mathbf{c}_B$ does not change, and the only inequality that is affected is the one for the reduced cost of x_j; we need

$$\mathbf{c}_B'\mathbf{B}^{-1}\mathbf{A}_j \leq c_j + \delta,$$

or

$$\delta \geq -\bar{c}_j.$$

If this condition holds, the current basis remains optimal; otherwise, we can apply the primal simplex method starting from the current basic feasible solution.

If c_j is the cost coefficient of the ℓth basic variable, that is, if $j = B(\ell)$, then $\mathbf{c}_B$ becomes $\mathbf{c}_B + \delta\mathbf{e}_\ell$ and all of the optimality conditions will be affected. The optimality conditions for the new problem are

$$\big(\mathbf{c}_B + \delta\mathbf{e}_\ell\big)'\mathbf{B}^{-1}\mathbf{A}_i \leq c_i, \qquad \forall\, i \neq j.$$

(Since x_j is a basic variable, its reduced cost stays at zero and need not be examined.) Equivalently,

$$\delta q_{\ell i} \leq \bar{c}_i, \qquad \forall\, i \neq j,$$

where $q_{\ell i}$ is the ℓth entry of $\mathbf{B}^{-1}\mathbf{A}_i$, which can be obtained from the simplex tableau. These inequalities determine the range of δ for which the same basis remains optimal.

Example 5.4 We consider once more the problem in Example 5.1 and determine the range of changes δ_i of c_i, under which the same basis remains optimal. Since x_3 and x_4 are nonbasic variables, we obtain the conditions

$$\begin{aligned} \delta_3 &\geq -\bar{c}_3 = -2, \\ \delta_4 &\geq -\bar{c}_4 = -7. \end{aligned}$$

Consider now adding δ_1 to c_1. From the simplex tableau, we obtain $q_{12} = 0$, $q_{13} = -3$, $q_{14} = 2$, and we are led to the conditions

$$\begin{aligned} \delta_1 &\geq -2/3, \\ \delta_1 &\leq 7/2. \end{aligned}$$

Changes in a nonbasic column of A

Suppose that some entry a_{ij} in the jth column $\mathbf{A}_j$ of the matrix $\mathbf{A}$ is changed to $a_{ij} + \delta$. We wish to determine the range of values of δ for which the old optimal basis remains optimal.

If the column $\mathbf{A}_j$ is nonbasic, the basis matrix $\mathbf{B}$ does not change, and the primal feasibility condition is unaffected. Furthermore, only the reduced cost of the jth column is affected, leading to the condition

$$c_j - \mathbf{p}'\big(\mathbf{A}_j + \delta\mathbf{e}_i\big) \geq 0,$$

or,

$$\bar{c}_j - \delta p_i \geq 0,$$

where $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$. If this condition is violated, the nonbasic column $\mathbf{A}_j$ can be brought into the basis, and we can continue with the primal simplex method.

Changes in a basic column of A

If one of the entries of a basic column $\mathbf{A}_j$ changes, then both the feasibility and optimality conditions are affected. This case is more complicated and we leave the full development for the exercises. As it turns out, the range of values of δ for which the same basis is optimal is again an interval (Exercise 5.3).

Suppose that the basic column $\mathbf{A}_j$ is changed to $\mathbf{A}_j + \delta\mathbf{e}_i$, where $\mathbf{e}_i$ is the ith unit vector. Assume that both the original problem and its dual have unique and nondegenerate optimal solutions $\mathbf{x}^*$ and $\mathbf{p}$, respectively. Let $\mathbf{x}^*(\delta)$ be an optimal solution to the modified problem, as a function of δ. It can be shown (Exercise 5.2) that for small δ we have

$$\mathbf{c}'\mathbf{x}^*(\delta) = \mathbf{c}'\mathbf{x}^* - \delta x_j^* p_i + O(\delta^2).$$

For an intuitive interpretation of this equation, let us consider the diet problem and recall that a_{ij} corresponds to the amount of the ith nutrient in the jth food. Given an optimal solution $\mathbf{x}^*$ to the original problem, an increase of a_{ij} by δ means that we are getting "for free" an additional amount δx_j^* of the ith nutrient. Since the dual variable p_i is the marginal cost per unit of the ith nutrient, we are getting for free something that is normally worth $\delta p_i x_j^*$, and this allows us to reduce our costs by that same amount.

Production planning revisited

In Section 1.2, we introduced a production planning problem that DEC had faced in the end of 1988. In this section, we answer some of the questions that we posed. Recall that there were two important choices, whether to use the constrained or the unconstrained mode of production for disk drives, and whether to use alternative memory boards. As discussed in Section 1.2, these four combinations of choices led to four different linear programming problems. We report the solution to these problems, as obtained from a linear programming package, in Table 5.1.

Table 5.1 indicates that revenues can substantially increase by using alternative memory boards, and the company should definitely do so. The decision of whether to use the constrained or the unconstrained mode of production for disk drives is less clear. In the constrained mode, the revenue is 248 million versus 213 million in the unconstrained mode. However, customer satisfaction and, therefore, future revenues might be affected, since in the constrained mode some customers will get a product different than the desired one. Moreover, these results are obtained assuming that the number of available 256K memory boards and disk drives were 8,000 and 3,000, respectively, which is the lowest value in the range that was estimated. We should therefore examine the sensitivity of the solution as the number of available 256K memory boards and disk drives increases.

Alt. boards	**Mode**	**Revenue**	x_1	x_2	x_3	x_4	x_5
no	constr.	145	0	2.5	0	0.5	2
yes	constr.	248	1.8	2	0	1	2
no	unconstr.	133	0.272	1.304	0.3	0.5	2.7
yes	unconstr.	213	1.8	1.035	0.3	0.5	2.7

Table 5.1: Optimal solutions to the four variants of the production planning problem. Revenue is in millions of dollars and the quantities x_i are in thousands.

With most linear programming packages, the output includes the values of the dual variables, as well as the range of parameter variations under which local sensitivity analysis is valid. Table 5.2 presents the values of the dual variables associated with the constraints on available disk drives and 256K memory boards. In addition, it provides the range of allowed changes on the number of disk drives and memory boards that would leave the dual variables unchanged. This information is provided for the two linear programming problems corresponding to constrained and unconstrained mode of production for disk drives, respectively, under the assumption that alternative memory boards will be used.

Mode	**Constrained**	**Unconstrained**
Revenue	248	213
Dual variable for 256K boards	15	0
Range for 256K boards	$[-1.5, 0.2]$	$[-1.62, \infty]$
Dual variable for disk drives	0	23.52
Range for disk drives	$[-0.2, 0.75]$	$[-0.91, 1.13]$

Table 5.2: Dual prices and ranges for the constraints corresponding to the availability of the number of 256K memory boards and disk drives.

In the constrained mode, increasing the number of available 256K boards by 0.2 thousand (the largest number in the allowed range) results in a revenue increase of $15 \times 0.2 = 3$ million. In the unconstrained mode, increasing the number of available 256K boards has no effect on revenues, because the dual variable is zero and the range extends upwards to infinity. In the constrained mode, increasing the number of available disk drives by up to 0.75 thousand (the largest number in the allowed range) has no effect on revenue. Finally, in the unconstrained mode, increasing the number of available disk drives by 1.13 thousand results in a revenue increase of $23.52 \times 1.13 = 26.57$ million.

In conclusion, in the constrained mode of production, it is important to aim at an increase of the number of available 256K memory boards, while in the unconstrained mode, increasing the number of disk drives is more important.

This example demonstrates that even a small linear programming problem (with five variables, in this case) can have an impact on a company's planning process. Moreover, the information provided by linear programming solvers (dual variables, ranges, etc.) can offer significant insights and can be a very useful aid to decision makers.

5.2 Global dependence on the right-hand side vector

In this section, we take a global view of the dependence of the optimal cost on the requirement vector $\mathbf{b}$.

Let

$$P(\mathbf{b}) = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}$$

be the feasible set, and note that our notation makes the dependence on $\mathbf{b}$ explicit. Let

$$S = \{\mathbf{b} \mid P(\mathbf{b}) \text{ is nonempty}\},$$

and observe that

$$S = \{\mathbf{A}\mathbf{x} \mid \mathbf{x} \geq \mathbf{0}\};$$

in particular, S is a convex set. For any $\mathbf{b} \in S$, we define

$$F(\mathbf{b}) = \min_{\mathbf{x} \in P(\mathbf{b})} \mathbf{c}'\mathbf{x},$$

which is the optimal cost as a function of $\mathbf{b}$.

Throughout this section, we assume that the dual feasible set $\{\mathbf{p} \mid \mathbf{p}'\mathbf{A} \leq \mathbf{c}'\}$ is nonempty. Then, duality theory implies that the optimal primal cost $F(\mathbf{b})$ is finite for every $\mathbf{b} \in S$. Our goal is to understand the structure of the function $F(\mathbf{b})$, for $\mathbf{b} \in S$.

Let us fix a particular element $\mathbf{b}^*$ of S. Suppose that there exists a nondegenerate primal optimal basic feasible solution, and let $\mathbf{B}$ be the corresponding optimal basis matrix. The vector $\mathbf{x}_B$ of basic variables at that optimal solution is given by $\mathbf{x}_B = \mathbf{B}^{-1}\mathbf{b}^*$, and is positive by nondegeneracy. In addition, the vector of reduced costs is nonnegative. If we change $\mathbf{b}^*$ to $\mathbf{b}$ and if the difference $\mathbf{b} - \mathbf{b}^*$ is sufficiently small, $\mathbf{B}^{-1}\mathbf{b}$ remains positive and we still have a basic feasible solution. The reduced costs are not affected by the change from $\mathbf{b}^*$ to $\mathbf{b}$ and remain nonnegative. Therefore, $\mathbf{B}$ is an optimal basis for the new problem as well. The optimal cost $F(\mathbf{b})$ for the new problem is given by

$$F(\mathbf{b}) = \mathbf{c}_B'\mathbf{B}^{-1}\mathbf{b} = \mathbf{p}'\mathbf{b}, \qquad \text{for } \mathbf{b} \text{ close to } \mathbf{b}^*,$$

where $\mathbf{p}' = \mathbf{c}_B'\mathbf{B}^{-1}$ is the optimal solution to the dual problem. This establishes that in the vicinity of $\mathbf{b}^*$, $F(\mathbf{b})$ is a linear function of $\mathbf{b}$ and its gradient is given by $\mathbf{p}$.

We now turn to the global properties of $F(\mathbf{b})$.

Theorem 5.1 *The optimal cost $F(\mathbf{b})$ is a convex function of $\mathbf{b}$ on the set S.*

Proof. Let $\mathbf{b}^1$ and $\mathbf{b}^2$ be two elements of S. For $i = 1, 2$, let $\mathbf{x}^i$ be an optimal solution to the problem of minimizing $\mathbf{c}'\mathbf{x}$ subject to $\mathbf{x} \geq \mathbf{0}$ and $\mathbf{A}\mathbf{x} = \mathbf{b}^i$. Thus, $F(\mathbf{b}^1) = \mathbf{c}'\mathbf{x}^1$ and $F(\mathbf{b}^2) = \mathbf{c}'\mathbf{x}^2$. Fix a scalar $\lambda \in [0, 1]$, and note that the vector $\mathbf{y} = \lambda\mathbf{x}^1 + (1-\lambda)\mathbf{x}^2$ is nonnegative and satisfies $\mathbf{A}\mathbf{y} = \lambda\mathbf{b}^1 + (1-\lambda)\mathbf{b}^2$. In particular, $\mathbf{y}$ is a feasible solution to the linear programming problem obtained when the requirement vector $\mathbf{b}$ is set to $\lambda\mathbf{b}^1 + (1-\lambda)\mathbf{b}^2$. Therefore,

$$F\big(\lambda\mathbf{b}^1 + (1-\lambda)\mathbf{b}^2\big) \leq \mathbf{c}'\mathbf{y} = \lambda\mathbf{c}'\mathbf{x}^1 + (1-\lambda)\mathbf{c}'\mathbf{x}^2 = \lambda F(\mathbf{b}^1) + (1-\lambda)F(\mathbf{b}^2),$$

establishing the convexity of F. □

We now corroborate Theorem 5.1 by taking a different approach, involving the dual problem

$$\begin{aligned} \text{maximize} \quad & \mathbf{p}'\mathbf{b} \\ \text{subject to} \quad & \mathbf{p}'\mathbf{A} \leq \mathbf{c}', \end{aligned}$$

which has been assumed feasible. For any $\mathbf{b} \in S$, $F(\mathbf{b})$ is finite and, by strong duality, is equal to the optimal value of the dual objective. Let $\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^N$ be the extreme points of the dual feasible set. (Our standing assumption is that the matrix $\mathbf{A}$ has linearly independent rows; hence its columns span $\Re^m$. Equivalently, the rows of $\mathbf{A}'$ span $\Re^m$ and Theorem 2.6 in Section 2.5 implies that the dual feasible set must have at least one

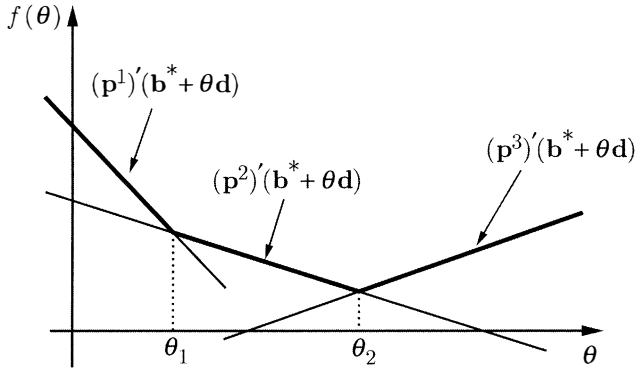


Figure 5.1: The optimal cost when the vector $\mathbf{b}$ is a function of a scalar parameter. Each linear piece is of the form $(\mathbf{p}^i)'(\mathbf{b}^* + \theta\mathbf{d})$, where $\mathbf{p}^i$ is the ith extreme point of the dual feasible set. In each one of the intervals $\theta < \theta_1$, $\theta_1 < \theta < \theta_2$, and $\theta > \theta_2$, we have different dual optimal solutions, namely, $\mathbf{p}^1$, $\mathbf{p}^2$, and $\mathbf{p}^3$, respectively. For $\theta = \theta_1$ or $\theta = \theta_2$, the dual problem has multiple optimal solutions.

extreme point.) Since the optimum of the dual must be attained at an extreme point, we obtain

$$F(\mathbf{b}) = \max_{i=1,\ldots,N} (\mathbf{p}^i)'\mathbf{b}, \qquad \mathbf{b} \in S. \tag{5.2}$$

In particular, F is equal to the maximum of a finite collection of linear functions. It is therefore a piecewise linear convex function, and we have a new proof of Theorem 5.1. In addition, within a region where F is linear, we have $F(\mathbf{b}) = (\mathbf{p}^i)'\mathbf{b}$, where $\mathbf{p}^i$ is a corresponding dual optimal solution, in agreement with our earlier discussion.

For those values of $\mathbf{b}$ for which F is not differentiable, that is, at the junction of two or more linear pieces, the dual problem does not have a unique optimal solution and this implies that every optimal basic feasible solution to the primal is degenerate. (This is because, as shown earlier in this section, the existence of a nondegenerate optimal basic feasible solution to the primal implies that F is locally linear.)

We now restrict attention to changes in $\mathbf{b}$ of a particular type, namely, $\mathbf{b} = \mathbf{b}^* + \theta\mathbf{d}$, where $\mathbf{b}^*$ and $\mathbf{d}$ are fixed vectors and θ is a scalar. Let $f(\theta) = F(\mathbf{b}^* + \theta\mathbf{d})$ be the optimal cost as a function of the scalar parameter θ. Using Eq. (5.2), we obtain

$$f(\theta) = \max_{i=1,\ldots,N} (\mathbf{p}^i)'(\mathbf{b}^* + \theta\mathbf{d}), \qquad \mathbf{b}^* + \theta\mathbf{d} \in S.$$

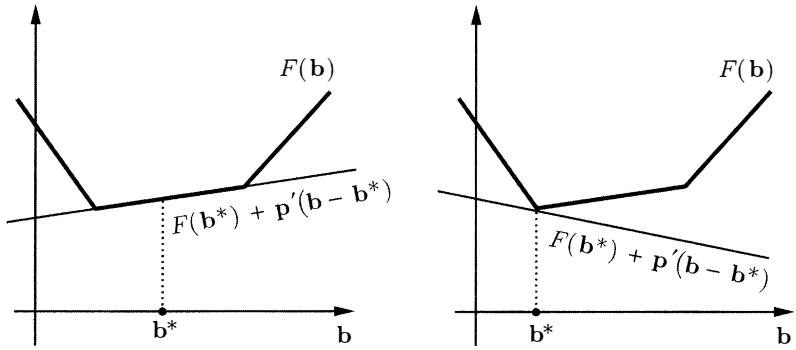


Figure 5.2: Illustration of subgradients of a function F at a point $\mathbf{b}^*$. A subgradient $\mathbf{p}$ is the gradient of a linear function $F(\mathbf{b}^*) + \mathbf{p}'(\mathbf{b} - \mathbf{b}^*)$ that lies below the function $F(\mathbf{b})$ and agrees with it for $\mathbf{b} = \mathbf{b}^*$.

This is essentially a "section" of the function F; it is again a piecewise linear convex function; see Figure 5.1. Once more, at breakpoints of this function, every optimal basic feasible solution to the primal must be degenerate.

5.3 The set of all dual optimal solutions*

We have seen that if the function F is defined, finite, and linear in the vicinity of a certain vector $\mathbf{b}^*$, then there is a unique optimal dual solution, equal to the gradient of F at that point, which leads to the interpretation of dual optimal solutions as marginal costs. We would like to extend this interpretation so that it remains valid at the breakpoints of F. This is indeed possible: we will show shortly that any dual optimal solution can be viewed as a "generalized gradient" of F. We first need the following definition, which is illustrated in Figure 5.2.

Definition 5.1 *Let F be a convex function defined on a convex set S. Let $\mathbf{b}^*$ be an element of S. We say that a vector $\mathbf{p}$ is a* **subgradient** *of F at $\mathbf{b}^*$ if*

$$F(\mathbf{b}^*) + \mathbf{p}'(\mathbf{b} - \mathbf{b}^*) \leq F(\mathbf{b}), \qquad \forall\, \mathbf{b} \in S.$$

Note that if $\mathbf{b}^*$ is a breakpoint of the function F, then there are several subgradients. On the other hand, if F is linear near $\mathbf{b}^*$, there is a unique subgradient, equal to the gradient of F.

Theorem 5.2 *Suppose that the linear programming problem of minimizing* $\mathbf{c}'\mathbf{x}$ *subject to* $\mathbf{A}\mathbf{x} = \mathbf{b}^*$ *and* $\mathbf{x} \geq \mathbf{0}$ *is feasible and that the optimal cost is finite. Then, a vector* $\mathbf{p}$ *is an optimal solution to the dual problem if and only if it is a subgradient of the optimal cost function* F *at the point* $\mathbf{b}^*$.

Proof. Recall that the function F is defined on the set S, which is the set of vectors $\mathbf{b}$ for which the set $P(\mathbf{b})$ of feasible solutions to the primal problem is nonempty. Suppose that $\mathbf{p}$ is an optimal solution to the dual problem. Then, strong duality implies that $\mathbf{p}'\mathbf{b}^* = F(\mathbf{b}^*)$. Consider now some arbitrary $\mathbf{b} \in S$. For any feasible solution $\mathbf{x} \in P(\mathbf{b})$, weak duality yields $\mathbf{p}'\mathbf{b} \leq \mathbf{c}'\mathbf{x}$. Taking the minimum over all $\mathbf{x} \in P(\mathbf{b})$, we obtain $\mathbf{p}'\mathbf{b} \leq F(\mathbf{b})$. Hence, $\mathbf{p}'\mathbf{b} - \mathbf{p}'\mathbf{b}^* \leq F(\mathbf{b}) - F(\mathbf{b}^*)$, and we conclude that $\mathbf{p}$ is a subgradient of F at $\mathbf{b}^*$.

We now prove the converse. Let $\mathbf{p}$ be a subgradient of F at $\mathbf{b}^*$; that is,

$$F(\mathbf{b}^*) + \mathbf{p}'(\mathbf{b} - \mathbf{b}^*) \leq F(\mathbf{b}), \qquad \forall\, \mathbf{b} \in S. \tag{5.3}$$

Pick some $\mathbf{x} \geq \mathbf{0}$, let $\mathbf{b} = \mathbf{A}\mathbf{x}$, and note that $\mathbf{x} \in P(\mathbf{b})$. In particular, $F(\mathbf{b}) \leq \mathbf{c}'\mathbf{x}$. Using Eq. (5.3), we obtain

$$\mathbf{p}'\mathbf{A}\mathbf{x} = \mathbf{p}'\mathbf{b} \leq F(\mathbf{b}) - F(\mathbf{b}^*) + \mathbf{p}'\mathbf{b}^* \leq \mathbf{c}'\mathbf{x} - F(\mathbf{b}^*) + \mathbf{p}'\mathbf{b}^*.$$

Since this is true for all $\mathbf{x} \geq \mathbf{0}$, we must have $\mathbf{p}'\mathbf{A} \leq \mathbf{c}'$, which shows that $\mathbf{p}$ is a dual feasible solution. Also, by letting $\mathbf{x} = \mathbf{0}$, we obtain $F(\mathbf{b}^*) \leq \mathbf{p}'\mathbf{b}^*$. Using weak duality, every dual feasible solution $\mathbf{q}$ must satisfy $\mathbf{q}'\mathbf{b}^* \leq F(\mathbf{b}^*) \leq \mathbf{p}'\mathbf{b}^*$, which shows that $\mathbf{p}$ is a dual optimal solution. □

5.4 Global dependence on the cost vector

In the last two sections, we fixed the matrix $\mathbf{A}$ and the vector $\mathbf{c}$, and we considered the effect of changing the vector $\mathbf{b}$. The key to our development was the fact that the set of dual feasible solutions remains the same as $\mathbf{b}$ varies. In this section, we study the case where $\mathbf{A}$ and $\mathbf{b}$ are fixed, but the vector $\mathbf{c}$ varies. In this case, the primal feasible set remains unaffected; our standing assumption will be that it is nonempty.

We define the dual feasible set

$$Q(\mathbf{c}) = \{\mathbf{p} \mid \mathbf{p}'\mathbf{A} \leq \mathbf{c}\},$$

and let

$$T = \{\mathbf{c} \mid Q(\mathbf{c}) \text{ is nonempty}\}.$$

If $\mathbf{c}^1 \in T$ and $\mathbf{c}^2 \in T$, then there exist $\mathbf{p}^1$ and $\mathbf{p}^2$ such that $(\mathbf{p}^1)'\mathbf{A} \leq \mathbf{c}'$ and $(\mathbf{p}^2)'\mathbf{A} \leq \mathbf{c}'$. For any scalar $\lambda \in [0, 1]$, we have

$$\big(\lambda(\mathbf{p}^1)' + (1 - \lambda)(\mathbf{p}^2)'\big)\mathbf{A} \leq \lambda\mathbf{c}^1 + (1 - \lambda)\mathbf{c}^2,$$

and this establishes that $\lambda \mathbf{c}^1 + (1-\lambda)\mathbf{c}^2 \in T$. We have therefore shown that T is a convex set.

If $\mathbf{c} \notin T$, the infeasibility of the dual problem implies that the optimal primal cost is $-\infty$. On the other hand, if $\mathbf{c} \in T$, the optimal primal cost must be finite. Thus, the optimal primal cost, which we will denote by $G(\mathbf{c})$, is finite if and only if $\mathbf{c} \in T$.

Let $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N$ be the basic feasible solutions in the primal feasible set; clearly, these do not depend on $\mathbf{c}$. Since an optimal solution to a standard form problem can always be found at an extreme point, we have

$$G(\mathbf{c}) = \min_{i=1,\ldots,N} \mathbf{c}'\mathbf{x}^i.$$

Thus, $G(\mathbf{c})$ is the minimum of a finite collection of linear functions and is a piecewise linear concave function. If for some value $\mathbf{c}^*$ of $\mathbf{c}$, the primal has a unique optimal solution $\mathbf{x}^i$, we have $(\mathbf{c}^*)'\mathbf{x}^i < (\mathbf{c}^*)'\mathbf{x}^j$, for all $j \neq i$. For $\mathbf{c}$ very close to $\mathbf{c}^*$, the inequalities $\mathbf{c}'\mathbf{x}^i < \mathbf{c}'\mathbf{x}^j$, $j \neq i$, continue to hold, implying that $\mathbf{x}^i$ is still a unique primal optimal solution with cost $\mathbf{c}'\mathbf{x}^i$. We conclude that, locally, $G(\mathbf{c}) = \mathbf{c}'\mathbf{x}^i$. On the other hand, at those values of $\mathbf{c}$ that lead to multiple primal optimal solutions, the function G has a breakpoint.

We summarize the main points of the preceding discussion.

Theorem 5.3 *Consider a feasible linear programming problem in standard form.*

(a) *The set T of all $\mathbf{c}$ for which the optimal cost is finite, is convex.*

(b) *The optimal cost $G(\mathbf{c})$ is a concave function of $\mathbf{c}$ on the set T.*

(c) *If for some value of $\mathbf{c}$ the primal problem has a unique optimal solution $\mathbf{x}^*$, then G is linear in the vicinity of $\mathbf{c}$ and its gradient is equal to $\mathbf{x}^*$.*

5.5 Parametric programming

Let us fix $\mathbf{A}$, $\mathbf{b}$, $\mathbf{c}$, and a vector $\mathbf{d}$ of the same dimension as $\mathbf{c}$. For any scalar θ, we consider the problem

$$\begin{aligned} \text{minimize} \quad & (\mathbf{c} + \theta\mathbf{d})'\mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

and let $g(\theta)$ be the optimal cost as a function of θ. Naturally, we assume that the feasible set is nonempty. For those values of θ for which the optimal cost is finite, we have

$$g(\theta) = \min_{i=1,\ldots,N} (\mathbf{c} + \theta\mathbf{d})'\mathbf{x}^i,$$

where $\mathbf{x}^1, \ldots, \mathbf{x}^N$ are the extreme points of the feasible set; see Figure 5.3. In particular, $g(\theta)$ is a piecewise linear and concave function of the parameter θ. In this section, we discuss a systematic procedure, based on the simplex method, for obtaining $g(\theta)$ for all values of θ. We start with an example.

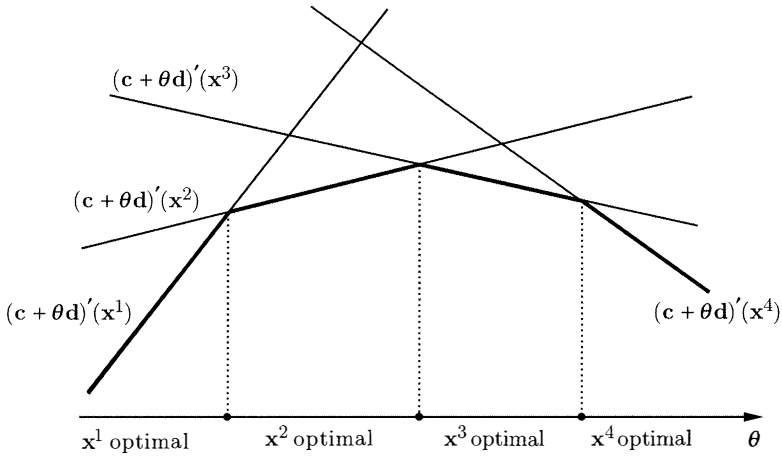


Figure 5.3: The optimal cost $g(\theta)$ as a function of θ.

Example 5.5 Consider the problem

$$\begin{array}{lrcrcrcl} \text{minimize} & (-3+2\theta)x_1 & + & (3-\theta)x_2 & + & x_3 & & \\ \text{subject to} & x_1 & + & 2x_2 & - & 3x_3 & \leq & 5 \\ & 2x_1 & + & x_2 & - & 4x_3 & \leq & 7 \\ & x_1, x_2, x_3 & \geq & 0. & & & & \end{array}$$

We introduce slack variables in order to bring the problem into standard form, and then let the slack variables be the basic variables. This determines a basic feasible solution and leads to the following tableau.

		x_1	x_2	x_3	x_4	x_5
	0	$-3+2\theta$	$3-\theta$	1	0	0
$x_4 =$	5	1	2	-3	1	0
$x_5 =$	7	2	1	-4	0	1

If $-3+2\theta \geq 0$ and $3-\theta \geq 0$, all reduced costs are nonnegative and we have an optimal basic feasible solution. In particular,

$$g(\theta) = 0, \qquad \text{if } \frac{3}{2} \leq \theta \leq 3.$$

If θ is increased slightly above 3, the reduced cost of x_2 becomes negative and we no longer have an optimal basic feasible solution. We let x_2 enter the basis, x_4 exits, and we obtain the new tableau:

		x_1	x_2	x_3	x_4	x_5
	$-7.5+2.5\theta$	$-4.5+2.5\theta$	0	$5.5-1.5\theta$	$-1.5+0.5\theta$	0
$x_2 =$	2.5	0.5	1	-1.5	0.5	0
$x_5 =$	4.5	1.5	0	-2.5	-0.5	1

We note that all reduced costs are nonnegative if and only if $3 \leq \theta \leq 5.5/1.5$. The optimal cost for that range of values of θ is

$$g(\theta) = 7.5 - 2.5\theta, \qquad \text{if } 3 \leq \theta \leq \frac{5.5}{1.5}.$$

If θ is increased beyond $5.5/1.5$, the reduced cost of x_3 becomes negative. If we attempt to bring x_3 into the basis, we cannot find a positive pivot element in the third column of the tableau, and the problem is unbounded, with $g(\theta) = -\infty$.

Let us now go back to the original tableau and suppose that θ is decreased to a value slightly below $3/2$. Then, the reduced cost of x_1 becomes negative, we let x_1 enter the basis, and x_5 exits. The new tableau is:

		x_1	x_2	x_3	x_4	x_5
	$10.5-7\theta$	0	$4.5-2\theta$	$-5+4\theta$	0	$1.5-\theta$
$x_4 =$	1.5	0	1.5	-1	1	-0.5
$x_1 =$	3.5	1	0.5	-2	0	0.5

We note that all of the reduced costs are nonnegative if and only if $5/4 \leq \theta \leq 3/2$. For these values of θ, we have an optimal solution, with an optimal cost of

$$g(\theta) = -10.5 + 7\theta, \qquad \text{if } \frac{5}{4} \leq \theta \leq \frac{3}{2}.$$

Finally, for $\theta < 5/4$, the reduced cost of x_3 is negative, but the optimal cost is equal to $-\infty$, because all entries in the third column of the tableau are negative. We plot the optimal cost in Figure 5.4.

We now generalize the steps in the preceding example, in order to obtain a broader methodology. The key observation is that once a basis is fixed, the reduced costs are affine (linear plus a constant) functions of θ. Then, if we require that all reduced costs be nonnegative, we force θ to belong to some interval. (The interval could be empty but if it is nonempty, its endpoints are also included.) We conclude that for any given basis, the set of θ for which this basis is optimal is a closed interval.

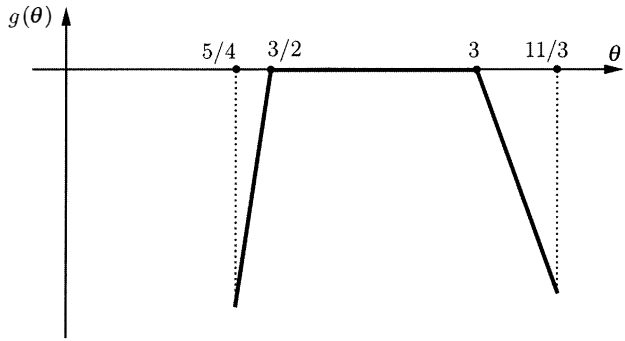


Figure 5.4: The optimal cost $g(\theta)$ as a function of θ, in Example 5.5. For θ outside the interval $[5/4, 11/3]$, $g(\theta)$ is equal to $-\infty$.

Let us now assume that we have chosen a basic feasible solution and an associated basis matrix $\mathbf{B}$, and suppose that this basis is optimal for θ satisfying $\theta_1 \leq \theta \leq \theta_2$. Let x_j be a variable whose reduced cost becomes negative for $\theta > \theta_2$. Since this reduced cost is nonnegative for $\theta_1 \leq \theta \leq \theta_2$, it must be equal to zero when $\theta = \theta_2$. We now attempt to bring x_j into the basis and consider separately the different cases that may arise.

Suppose that no entry of the jth column $\mathbf{B}^{-1}\mathbf{A}_j$ of the simplex tableau is positive. For $\theta > \theta_2$, the reduced cost of x_j is negative, and this implies that the optimal cost is $-\infty$ in that range.

If the jth column of the tableau has at least one positive element, we carry out a change of basis and obtain a new basis matrix $\overline{\mathbf{B}}$. For $\theta = \theta_2$, the reduced cost of the entering variable is zero and, therefore, the cost associated with the new basis is the same as the cost associated with the old basis. Since the old basis was optimal for $\theta = \theta_2$, the same must be true for the new basis. On the other hand, for $\theta < \theta_2$, the entering variable x_j had a positive reduced cost. According to the pivoting mechanics, and for $\theta < \theta_2$, a negative multiple of the pivot row is added to the pivot row, and this makes the reduced cost of the exiting variable negative. This implies that the new basis cannot be optimal for $\theta < \theta_2$. We conclude that the range of values of θ for which the new basis is optimal is of the form $\theta_2 \leq \theta \leq \theta_3$, for some θ_3. By continuing similarly, we obtain a sequence of bases, with the ith basis being optimal for $\theta_i \leq \theta \leq \theta_{i+1}$.

Note that a basis which is optimal for $\theta \in [\theta_i, \theta_{i+1}]$ cannot be optimal for values of θ greater than θ_{i+1}. Thus, if $\theta_{i+1} > \theta_i$ for all i, the same basis cannot be encountered more than once and the entire range of values of θ will be traced in a finite number of iterations, with each iteration leading to a new breakpoint of the optimal cost function $g(\theta)$. (The number of breakpoints may increase exponentially with the dimension of the problem.)

The situation is more complicated if for some basis we have $\theta_i = \theta_{i+1}$. In this case, it is possible that the algorithm keeps cycling between a finite number of different bases, all of which are optimal only for $\theta = \theta_i = \theta_{i+1}$. Such cycling can only happen in the presence of degeneracy in the primal problem (Exercise 5.17), but can be avoided if an appropriate anticycling rule is followed. In conclusion, the procedure we have outlined, together with an anticycling rule, partitions the range of possible values of θ into consecutive intervals and, for each interval, provides us with an optimal basis and the optimal cost function as a function of θ.

There is another variant of parametric programming that can be used when $\mathbf{c}$ is kept fixed but $\mathbf{b}$ is replaced by $\mathbf{b} + \theta\mathbf{d}$, where $\mathbf{d}$ is a given vector and θ is a scalar. In this case, the zeroth column of the tableau depends on θ. Whenever θ reaches a value at which some basic variable becomes negative, we apply the dual simplex method in order to recover primal feasibility.

5.6 Summary

In this chapter, we have studied the dependence of optimal solutions and of the optimal cost on the problem data, that is, on the entries of $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{c}$. For many of the cases that we have examined, a common methodology was used. Subsequent to a change in the problem data, we first examine its effects on the feasibility and optimality conditions. If we wish the same basis to remain optimal, this leads us to certain limitations on the magnitude of the changes in the problem data. For larger changes, we no longer have an optimal basis and some remedial action (involving the primal or dual simplex method) is typically needed.

We close with a summary of our main results.

(a) If a new variable is added, we check its reduced cost and if it is negative, we add a new column to the tableau and proceed from there.

(b) If a new constraint is added, we check whether it is violated and if so, we form an auxiliary problem and its tableau, and proceed from there.

(c) If an entry of $\mathbf{b}$ or $\mathbf{c}$ is changed by δ, we obtain an interval of values of δ for which the same basis remains optimal.

(d) If an entry of $\mathbf{A}$ is changed by δ, a similar analysis is possible. However, this case is somewhat complicated if the change affects an entry of a basic column.

(e) Assuming that the dual problem is feasible, the optimal cost is a piecewise linear convex function of the vector $\mathbf{b}$ (for those $\mathbf{b}$ for which the primal is feasible). Furthermore, subgradients of the optimal cost function correspond to optimal solutions to the dual problem.

(f) Assuming that the primal problem is feasible, the optimal cost is a piecewise linear concave function of the vector $\mathbf{c}$ (for those $\mathbf{c}$ for which the primal has finite cost).

(g) If the cost vector is an affine function of a scalar parameter θ, there is a systematic procedure (parametric programming) for solving the problem for all values of θ. A similar procedure is possible if the vector $\mathbf{b}$ is an affine function of a scalar parameter.

5.7 Exercises

Exercise 5.1 Consider the same problem as in Example 5.1, for which we already have an optimal basis. Let us introduce the additional constraint $x_1 + x_2 = 3$. Form the auxiliary problem described in the text, and solve it using the primal simplex method. Whenever the "large" constant M is compared to another number, M should be treated as being the larger one.

Exercise 5.2 (Sensitivity with respect to changes in a basic column of A) In this problem (and the next two) we study the change in the value of the optimal cost when an entry of the matrix $\mathbf{A}$ is perturbed by a small amount. We consider a linear programming problem in standard form, under the usual assumption that $\mathbf{A}$ has linearly independent rows. Suppose that we have an optimal basis $\mathbf{B}$ that leads to a nondegenerate optimal solution $\mathbf{x}^*$, and a nondegenerate dual optimal solution $\mathbf{p}$. We assume that the first column is basic. We will now change the first entry of $\mathbf{A}_1$ from a_{11} to $a_{11} + \delta$, where δ is a small scalar. Let $\mathbf{E}$ be a matrix of dimensions $m \times m$ (where m is the number of rows of $\mathbf{A}$), whose entries are all zero except for the top left entry e_{11}, which is equal to 1.

(a) Show that if δ is small enough, $\mathbf{B}+\delta\mathbf{E}$ is a basis matrix for the new problem.

(b) Show that under the basis $\mathbf{B} + \delta\mathbf{E}$, the vector $\mathbf{x}_B$ of basic variables in the new problem is equal to $(\mathbf{I} + \delta\mathbf{B}^{-1}\mathbf{E})^{-1}\mathbf{B}^{-1}\mathbf{b}$.

(c) Show that if δ is sufficiently small, $\mathbf{B} + \delta\mathbf{E}$ is an optimal basis for the new problem.

(d) We use the symbol $\approx$ to denote equality when second order terms in δ are ignored. The following approximation is known to be true: $(\mathbf{I}+\delta\mathbf{B}^{-1}\mathbf{E})^{-1} \approx \mathbf{I} - \delta\mathbf{B}^{-1}\mathbf{E}$. Using this approximation, show that

$$\mathbf{c}_B'\mathbf{x}_B \approx \mathbf{c}'\mathbf{x}^* - \delta p_1 x_1^*,$$

where x_1^* (respectively, p_1) is the first component of the optimal solution to the original primal (respectively, dual) problem, and $\mathbf{x}_B$ has been defined in part (b).

Exercise 5.3 (Sensitivity with respect to changes in a basic column of A) Consider a linear programming problem in standard form under the usual assumption that the rows of the matrix $\mathbf{A}$ are linearly independent. Suppose that the columns $\mathbf{A}_1, \ldots, \mathbf{A}_m$ form an optimal basis. Let $\mathbf{A}_0$ be some vector and suppose that we change $\mathbf{A}_1$ to $\mathbf{A}_1 + \delta\mathbf{A}_0$. Consider the matrix $\mathbf{B}(\delta)$ consisting of

the columns $\mathbf{A}_0 + \delta\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_m$. Let $[\delta_1, \delta_2]$ be a closed interval of values of δ that contains zero and in which the determinant of $\mathbf{B}(\delta)$ is nonzero. Show that the subset of $[\delta_1, \delta_2]$ for which $\mathbf{B}(\delta)$ is an optimal basis is also a closed interval.

Exercise 5.4 Consider the problem in Example 5.1, with a_{11} changed from 3 to $3 + \delta$. Let us keep x_1 and x_2 as the basic variables and let $\mathbf{B}(\delta)$ be the corresponding basis matrix, as a function of δ.

(a) Compute $\mathbf{B}(\delta)^{-1}\mathbf{b}$. For which values of δ is $\mathbf{B}(\delta)$ a feasible basis?

(b) Compute $\mathbf{c}_B'\mathbf{B}(\delta)^{-1}$. For which values of δ is $\mathbf{B}(\delta)$ an optimal basis?

(c) Determine the optimal cost, as a function of δ, when δ is restricted to those values for which $\mathbf{B}(\delta)$ is an optimal basis matrix.

Exercise 5.5 While solving a standard form linear programming problem using the simplex method, we arrive at the following tableau:

		x_1	x_2	x_3	x_4	x_5
		0	0	$\overline{c}_3$	0	$\overline{c}_5$
$x_2 =$	1	0	1	-1	0	β
$x_4 =$	2	0	0	2	1	γ
$x_1 =$	3	1	0	4	0	δ

Suppose also that the last three columns of the matrix $\mathbf{A}$ form an identity matrix.

(a) Give necessary and sufficient conditions for the basis described by this tableau to be optimal (in terms of the coefficients in the tableau).

(b) Assume that this basis is optimal and that $\overline{c}_3 = 0$. Find an optimal basic feasible solution, other than the one described by this tableau.

(c) Suppose that $\gamma \geq 0$. Show that there exists an optimal basic feasible solution, regardless of the values of $\overline{c}_3$ and $\overline{c}_5$.

(d) Assume that the basis associated with this tableau is optimal. Suppose also that b_1 in the original problem is replaced by $b_1 + \epsilon$. Give upper and lower bounds on ϵ so that this basis remains optimal.

(e) Assume that the basis associated with this tableau is optimal. Suppose also that c_1 in the original problem is replaced by $c_1 + \epsilon$. Give upper and lower bounds on ϵ so that this basis remains optimal.

Exercise 5.6 Company A has agreed to supply the following quantities of special lamps to Company B during the next 4 months:

Month	January	February	March	April
Units	150	160	225	180

Company A can produce a maximum of 160 lamps per month at a cost of \$35 per unit. Additional lamps can be purchased from Company C at a cost of \$50

per lamp. Company A incurs an inventory holding cost of \$5 per month for each lamp held in inventory.

(a) Formulate the problem that Company A is facing as a linear programming problem.

(b) Solve the problem using a linear programming package.

(c) Company A is considering some preventive maintenance during one of the first three months. If maintenance is scheduled for January, the company can manufacture only 151 units (instead of 160); similarly, the maximum possible production if maintenance is scheduled for February or March is 153 and 155 units, respectively. What maintenance schedule would you recommend and why?

(d) Company D has offered to supply up to 50 lamps (total) to Company A during either January, February or March. Company D charges \$45 per lamp. Should Company A buy lamps from Company D? If yes, when and how many lamps should Company A purchase, and what is the impact of this decision on the total cost?

(e) Company C has offered to lower the price of units supplied to Company A during February. What is the maximum decrease that would make this offer attractive to Company A?

(f) Because of anticipated increases in interest rates, the holding cost per lamp is expected to increase to \$8 per unit in February. How does this change affect the total cost and the optimal solution?

(g) Company B has just informed Company A that it requires only 90 units in January (instead of 150 requested previously). Calculate upper and lower bounds on the impact of this order on the optimal cost using information from the optimal solution to the original problem.

Exercise 5.7 A paper company manufactures three basic products: pads of paper, 5-packs of paper, and 20-packs of paper. The pad of paper consists of a single pad of 25 sheets of lined paper. The 5-pack consists of 5 pads of paper, together with a small notebook. The 20-pack of paper consists of 20 pads of paper, together with a large notebook. The small and large notebooks are not sold separately.

Production of each pad of paper requires 1 minute of paper-machine time, 1 minute of supervisory time, and \$.10 in direct costs. Production of each small notebook takes 2 minutes of paper-machine time, 45 seconds of supervisory time, and \$.20 in direct cost. Production of each large notebook takes 3 minutes of paper machine time, 30 seconds of supervisory time and \$.30 in direct costs. To package the 5-pack takes 1 minute of packager's time and 1 minute of supervisory time. To package the 20-pack takes 3 minutes of packager's time and 2 minutes of supervisory time. The amounts of available paper-machine time, supervisory time, and packager's time are constants b_1, b_2, b_3, respectively. Any of the three products can be sold to retailers in any quantity at the prices \$.30, \$1.60, and \$7.00, respectively.

Provide a linear programming formulation of the problem of determining an optimal mix of the three products. (You may ignore the constraint that only integer quantities can be produced.) Try to formulate the problem in such a

way that the following questions can be answered by looking at a single dual variable or reduced cost in the final tableau. Also, for each question, give a brief explanation of why it can be answered by looking at just one dual price or reduced cost.

(a) What is the marginal value of an extra unit of supervisory time?

(b) What is the lowest price at which it is worthwhile to produce single pads of paper for sale?

(c) Suppose that part-time supervisors can be hired at \$8 per hour. Is it worthwhile to hire any?

(d) Suppose that the direct cost of producing pads of paper increases from \$.10 to \$.12. What is the profit decrease?

Exercise 5.8 A pottery manufacturer can make four different types of dining room service sets: JJP English, Currier, Primrose, and Bluetail. Furthermore, Primrose can be made by two different methods. Each set uses clay, enamel, dry room time, and kiln time, and results in a profit shown in Table 5.3. (Here, lbs is the abbreviation for pounds).

Resources	**E**	**C**	**P_1**	**P_2**	**B**	**Total**
Clay (lbs)	10	15	10	10	20	130
Enamel (lbs)	1	2	2	1	1	13
Dry room (hours)	3	1	6	6	3	45
Kiln (hours)	2	4	2	5	3	23
Profit	51	102	66	66	89	

Table 5.3: The rightmost column in the table gives the manufacturer's resource availability for the remainder of the week. Notice that Primrose can be made by two different methods. They both use the same amount of clay (10 lbs.) and dry room time (6 hours). But the second method uses one pound less of enamel and three more hours in the kiln.

The manufacturer is currently committed to making the same amount of Primrose using methods 1 and 2. The formulation of the profit maximization problem is given below. The decision variables E, C, P_1, P_2, B are the number of sets of type English, Currier, Primrose Method 1, Primrose Method 2, and Bluetail, respectively. We assume, for the purposes of this problem, that the number of sets of each type can be fractional.

$$
\begin{array}{lrcrcrcrcl}
\text{maximize} & 51E & + & 102C & + & 66P_1 & + & 66P_2 & + & 89B & \\
\text{subject to} & 10E & + & 15C & + & 10P_1 & + & 10P_2 & + & 20B & \leq 130 \\
& E & + & 2C & + & 2P_1 & + & P_2 & + & B & \leq 13 \\
& 3E & + & C & + & 6P_1 & + & 6P_2 & + & 3B & \leq 45 \\
& 2E & + & 4C & + & 2P_1 & + & 5P_2 & + & 3B & \leq 23 \\
& & & & & P_1 & - & P_2 & = & 0 & \\
& E, C, P_1, P_2, B \geq 0. & & & & & & & & &
\end{array}
$$

The optimal solution to the primal and the dual, respectively, together with sensitivity information, is given in Tables 5.4 and 5.5. Use this information to answer the questions that follow.

	Optimal Value	Reduced Cost	Objective Coefficient	Allowable Increase	Allowable Decrease
E	0	−3.571	51	3.571	∞
C	2	0	102	16.667	12.5
$\mathbf{P_1}$	0	0	66	37.571	∞
$\mathbf{P_2}$	0	−37.571	66	37.571	∞
B	5	0	89	47	12.5

Table 5.4: The optimal primal solution and its sensitivity with respect to changes in coefficients of the objective function. The last two columns describe the allowed changes in these coefficients for which the same solution remains optimal.

(a) What is the optimal quantity of each service set, and what is the total profit?

(b) Give an economic (not mathematical) interpretation of the optimal dual variables appearing in the sensitivity report, for each of the five constraints.

(c) Should the manufacturer buy an additional 20 lbs. of Clay at \$1.1 per pound?

(d) Suppose that the number of hours available in the dry room decreases by 30. Give a bound for the decrease in the total profit.

(e) In the current model, the number of Primrose produced using method 1 was required to be the same as the number of Primrose produced by method 2. Consider a revision of the model in which this constraint is replaced by the constraint $P_1 - P_2 \geq 0$. In the reformulated problem would the amount of Primrose made by method 1 be positive?

Exercise 5.9 Using the notation of Section 5.2, show that for any positive scalar λ and any $\mathbf{b} \in S$, we have $F(\lambda \mathbf{b}) = \lambda F(\mathbf{b})$. Assume that the dual feasible set is nonempty, so that $F(\mathbf{b})$ is finite.

	Slack Value	Dual Variable	Constr. RHS	Allowable Increase	Allowable Decrease
Clay	130	1.429	130	23.33	43.75
Enamel	9	0	13	∞	4
Dry Rm.	17	0	45	∞	28
Kiln	23	20.143	23	5.60	3.50
Prim.	0	11.429	0	3.50	0

Table 5.5: The optimal dual solution and its sensitivity. The column labeled "slack value" gives us the optimal values of the slack variables associated with each of the primal constraints. The third column simply repeats the right-hand side vector $\mathbf{b}$, while the last two columns describe the allowed changes in the components of $\mathbf{b}$ for which the optimal dual solution remains the same.

Exercise 5.10 Consider the linear programming problem:

$$\begin{aligned} \text{minimize} \quad & x_1 + x_2 \\ \text{subject to} \quad & x_1 + 2x_2 = \theta, \\ & x_1, x_2 \geq 0. \end{aligned}$$

(a) Find (by inspection) an optimal solution, as a function of θ.

(b) Draw a graph showing the optimal cost as a function of θ.

(c) Use the picture in part (b) to obtain the set of all dual optimal solutions, for every value of θ.

Exercise 5.11 Consider the function $g(\theta)$, as defined in the beginning of Section 5.5. Suppose that $g(\theta)$ is linear for $\theta \in [\theta_1, \theta_2]$. Is it true that there exists a unique optimal solution when $\theta_1 < \theta < \theta_2$? Prove or provide a counterexample.

Exercise 5.12 Consider the parametric programming problem discussed in Section 5.5.

(a) Suppose that for some value of θ, there are exactly two distinct basic feasible solutions that are optimal. Show that they must be adjacent.

(b) Let θ^* be a breakpoint of the function $g(\theta)$. Let $\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3$ be basic feasible solutions, all of which are optimal for $\theta = \theta^*$. Suppose that $\mathbf{x}^1$ is a unique optimal solution for $\theta < \theta^*$, $\mathbf{x}^3$ is a unique optimal solution for $\theta > \theta^*$, and $\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3$ are the only optimal basic feasible solutions for $\theta = \theta^*$. Provide an example to show that $\mathbf{x}^1$ and $\mathbf{x}^3$ need not be adjacent.

Exercise 5.13 Consider the following linear programming problem:

$$\begin{array}{lrcrcrcl} \text{minimize} & 4x_1 & & & + & 5x_3 & & \\ \text{subject to} & 2x_1 & + & x_2 & - & 5x_3 & = & 1 \\ & -3x_1 & & & + & 4x_3 & + & x_4 = 2 \\ & & & x_1, x_2, x_3, x_4 \geq 0. & & & & \end{array}$$

(a) Write down a simplex tableau and find an optimal solution. Is it unique?

(b) Write down the dual problem and find an optimal solution. Is it unique?

(c) Suppose now that we change the vector $\mathbf{b}$ from $\mathbf{b} = (1, 2)$ to $\mathbf{b} = (1 - 2\theta, 2 - 3\theta)$, where θ is a scalar parameter. Find an optimal solution and the value of the optimal cost, as a function of θ. (For all θ, both positive and negative.)

Exercise 5.14 Consider the problem

$$\begin{array}{lrcl} \text{minimize} & (\mathbf{c} + \theta\mathbf{d})'\mathbf{x} & & \\ \text{subject to} & \mathbf{A}\mathbf{x} & = & \mathbf{b} + \theta\mathbf{f} \\ & \mathbf{x} & \geq & \mathbf{0}, \end{array}$$

where $\mathbf{A}$ is an $m \times n$ matrix with linearly independent rows. We assume that the problem is feasible and the optimal cost $f(\theta)$ is finite for all values of θ in some interval $[\theta_1, \theta_2]$.

(a) Suppose that a certain basis is optimal for $\theta = -10$ and for $\theta = 10$. Prove that the same basis is optimal for $\theta = 5$.

(b) Show that $f(\theta)$ is a piecewise quadratic function of θ. Give an upper bound on the number of "pieces."

(c) Let $\mathbf{b} = \mathbf{0}$ and $\mathbf{c} = \mathbf{0}$. Suppose that a certain basis is optimal for $\theta = 1$. For what other nonnegative values of θ is that same basis optimal?

(d) Is $f(\theta)$ convex, concave or neither?

Exercise 5.15 Consider the problem

$$\begin{array}{lrcl} \text{minimize} & \mathbf{c}'\mathbf{x} & & \\ \text{subject to} & \mathbf{A}\mathbf{x} & = & \mathbf{b} + \theta\mathbf{d} \\ & \mathbf{x} & \geq & \mathbf{0}, \end{array}$$

and let $f(\theta)$ be the optimal cost, as a function of θ.

(a) Let $X(\theta)$ be the set of all optimal solutions, for a given value of θ. For any nonnegative scalar t, define $X(0, t)$ to be the union of the sets $X(\theta)$, $0 \leq \theta \leq t$. Is $X(0, t)$ a convex set? Provide a proof or a counterexample.

(b) Suppose that we remove the nonnegativity constraints $\mathbf{x} \geq \mathbf{0}$ from the problem under consideration. Is $X(0, t)$ a convex set? Provide a proof or a counterexample.

(c) Suppose that $\mathbf{x}^1$ and $\mathbf{x}^2$ belong to $X(0, t)$. Show that there is a continuous path from $\mathbf{x}^1$ to $\mathbf{x}^2$ that is contained within $X(0, t)$. That is, there exists a continuous function $g(\lambda)$ such that $g(\lambda_1) = \mathbf{x}^1$, $g(\lambda_2) = \mathbf{x}^2$, and $g(\lambda) \in X(0, t)$ for all $\lambda \in (\lambda_1, \lambda_2)$.

Exercise 5.16 Consider the parametric programming problem of Section 5.5. Suppose that some basic feasible solution is optimal if and only if θ is equal to some θ^*.

(a) Suppose that the feasible set is unbounded. Is it true that there exist at least three distinct basic feasible solutions that are optimal when $\theta = \theta^*$?

(b) Answer the question in part (a) for the case where the feasible set is bounded.

Exercise 5.17 Consider the parametric programming problem. Suppose that every basic solution encountered by the algorithm is nondegenerate. Prove that the algorithm does not cycle.

5.8 Notes and sources

The material in this chapter, with the exception of Section 5.3, is standard, and can be found in any text on linear programming.

5.1. A more detailed discussion of the results of the production planning case study can be found in Freund and Shannahan (1992).

5.3. The results in this section have beautiful generalizations to the case of nonlinear convex optimization; see, e.g., Rockafellar (1970).

5.5. Anticycling rules for parametric programming can be found in Murty (1983).

References

AHUJA, R. K., T. L. MAGNANTI, and J. B. ORLIN. 1993. *Network Flows*, Prentice Hall, Englewood Cliffs, NJ.

ANDERSEN, E., J. GONDZIIO, C. MESZAROS, and X. XU. 1996. Implementation of interior point methods for large scale linear programming, in *Interior point methods in mathematical programming*, T. Terlaky (ed.), Kluwer Academic Publisher, Boston, MA.

APPLEGATE, D., and W. COOK. 1991. A computational study of the job shop scheduling problem, *ORSA Journal on Computing*, **3**, 149-156.

BALAS, E., S. CERIA, G. CORNUÉJOLS, and N. NATRAJ. 1995. Gomory cuts revisited, working paper, Carnegie-Mellon University, Pittsburgh, PA.

BALAS, E., S. CERIA, and G. CORNUÉJOLS. 1995. Mixed $0-1$ programming by lift-and-project in a branch and cut environment, working paper, Carnegie-Mellon University, Pittsburgh, PA.

BARAHONA, F., and É. TARDOS. 1989. Note on Weintraub's minimum cost circulation algorithm, *SIAM Journal on Computing*, **18**, 579-583.

BARNES, E. R. 1986. A variation on Karmarkar's algorithm for solving linear programming problems, *Mathematical Programming*, **36**, 174-182.

BARR, R. S., F. GLOVER, and D. KLINGMAN. 1977. The alternating path basis algorithm for the assignment problem, *Mathematical Programming,* **13**, 1-13.

BARTHOLDI, J. J., J. B. ORLIN, and H. D. RATLIFF. 1980. Cyclic scheduling via integer programs with circular ones, *Operations Research*, **28**, 1074-1085.

BAZARAA, M. S., J. J. JARVIS, and H. D. SHERALI. 1990. *Linear Programming and Network Flows*, 2nd edition, Wiley, New York, NY.

BEALE, E. M. L. 1955. Cycling in the dual simplex algorithm, *Naval Research Logistics Quarterly,* **2**, 269-275.

BELLMAN, R. E. 1958. On a routing problem, *Quarterly of Applied Mathematics*, **16**, 87-90.

BENDERS, J. F. 1962. Partitioning procedures for solving mixed-variables programming problems, *Numerische Mathematik*, **4**, 238-252.

BERTSEKAS, D. P. 1979. A distributed algorithm for the assignment problem, working paper, Laboratory for Information and Decision Systems, M.I.T., Cambridge, MA.

BERTSEKAS, D. P. 1981. A new algorithm for the assignment problem. *Mathematical Programming*, **21**, 152-171.

BERTSEKAS, D. P. 1991. *Linear Network Optimization,* M.I.T. Press, Cambridge, MA.

BERTSEKAS, D. P. 1995a. *Dynamic Programming and Optimal Control,* Athena Scientific, Belmont, MA.

BERTSEKAS, D. P. 1995b. *Nonlinear Programming,* Athena Scientific, Belmont, MA.

BERTSEKAS, D. P., and J. N. TSITSIKLIS. 1989. *Parallel and Distributed Computation: Numerical Methods*, Prentice Hall, Englewood Cliffs, NJ.

BERTSIMAS, D., and L. HSU. 1997. A branch and cut algorithm for the job shop scheduling problem, working paper, Operations Research Center, M.I.T., Cambridge, MA.

BERTSIMAS, D., and X. LUO. 1997. On the worst case complexity of potential reduction algorithms for linear programming, *Mathematical Programming*, to appear.

BERTSIMAS, D., and S. STOCK. 1997. The air traffic flow management problem with enroute capacities, *Operations Research*, to appear.

BLAND, R. G. 1977. New finite pivoting rules for the simplex method, *Mathematics of Operations Research*, **2**, 103-107.

BLAND, R. G., D. GOLDFARB, and M. J. TODD. 1981. The ellipsoid method: a survey, *Operations Research*, **29**, 1039-1091.

BORGWARDT, K.-H. 1982. The average number of pivot steps required by the simplex-method is polynomial, *Zeitschrift für Operations Research,* **26**, 157-177.

BOYD, S., and L. VANDENBERGHE. 1995. *Introduction to convex optimization with engineering applications*, lecture notes, Stanford University, Stanford, CA.

BRADLEY, S. P., A. C. HAX, and T. L. MAGNANTI. 1977. *Applied Mathematical Programming*, Addison-Wesley, Reading, MA.

CARATHÉODORY, C. 1907. Über den Variabilitätsbereich der Koeffizienten von Potenzreihen, die gegebene Werte nicht annehmen, *Mathematische Annalen*, **64**, 95-115.

CERIA, S., C. CORDIER, H. MARCHAND, and L. A. WOLSEY. 1995. Cutting planes for integer programs with general integer variables, working paper, Columbia University, New York, NY.

CHARNES, A. 1952. Optimality and degeneracy in linear programming, *Econometrica,* **20**, 160-170.

CHRISTOFIDES, N. 1975. Worst-case analysis of a new heuristic for the traveling salesman problem, Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA.

CHVÁTAL, V. 1983. *Linear Programming*, W. H. Freeman, New York, NY.

CLARK, F. E. 1961. Remark on the constraint sets in linear programming, *American Mathematical Monthly,* **68**, 351-352.

COBHAM, A. 1965. The intrinsic computational difficulty of functions, in *Logic, Methodology and Philosophy of Science*, Y. Bar-Hillel (ed.), North-Holland, Amsterdam, The Netherlands, 24-30.

COOK, S. A. 1971. The complexity of theorem proving procedures, in *Proceedings of the 3rd ACM Symposium on the Theory of Computing*, 151-158.

CORMEN, T. H., C. E. LEISERSON, and R. L. RIVEST. 1990. *Introduction to Algorithms,* McGraw-Hill, New York, NY.

CUNNINGHAM, W. H. 1976. A network simplex method, *Mathematical Programming*, **11**, 105-116.

DAHLEH, M. A., and I. DIAZ-BOBILLO. 1995. *Control of Uncertain Systems: A Linear Programming Approach*, Prentice Hall, Englewood Cliffs, NJ.

DANTZIG, G. B. 1951. Application of the simplex method to a transportation problem, in *Activity Analysis of Production and Allocation*, T. C. Koopmans (ed.), Wiley, New York, NY, 359-373.

DANTZIG, G. B. 1963. *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ.

DANTZIG, G. B. 1992. An ϵ-precise feasible solution to a linear program with a convexity constraint in $1/\epsilon^2$ iterations independent of problem size, working paper, Stanford University, Stanford, CA.

DANTZIG, G. B., A. ORDEN, and P. WOLFE. 1955. The generalized simplex method for minimizing a linear form under linear inequality constraints, *Pacific Journal of Mathematics,* **5**, 183-195.

DANTZIG, G. B., and P. WOLFE. 1960. The decomposition principle for linear programs, *Operations Research*, **8**, 101-111.

DIJKSTRA, E. 1959. A note on two problems in connexion with graphs, *Numerische Mathematik*, **1**, 269-271.

DIKIN, I. I. 1967. Iterative solutions of problems of linear and quadratic programming, *Soviet Mathematics Doklady*, **8**, 674-675.

DIKIN, I. I. 1974. On the convergence of an iterative process, *Upravlyaemye Sistemi*, **12**, 54-60. (In Russian.)

DINES, L. L. 1918. Systems of linear inequalities, *Annals of Mathematics*, **20**, 191-199.

DUDA, R. O., and P. E. HART. 1973. *Pattern Classification and Scene Analysis*, Wiley, New York, NY.

EDMONDS, J. 1965a. Paths, trees, and flowers, *Canadian Journal of Mathematics*, **17**, 449-467.

EDMONDS, J. 1965b. Maximum matching and a polyhedron with $0-1$ vertices, *Journal of Research of the National Bureau of Standards*, **69B**, 125-130.

EDMONDS, J. 1971. Matroids and the greedy algorithm, *Mathematical Programming*, **1**, 127-136.

EDMONDS, J., and R. M. KARP. 1972. Theoretical improvements in algorithmic efficiency for network flow problems, *Journal of the ACM*, **19**, 248-264.

ELIAS, P., A. FEINSTEIN, and C. E. SHANNON. 1956. Note on maximum flow through a network, *IRE Transactions on Information Theory,* **2**, 117-119.

FARKAS, G. 1894. On the applications of the mechanical principle of Fourier, *Mathematikai és Természettudományi Értesitö*, **12**, 457-472. (In Hungarian.)

FIACCO, A. V., and G. P. MCCORMICK. 1968. *Nonlinear programming: sequential unconstrained minimization techniques*, Wiley, New York, NY.

FEDERGRUEN, A., and H. GROENEVELT. 1986. Preemptive scheduling of uniform machines by ordinary network flow techniques, *Management Science*, **32**, 341-349.

FISHER, H., and G. L. THOMPSON. 1963. Probabilistic learning combinations of local job shop scheduling rules, in *Industrial Scheduling*, J. F. Muth and G. L. Thompson (eds.), Prentice Hall, Englewood Cliffs, NJ, 225-251.

FLOYD, R. W. 1962. Algorithm 97: shortest path. *Communications of ACM*, **5**, 345.

FORD, L. R. 1956. Network flow theory, report P-923, Rand Corp., Santa Monica, CA.

FORD, L. R., and D. R. FULKERSON. 1956a. Maximal flow through a network, *Canadian Journal of Mathematics,* **8**, 399-404.

FORD, L. R., and D. R. FULKERSON. 1956b. Solving the transportation problem, *Management Science*, **3**, 24-32.

FORD, L. R., and D. R. FULKERSON. 1962. *Flows in Networks,* Princeton University Press, Princeton, NJ.

FOURIER. J. B. J. 1827. Analyse des Travaux de l'Académie Royale des Sciences, pendant l'année 1824, Partie mathématique, *Histoire de l'Académie Royale des Sciences de l'Institut de France*, **7**, xlvii-lv.

FREUND, R. M. 1991. Polynomial-time algorithms for linear programming based only on primal affine scaling and projected gradients of a potential function, *Mathematical Programming*, **51**, 203-222.

FREUND, R. M., and B. SHANNAHAN. 1992. Short-run manufacturing problems at DEC, report, Sloan School of Management, M.I.T., Cambridge, MA.

FRISCH, M. R. 1956. La résolution des problèmes de programme linéaire par la méthode du potential logarithmique, *Cahiers du Séminaire D' Econometrie*, **4**, 7-20.

FULKERSON, D. R., and G. B. DANTZIG. 1955. Computation of maximum flow in networks, *Naval Research Logistics Quarterly,* **2**, 277-283.

GALE, D., H. W. KUHN, and A. W. TUCKER. 1951. Linear programming and the theory of games, in *Activity Analysis of Production and Allocation,* T. C. Koopmans (ed.), Wiley, New York, NY, 317-329.

GALE, D., and L. S. SHAPLEY. 1962. College admissions and the stability of marriage, *American Mathematical Monthly*, **69**, 9-15.

GAREY, M. R., and D. S. JOHNSON. 1979. *Computers and Intractability: a Guide to the Theory of NP-completeness*, W. H. Freeman, New York, NY.

GEMAN, S. and D. GEMAN. 1984. Stochastic Relaxation, Gibbs distribution, and the Bayesian restoration of images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**, 721-741.

GILL, P. E., W. MURRAY, and M. H. WRIGHT. 1981. Practical Optimization, Academic Press, New York, NY.

GILMORE, P. C., and R. E. GOMORY. 1961. A linear programming approach to the cutting stock problem, *Operations Research*, **9**, 849-859.

GILMORE, P. C., and R. E. GOMORY. 1963. A linear programming approach to the cutting stock problem – part II, *Operations Research*, **11**, 863-888.

GOEMANS, M., and D. BERTSIMAS. 1993. Survivable networks, LP relaxations and the parsimonious property, *Mathematical Programming*, **60**, 145-166.

GOEMANS, M., and D. WILLIAMSON. 1993. A new 3/4 approximation algorithm for MAX SAT, in *Proceedings of the 3rd International Conference in Integer Programming and Combinatorial Optimization*, 313-321.

GOLDBERG, A. V., and R. E. TARJAN. 1988. A new approach to the maximum flow problem, *Journal of the ACM*, **35**, 921-940.

GOLDBERG, A. V., and R. E. TARJAN. 1989. Finding minimum-cost circulations by cancelling negative cycles, *Journal of the ACM*, **36**, 873-886.

GOLDFARB, D., and J. K. REID. 1977. A practicable steepest-edge simplex algorithm, *Mathematical Programming*, **12**, 361-371.

GOLUB, G. H., and C. F. VAN LOAN. 1983. *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD.

GOMORY, R. E. 1958. Outline of an algorithm for integer solutions to linear programs, *Bulletin of the American Mathematical Society*, **64**, 275-278.

GONZAGA, C. 1989. An algorithm for solving linear programming in $O(n^3L)$ operations, in *Progress in Mathematical Programming*, N. Megiddo (ed.), Springer-Verlag, New York, NY, 1-28.

GONZAGA, C. 1990. Polynomial affine algorithms for linear programming, *Mathematical Programming*, **49**, 7-21.

GRÖTSCHEL, M., L. LOVÁSZ, and A. SCHRIJVER. 1981. The ellipsoid method and its consequences in combinatorial optimization, *Combinatorica*, **1**, 169-197.

GRÖTSCHEL, M., L. LOVÁSZ, and A. SCHRIJVER. 1988. *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, New York, NY.

HAIMOVICH, M. 1983. The simplex method is very good! – on the expected number of pivot steps and related properties of random linear programs, preprint.

HAJEK, B. 1988. Cooling schedules for optimal annealing, *Mathematics of Operations Research*, **13**, 311-329.

HALL, L. A., and R. J. VANDERBEI. 1993. Two thirds is sharp for affine scaling, *Operations Research Letters*, **13**, 197-201.

HALL, L. A., A. S. SCHULTZ, D. B. SHMOYS, and J. WEIN. 1996. Scheduling to minimize average completion time; off-line and on-line approximation algorithms, working paper, Johns Hopkins University, Baltimore, MD.

HANE , C. A., C. BARNHART, E. L. JOHNSON, R. E. MARSTEN, G. L. NEMHAUSER, and G. SIGISMONDI. 1995. The fleet assignment problem: solving a large-scale integer program, *Mathematical Programming*, **70**, 211-232.

HARRIS, P. M. J. 1973. Pivot selection methods of the Devex LP code, *Mathematical Programming*, **5**, 1-28.

HAYKIN, S. 1994. *Neural Networks: A Comprehensive Foundation*, McMillan, New York, NY.

HELD, M., and R. M. KARP. 1962. A dynamic programming approach to sequencing problems, *SIAM Journal on Applied Mathematics*, **10**, 196-210.

HELD, M., and R. M. KARP. 1970. The traveling salesman problem and minimum spanning trees, *Operations Research*, **18**, 1138-1162.

HELD, M., and R. M. KARP. 1971. The traveling salesman problem and minimum spanning trees: part II, *Mathematical Programming*, **1**, 6-25.

HELLY, E. 1923. Über Mengen konvexer Körper mit gemeinschaftlichen Punkten, *Jahresbericht Deutsche Mathematische Vereinungen,* **32**, 175-176.

HOCHBAUM, D. (ed.). 1996. *Approximation algorithms for NP-hard problems*, Kluwer Academic Publishers, Boston, MA.

HU, T. C. 1969. *Integer Programming and Network Flows,* Addison-Wesley, Reading, MA.

IBARRA, O. H., and C. E. KIM. 1975. Fast approximation algorithms for the knapsack and sum of subset problems, *Journal of the ACM*, **22**, 463-468.

INFANGER, G. 1993. *Planning under uncertainty: solving large-scale stochastic linear programs*, Boyd & Fraser, Danvers, MA.

JOHNSON, D. S., C. ARAGON, L. MCGEOCH, and C. SCHEVON. 1990. Optimization by simulated annealing: an experimental evaluation, part I: graph partitioning, *Operations Research*, **37**, 865-892.

JOHNSON, D. S., C. ARAGON, L. MCGEOCH, and C. SCHEVON. 1992. Optimization by simulated annealing: an experimental evaluation, part II: graph coloring and number partitioning, *Operations Research*, **39**, 378-406.

KALAI, G., and D. KLEITMAN. 1992. A quasi-polynomial bound for the diameter of graphs of polyhedra, *Bulletin of the American Mathematical Society*, **26**, 315-316.

KALL, P., and S. W. WALLACE. 1994. *Stochastic Programming*, Wiley, New York, NY.

KARMARKAR, N. 1984. A new polynomial-time algorithm for linear programming, *Combinatorica*, **4**, 373-395.

KARP, R. M. 1972. Reducibility among combinatorial problems, in *Complexity of Computer Computations*, R. E. Miller and J. W. Thacher (eds.), Plenum Press, New York, NY, 85-103.

KARP, R. M. 1978. A characterization of the minimum cycle mean in a digraph, *Discrete Mathematics*, **23**, 309-311.

KARP, R. M., and C. H. PAPADIMITRIOU. 1982. On linear characterizations of combinatorial optimization problems, *SIAM Journal on Computing*, **11**, 620-632.

KHACHIAN, L. G. 1979. A polynomial algorithm in linear programming, *Soviet Mathematics Doklady*, **20**, 191-194.

KIRKPATRICK, S., C. D. GELATT, JR., and M. P. VECCHI. 1983. Optimization by simulated annealing, *Science*, **220**, 671-680.

KLEE, V., and G. J. MINTY. 1972. How good is the simplex algorithm?, in *Inequalities – III*, O. Shisha (ed.), Academic Press, New York, NY, 159-175.

KLEE, V., and D. W. WALKUP. 1967. The d-step conjecture for polyhedra of dimension $d < 6$, *Acta Mathematica*, **117**, 53-78.

KLEIN, M. 1967. A primal method for minimal cost flows with application to the assignment and transportation problems, *Management Science*, **14**, 205-220.

KOJIMA, M., S. MIZUNO, and A. YOSHISE. 1989. A primal-dual interior point algorithm for linear programming, in *Progress in Mathematical Programming*, N. Megiddo (ed.), Springer-Verlag, New York, NY, 29-47.

KUHN, H. W. 1955. The Hungarian method for the assignment problem, *Naval Research Logistics Quarterly*, **2**, 83-97.

LAWLER, E. L. 1976. *Combinatorial Optimization: Networks and Matroids,* Holt, Rinehart, and Winston, New York, NY.

LAWLER, E. L., J. K. LENSTRA, A. H. G. RINNOOY KAN, and D. B. SHMOYS (eds.). 1985. *The Traveling Salesman Problem: a Guided Tour of Combinatorial Optimization*, Wiley, New York, NY.

LENSTRA, J. K., A. H. G. RINNOOY KAN, and A. SCHRIJVER (eds.). 1991. *History of Mathematical Programming: A Collection of Personal Reminiscences,* Elsevier, Amsterdam, The Netherlands.

LEVIN, A. Y. 1965. On an algorithm for the minimization of convex functions, *Soviet Mathematics Doklady*, **6**, 286-290.

LEVIN, L. A. 1973. Universal sorting problems, *Problemy Peredachi Informatsii*, **9**, 265-266. (In Russian.)

LEWIS, H. R., and C. H. PAPADIMITRIOU. 1981. *Elements of the Theory of Computation*, Prentice Hall, Englewood Cliffs, NJ.

LUENBERGER, D. G. 1969. *Optimization by Vector Space Methods*, Wiley, New York, NY.

LUENBERGER, D. G. 1984. *Linear and Nonlinear Programming,* 2nd ed., Addison-Wesley, Reading, MA.

LUSTIG, I., R. E. MARSTEN, and D. SHANNO. 1994. Interior point methods: computational state of the art, *ORSA Journal on Computing*, **6**, 1-14.

MAGNANTI, T. L., and L. A. WOLSEY. 1995. Optimal Trees, in *Handbook of Operations Research and Management Science, Volume 6, Network Models*, M. O. Ball, C. L. Monma, T. L. Magnanti and G. L. Nemhauser (eds.), North Holland, Amsterdam, The Netherlands, 503-615.

MARSHALL, K. T., and J. W. SUURBALLE. 1969. A note on cycling in the simplex method, *Naval Research Logistics Quarterly*, **16**, 121-137.

MARTIN, P., and D. B. SHMOYS. 1996. A new approach to computing optimal schedules for the job-shop scheduling problem, in *Proceedings of the 5th International Conference in Integer Programming and Combinatorial Optimization*, 389-403.

McShane, K. A., C. L. Monma, and D. Shanno. 1991. An implementation of a primal-dual interior point method for linear programming, *ORSA Journal on Computing*, **1**, 70-83.

Megiddo, N. 1989. Pathways to the optimal set in linear programming, in *Progress in Mathematical Programming*, N. Megiddo (ed.), Springer-Verlag, New York, NY, 131-158.

Megiddo, N., and Shub, M. 1989. Boundary behavior of interior point algorithms in linear programming, *Mathematics of Operations Research*, **14**, 97-146.

Minkowski, H. 1896. *Geometrie der Zahlen*, Teubner, Leipzig, Germany.

Mizuno, S. 1996. Infeasible interior point algorithms, in *Interior Point Algorithms in Mathematical Programming*, T. Terlaky (ed.), Kluwer Academic Publishers, Boston, MA.

Monteiro, R. D. C., and I. Adler. 1989a. Interior path following primal-dual algorithms; part I: linear programming, *Mathematical Programming*, **44**, 27-41.

Monteiro, R. D. C., and I. Adler. 1989b. Interior path following primal-dual algorithms; part II: convex quadratic programming, *Mathematical Programming*, **44**, 43-66.

Motzkin, T. S. 1936. *Beiträge zur Theorie der linearen Ungleichungen* (Inaugural Dissertation Basel), Azriel, Jerusalem.

Murty, K. G. 1983. *Linear Programming*, Wiley, New York, NY.

Nemhauser, G. L., and L. A. Wolsey. 1988. *Integer and Combinatorial Optimization,* Wiley, New York, NY.

Nesterov, Y., and A. Nemirovskii. 1994. *Interior point polynomial algorithms for convex programming*, SIAM, Studies in Applied Mathematics, **13**, Philadelphia, PA.

von Neumann, J. 1947. Discussion of a maximum problem, unpublished working paper, Institute for Advanced Studies, Princeton, NJ.

von Neumann, J. 1953. A certain zero-sum two-person game equivalent to the optimal assignment problem, in *Contributions to the Theory of Games, II*, H. W. Kuhn and A. W. Tucker (eds.), *Annals of Mathematics Studies*, **28**, Princeton University Press, Princeton, NJ, 5-12.

Orden, A. 1993. LP from the '40s to the '90s, *Interfaces*, **23**, 2-12.

Orlin, J. B. 1984. Genuinely polynomial simplex and non-simplex algorithms for the minimum cost flow problem, technical report 1615-84, Sloan School of Management, M.I.T., Cambridge, MA.

Padberg, M. W., and M. R. Rao. 1980. The Russian method and integer programming, working paper, New York University, New York, NY.

Papadimitriou, C. H. 1994. *Computational Complexity*, Addison-Wesley, Reading, MA.

Papadimitriou, C. H., and K. Steiglitz. 1982. *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NJ.

Plotkin, S., and É. Tardos. 1990. Improved dual network simplex, in *Proceedings of the First ACM-SIAM Symposium on Discrete Algorithms,* 367-376.

Poljak, B. T. 1987. *Introduction to Optimization*, Optimization Software Inc., New York, NY.

Prim, R. C. 1957. Shortest connection networks and some generalizations, *Bell System Technical Journal*, **36**, 1389-1401.

QUEYRANNE, M. 1993. Structure of a simple scheduling polyhedron, *Mathematical Programming*, **58**, 263-285.

RECSKI, A. 1989. *Matroid Theory and its Applications in Electric Network Theory and in Statics*, Springer-Verlag, New York, NY.

RENEGAR, J. 1988. A polynomial time algorithm based on Newton's method for linear programming, *Mathematical Programming*, **40**, 59-93.

ROCKAFELLAR, R. T. 1970. *Convex Analysis*, Princeton University Press, Princeton, NJ.

ROCKAFELLAR, R. T. 1984. *Network Flows and Monotropic Optimization,* Wiley, New York, NY.

ROSS, S. 1976. Risk, return, and arbitrage, in *Risk and Return in Finance*, I. Friend, and J. Bicksler (eds.), Cambridge, Ballinger, England.

ROSS, S. 1978. A simple approach to the valuation of risky streams, *Journal of Business*, **51**, 453-475.

RUDIN, W. 1976. *Real Analysis,* McGraw-Hill, New York, NY.

RUSHMEIER, R. A., and S. A. KONTOGIORGIS. 1997. Advances in the optimization of airline fleet assignment, *Transportation Science*, to appear.

SCHRIJVER, A. 1986. *Theory of Linear and Integer Programming*, Wiley, New York, NY.

SCHULTZ, A. S. 1996. Scheduling to minimize total weighted completion time: performance guarantees of LP-based heuristics and lower bounds, in *Proceedings of the 5th International Conference in Integer Programming and Combinatorial Optimization*, 301-315.

SHOR, N. Z. 1970. Utilization of the operation of space dilation in the minimization of convex functions, *Cybernetics*, **6**, 7-15.

SMALE, S. 1983. On the average number of steps in the simplex method of linear programming, *Mathematical Programming,* **27**, 241-262.

SMITH, W. E. 1956. Various optimizers for single-stage production, *Naval Research Logistics Quarterly*, **3**, 59-66.

STIGLER, G. 1945. The cost of subsistence, *Journal of Farm Economics,* **27**, 303-314.

STOCK, S. 1996. Allocation of NSF graduate fellowships, report, Sloan School of Management, M.I.T., Cambridge, MA.

STONE, R. E., and C. A. TOVEY. 1991. The simplex and projective scaling algorithms as iteratively reweighted least squares, *SIAM Review*, **33**, 220-237.

STRANG, G. 1988. *Linear Algebra and its Applications*, 3rd ed., Academic Press, New York, NY.

TARDOS, É. 1985. A strongly polynomial minimum cost circulation algorithm, *Combinatorica,* **5**, 247-255.

TEO, C. 1996. Constructing approximation algorithms via linear programming relaxations: primal dual and randomized rounding techniques, Ph.D. thesis, Operations Research Center, M.I.T., Cambridge, MA.

TSENG, P. 1989. A simple complexity proof for a polynomial-time linear programming algorithm, *Operations Research Letters*, **8**, 155-159.

TSENG, P., and Z.-Q. LUO. 1992. On the convergence of the affine scaling algorithm, *Mathematical Programming*, **56**, 301-319.

TSUCHIYA, T. 1991. Global convergence of the affine scaling methods for degenerate linear programming problems, *Mathematical Programming*, **52**, 377-404.

TSUCHIYA, T., and M. MURAMATSU. 1995. Global convergence of a long-step affine scaling algorithm for degenerate linear programming problems, *SIAM Journal on Optimization*, **5**, 525-551.

TUCKER, A. W. 1956. Dual systems of homogeneous linear relations, in *Linear Inequalities and Related Systems,* H. W. Kuhn and A. W. Tucker (eds.), Princeton University Press, Princeton, NJ, 3-18.

VANDERBEI, R. J., M. S. MEKETON, and B. A. FREEDMAN. 1986. A modification of Karmarkar's linear programming algorithm, *Algorithmica*, **1**, 395-407.

VANDERBEI, R. J., J. C. LAGARIAS. 1990. I. I. Dikin's convergence result for the affine-scaling algorithm, in *Mathematical Developments Arising from Linear Programming*, J. C. Lagarias and M. J. Todd (eds.), American Mathematical Society, Providence, RI, *Contemporary Mathematics*, **114**, 109-119.

VRANAS, P. 1996. Optimal slot allocation for European air traffic flow management, working paper, German aerospace research establishment, Berlin, Germany.

WAGNER, H. M. 1959. On a class of capacitated transportation problems, *Management Science,* **5**, 304-318.

WARSHALL, S. 1962. A theorem on boolean matrices, *Journal of the ACM*, **23**, 11-12.

WEBER, R. 1995. Personal communication.

WEINTRAUB, A. 1974. A primal algorithm to solve network flow problems with convex costs, *Management Science*, **21**, 87-97.

WILLIAMS, H. P. 1990. *Model Building in Mathematical Programming*, Wiley, New York, NY.

WILLIAMSON, D. 1994. On the design of approximation algorithms for a class of graph problems, Ph.D. thesis, Department of EECS, M.I.T., Cambridge, MA.

YE, Y. 1991. An $O(n^3L)$ potential reduction algorithm for linear programming, *Mathematical Programming*, **50**, 239-258.

YE, Y., M. J. TODD, and S. MIZUNO. 1994. An $O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm, *Mathematics of Operations Research*, **19**, 53-67.

YUDIN, D. B., and A. NEMIROVSKII. 1977. Informational complexity and efficient methods for the solution of convex extremal problems, *Matekon*, **13**, 25-45.

ZHANG, Y., and R. A. TAPIA. 1993. A superlinearly convergent polynomial primal-dual interior point algorithm for linear programming, *SIAM Journal on Optimization*, **3**, 118-133.

Index

D

Z

This book provides a unified, insightful, and modern treatment of linear optimization, that is, linear programming, network flow problems, and discrete optimization. It includes classical topics as well as the state of the art, in both theory and practice.

Among its special features, the book:

- Develops the major algorithms and duality theory through a geometric perspective
- Provides a thorough treatment of the geometry, convergence, and complexity of interior point methods
- Covers the main methods for network flow problems
- Contains a detailed treatment of integer programming formulations and algorithms
- Discusses the art of formulating and solving large-scale problems through practical case studies
- Includes a large number of examples and exercises. Has been developed through extensive classroom use in graduate courses

Dimitris Bertsimas and **John N. Tsitsiklis** are Professors at the Massachusetts Institute of Technology.

Related books of interest by Athena Scientific:

Dimitri P. Bertsekas, *Dynamic Programming and Optimal Control* (2 Vols.), 1995.

Dimitri P. Bertsekas, *Nonlinear Programming*, 1995.

Dimitri P. Bertsekas and John N. Tsitsiklis, *Neuro-Dynamic Programming*, 1996.

ISBN 1-886529-19-1

90000

9 781886 529199