# Linear and Nonlinear Optimization

# Linear and Nonlinear Optimization

SECOND EDITION

Igor Griva
Stephen G. Nash
Ariela Sofer
George Mason University
Fairfax, Virginia

siam Society for Industrial and Applied Mathematics • Philadelphia

10 9 8 7 6 5 4 3 2 1



Cover image of the Golden Gate Bridge used with permission of istockphoto.com. Cover designed by Galina Spivak.

**Library of Congress Cataloging-in-Publication Data**

Griva, Igor.
Linear and nonlinear optimization / Igor Griva, Stephen G. Nash, Ariela Sofer. -- 2nd ed.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-898716-61-0
1. Linear programming. 2. Nonlinear programming. I. Nash, Stephen (Stephen G.) II. Sofer, Ariela. III. Title.
T57.74.G75 2008
519.7'2--dc22

2008032477

# Contents

# Preface

This book provides an introduction to the applications, theory, and algorithms of linear and nonlinear optimization. The emphasis is on practical aspects—modern algorithms, as well as the influence of theory on the interpretation of solutions or on the design of software. Two important goals of this book are to present linear and nonlinear optimization in an integrated setting, and to incorporate up-to-date interior-point methods in linear and nonlinear optimization.

As an illustration of this unified approach, almost every algorithm in this book is presented in the form of a General Optimization Algorithm. This algorithm has two major steps: an optimality test, and a step that improves the estimate of the solution. This framework is general enough to encompass the simplex method and various interior-point methods for linear programming, as well as Newton's method and active-set methods for nonlinear optimization. The optimality test in this algorithm motivates the discussion of optimality conditions for a variety of problems. The step procedure motivates the discussion of feasible directions (for constrained problems) and Newton's method and its variants (for nonlinear problems).

In general, there is an attempt to develop the material from a small number of basic concepts, emphasizing the interrelationships among the many topics. Our hope is that, by emphasizing a few fundamental principles, it will be easier to understand and assimilate the vast panorama of linear and nonlinear optimization.

We have attempted to make accessible a number of topics that are not often found in textbooks. Within linear programming, we have emphasized the importance of sparse matrices on the design of algorithms, described computational techniques used in sophisticated software packages, and derived the primal-dual interior-point method together with the predictor-corrector technique. Within nonlinear optimization, we have included discussions of truncated-Newton methods for large problems, convergence theory for trust-region methods, filter methods, and techniques for alleviating the ill-conditioning in barrier methods. We hope that the book serves as a useful introduction to research papers in these areas.

The book was designed for use in courses and course sequences that discuss both linear and nonlinear optimization. We have used consistent approaches when discussing the two topics, often using the same terminology and notation in order to emphasize the similarities between the two topics. However, it can also be used in traditional (and separate) courses in Linear Programming and Nonlinear Optimization—in fact, that is the way we use it in the courses that we teach. At the end of this preface are chapter descriptions and course outlines indicating these possibilities.

We have also used the book for more advanced courses. The later chapters (and the later sections within chapters) contain a great deal of material that would be difficult to cover in an introductory course. The Notes at the ends of many sections contain pointers to research papers and other references, and it would be straightforward to use such materials to supplement the book.

The book is divided into four parts plus appendices. Part I (Basics) contains material that might be used in a number of different topics. It is not intended that all of this material be presented in the classroom. Some of it might be irrelevant (as the sample course outlines illustrate). In other cases, material might be familiar to the students from other courses, or simple enough to be assigned as a reading exercise. The material in Part I could also be taught in stages, as it is needed. In a course on Nonlinear Optimization, for example, Chapter 4 (Representation of Linear Constraints) could be delayed until after Part III (Unconstrained Optimization). Our intention in designing Part I was to make the book as flexible as possible, and instructors should feel free to exploit this flexibility.

Part II (Linear Programming) and Part III (Unconstrained Optimization) are independent of each other. Either one could be taught or read before the other. In addition, it is not necessary to cover Part II before going on to Part IV (Nonlinear Optimization), although the material in Part IV will benefit from an understanding of Linear Programming. The material in the appendices may already be familiar. If not, it could either be presented in class or left for students to read independently.

Many sections in the book can be omitted without interrupting the flow of the discussions (detailed information on this is given below). Proofs of theorems and lemmas can similarly be omitted. Roughly speaking, it is possible to skip later sections within a chapter and later chapters within a part and move on to later chapters in the book. The book was organized in this way so that it would be accessible to a wider audience, as well as to increase its flexibility.

Many of the exercises are computational. In some cases, pencil-and-paper techniques would suffice, but the use of a computer is recommended. We have not specified how the computer might be used, and we leave this up to the instructor. In courses with an emphasis on modeling, a specialized linear or nonlinear optimization package might be appropriate. In other courses, the students might be asked to program algorithms themselves. We leave these decisions up to the instructor. Some information about software packages can be found in Appendix C. In addition, some exercises depend on auxiliary data sets that can be found on the web site for the book:

http://www.siam.org/books/ot108

In our own classes, we use the MATLAB® software package for class demonstrations and homework assignments. It allows us to demonstrate a great many techniques easily, and it allows students to program individual algorithms without much difficulty. It also includes (in its toolboxes) prepared algorithms for many of the optimization problems that we discuss.

We have gone to considerable effort to ensure the accuracy of the material in this book. Even so, we expect that some errors remain. For this reason, we have set up an online page for errata. It can be obtained at the book Web site.

**Using This Book**

This book is designed to be flexible. It can be read and taught in many different ways.

The material in the appendices can be taught as needed, or left to the students to read independently. Also, all formally identified proofs can be omitted.

Part II (Linear Programming) and Part III (Unconstrained Optimization) are independent of each other. Part II does not assume any knowledge of Calculus. Part IV (Nonlinear Optimization) does not assume that Part II has been read (with the exception of Section 14.4.1).

The only "essential" chapters in Part II are Chapters 4 (Geometry of Linear Programming), 5 (The Simplex Method), and 6 (Duality). The only "essential" chapter in Part III is Chapter 11 (Basics of Unconstrained Optimization). The other chapters can be skipped.

We now describe the chapters individually, pointing out various ways they can be used. The sample course outlines that follow indicate how chapters might be selected to construct individual courses (based on a 15-week semester).[1]

Part I: Basics

- *Chapter* 1: *Optimization Models.* This chapter is self-contained and describes a variety of optimization models. Sections 1.3–1.5 are independent of one another. Section 1.6 includes more realistic models and assumes that the reader is familiar with the basic models described in the earlier sections. The subsections of Section 1.6 are independent of one another.
- *Chapter* 2: *Fundamentals of Optimization.* For Part II, only Sections 2.1–2.4 are needed (and Section 2.3.1 can be omitted). For Parts III and IV the whole chapter is relevant.
- *Chapter* 3: *Representation of Linear Constraints.* Sections 3.3.2–3.3.4 can be omitted (although Section 3.3.2 is needed for Part IV). This chapter is only relevant to Parts II and IV; it is not needed for Part III.

Part II: Linear Programming

- *Chapter* 4: *Geometry of Linear Programming.* All sections of this chapter are needed in Part II.
- *Chapter* 5: *The Simplex Method.* Sections 5.1 and 5.2 are the most important. How the rest of the chapter is used depends on the goals of the instructor, in particular with regard to tableaus. In a number of examples, we use the full simplex tableau to display data for linear programs. Thus, it is necessary to be able to read these tableaus to extract information. This is the only use we make of the tableaus elsewhere in the book. It is not necessary to be able to manipulate these tableaus.

[1] Throughout the book, the number of a section or subsection begins with the chapter number. That is, Section 10.3 refers to the third section in Chapter 10, and Section 16.7.2 refers to the second subsection in the seventh section of Chapter 16. Also, a reference to Appendix A.9 refers to the ninth section of Appendix A. A similar system is used for tables, examples, theorems, etc.; Figure 8.10 refers to the tenth figure in Chapter 8, for example. For exercises, however, the chapter number is omitted, e.g., Exercise 4.7 is the seventh exercise in Section 4 of the current chapter (unless another chapter is specified).

- *Chapter* 6: *Duality and Sensitivity.* Sections 6.1 and 6.2 are the most important. The remaining sections can be skipped, if desired. If taught, we recommend that Sections 6.3–6.5 be taught in order, although Section 6.3 is only used in a minor way in the remaining two sections. It would be possible to stop after any section. Note: The remaining chapters in Part II are independent of each other.
- *Chapter* 7: *Enhancements of the Simplex Method.* The sections in this chapter are independent of each other. The instructor is free to pick and choose material, with one partial exception: the discussion of the decomposition principle is easier to understand if column generation has already been read.
- *Chapter* 8: *Network Problems.* In this chapter, the sections must be taught in order. It would be possible to stop after any section.
- *Chapter* 9: *Computational Complexity of Linear Programming.* The first two sections contain basic material used in Sections 9.3–9.5. Ideally, the remaining sections should be taught in order, although Sections 9.4 and 9.5 are independent of each other. Even if some topics are not of interest, at least the introductory paragraphs of each section should be read. (Section 9.5 requires some knowledge of statistics.)
- *Chapter* 10: *Interior-Point Methods for Linear Programming.* Sections 10.1 and 10.2 are the most important. The later sections could be skipped but, if taught, Sections 10.4–10.6 should be taught in order. Section 10.4 reviews some fundamental concepts from nonlinear optimization needed in Sections 10.5–10.6.

Part III: Unconstrained Optimization

- *Chapter* 11: *Basics of Unconstrained Optimization.* We recommend reading all of this chapter (with the exception of the proofs). If desired, either Section 11.5 or Section 11.6 could be omitted, but not both. Chapters 12 and 13 could be omitted. Chapter 13 makes more sense if taught after Chapter 12, but in fact, only Section 13.5 makes explicit use of the material in Chapter 12.
- *Chapter* 12: *Methods for Unconstrained Optimization.* Sections 12.1–12.3 are the most important. All the remaining sections and subsections can be taught independently of each other.
- *Chapter* 13: *Low-Storage Methods for Unconstrained Problems.* Once Sections 13.1 and 13.2 have been taught, the remaining sections are independent of each other.

Part IV: Nonlinear Optimization

- *Chapter* 14: *Optimality Conditions for Constrained Problems.* We recommend reading Sections 14.1–14.6. The rest of the chapter may be omitted. Within Section 14.8, Sections 14.8.3 and 14.8.5 can be taught without teaching the remaining subsections, although Section 14.8.5 depends on Section 14.8.3. (The discussion of nonlinear duality in Section 14.8 is only needed in Sections 16.6–16.8 of Chapter 16.)
- *Chapter* 15: *Feasible-Point Methods.* We recommend reading Sections 15.1–15.4 (although Section 15.4.1 could be omitted). These sections explain how to solve problems with linear constraints. Sections 15.5–15.7 discuss methods for problems

with nonlinear constraints. Sections 15.5 and 15.6 are independent of each other, but Section 15.7 depends on Section 15.5.

- *Chapter* 16: *Penalty and Barrier Methods.* We recommend reading Sections 16.1 and 16.2 (although Section 16.2.3 could be omitted). If more of the chapter is covered, then Section 16.3 should be read. Sections 16.4–16.8 are independent of each other. Sections 16.6–16.8 use Section 14.8.3 of Chapter 14.

### Changes in the Second Edition

The overall structure of the book has not changed in the new addition, and the major topic areas are the same. However, we have updated certain topics to reflect developments since the first edition appeared. We list the major changes here.

Chapter 1 has been expanded to include examples of more realistic optimization models (Section 1.6). The description of interior-point methods for linear programming has been thoroughly revised and restructured (Chapter 10). The discussion of derivative-free methods has been extensively revised to reflect advances in theory and algorithms (Section 12.5). In Part IV we have added material on filter methods (Section 15.7), nonlinear primal-dual methods (Section 16.7), and semidefinite programming (Section 16.8). In addition, numerous smaller changes have been made throughout the book.

Some material from the first edition has been omitted here. The most notable examples are the chapter on nonlinear least-squares data fitting, and the sections on interior-point methods for convex programming. These topics from the first edition are available at the book Web site (see above for the URL).

### Sample Course Outlines

We provide below some sample outlines for courses that might use this book. If a section is listed without mention of subsections, then it is assumed that all the subsections will be taught. If a subsection is specified, then the unmentioned subsections may be omitted.

### Proposed Course Outline: Linear Programming

#### I: Foundations

Chapter 1. Optimization Models
1. Introduction
3. Linear Equations
4. Linear Optimization
7. Optimization Applications
    1. Crew Scheduling and Fleet Scheduling

Chapter 2. Fundamentals of Optimization
1. Introduction
2. Feasibility and Optimality
3. Convexity
4. The General Optimization Algorithm

Chapter 3. Representation of Linear Constraints
1. Basic Concepts
2. Null and Range Spaces
3. Generating Null-Space Matrices
1. Variable Reduction Method

**II: Linear Programming**

Chapter 4. Geometry of Linear Programming
1. Introduction
2. Standard Form
3. Basic Solutions and Extreme Points
4. Representation of Solutions; Optimality

Chapter 5. The Simplex Method
1. Introduction
2. The Simplex Method
3. The Simplex Method (Details)
4. Getting Started—Artificial Variables
1. The Two-Phase Method
5. Degeneracy and Termination

Chapter 6. Duality and Sensitivity
1. The Dual Problem
2. Duality Theory
3. The Dual Simplex Method
4. Sensitivity

Chapter 7. Enhancements of the Simplex Method
1. Introduction
2. Problems with Upper Bounds
3. Column Generation
5. Representation of the Basis

Chapter 9. Computational Complexity of Linear Programming
1. Introduction
2. Computational Complexity
3. Worst-Case Behavior of the Simplex Method
4. The Ellipsoid Method
5. The Average-Case Behavior of the Simplex Method

Chapter 10. Interior-Point Methods for Linear Programming
1. Introduction
2. The Primal-Dual Interior-Point Method

**Proposed Course Outline: Nonlinear Optimization**

**I: Foundations**

Chapter 1. Optimization Models
1. Introduction

[2]Not all the applications need be taught.
[3]The material in Chapter 3 is not needed until Part IV.

**IV: Nonlinear Optimization**

Chapter 14. Optimality Conditions for Constrained Problems
- 1. Introduction
- 2. Optimality Conditions for Linear Equality Constraints
- 3. The Lagrange Multipliers and the Lagrangian Function
- 4. Optimality Conditions for Linear Inequality Constraints
- 5. Optimality Conditions for Nonlinear Constraints
- 6. Preview of Methods
- 8. Duality
  - 3. Wolfe Duality
  - 5. Duality in Support Vector Machines

Chapter 15. Feasible-Point Methods
- 1. Introduction
- 2. Linear Equality Constraints
- 3. Computing the Lagrange Multipliers
- 4. Linear Inequality Constraints
- 5. Sequential Quadratic Programming

Chapter 16. Penalty and Barrier Methods
- 1. Introduction
- 2. Classical Penalty and Barrier Methods

**Proposed Course Outline: Introduction to Optimization**

**I: Foundations**

Chapter 1. Optimization Models
- 1. Introduction
- 3. Linear Equations
- 4. Linear Optimization
- 5. Least-Squares Data Fitting
- 6. Nonlinear Optimization
- 7. Optimization Applications[4]

Chapter 2. Fundamentals of Optimization
- 1. Introduction
- 2. Feasibility and Optimality
- 3. Convexity
- 4. The General Optimization Algorithm
- 5. Rates of Convergence
- 6. Taylor Series
- 7. Newton's Method for Nonlinear Equations

Chapter 3. Representation of Linear Constraints
- 1. Basic Concepts
- 2. Null and Range Spaces

[4]Not all the applications need be taught.

**Acknowledgments**

We owe a great deal of thanks to the people who have assisted us in preparing this second edition of this book. In particular, we would like to thank the following individuals for reviewing various portions of the manuscript and providing helpful advice and guidance: Erling Andersen, Bob Bixby, Sanjay Mehrotra, Hans Mittelmann, Michael Overton, Virginia Torczon, and Bob Vanderbei. We are especially grateful to Sara Murphy at SIAM for guiding us through the preparation of the manuscript.

Special thanks also to Galina Spivak, whose design for the front cover skillfully conveys, in our minds, the spirit of the book.

We continue to be grateful to those individuals who contributed to the preparation of the first edition. These include: Kurt Anstreicher, John Anzalone, Todd Beltracchi, Dimitri Bertsekas, Bob Bixby, Paul Boggs, Dennis Bricker, Tony Chan, Jessie Cohen, Andrew Conn, Blaine Crowthers, John Dennis, Peter Foellbach, John Forrest, Bob Fourer, Christoph Luitpold Frommel, Saul Gass, David Gay, James Ho, Sharon Holland, Jeffrey Horn, Soonam Kahng, Przemyslaw Kowalik, Michael Lewis, Lorin Lund, Irvin Lustig, Maureen Mackin, Eric Munson, Arkadii Nemirovsky, Florian Potra, Michael Rothkopf, Michael Saunders, David Shanno, Eric Smith, Martin Smith, Pete Stewart, André Tits, Michael Todd, Virginia Torczon, Luis Vicente, Don Wagner, Bing Wang, and Tjalling Ypma.

While preparing the first edition, we received valuable support from the National Science Foundation. We also benefited from the facilities of the National Institute of Standards and Technology and Rice University.

*Igor Griva*
*Stephen G. Nash*
*Ariela Sofer*

# Part I

# Basics

# Chapter 1
# Optimization Models

## 1.1 Introduction

Optimization models attempt to express, in mathematical terms, the goal of solving a problem in the "best" way. That might mean running a business to maximize profit, minimize loss, maximize efficiency, or minimize risk. It might mean designing a bridge to minimize weight or maximize strength. It might mean selecting a flight plan for an aircraft to minimize time or fuel use. The desire to solve a problem in an optimal way is so common that optimization models arise in almost every area of application. They have even been used to explain the laws of nature, as in Fermat's derivation of the law of refraction for light.

Optimization models have been used for centuries, since their purpose is so appealing. In recent times they have come to be essential, as businesses become larger and more complicated, and as engineering designs become more ambitious. In many circumstances it is no longer possible, or economically feasible, for decisions to be made without the aid of such models. In a large, multinational corporation, for example, a minor percentage improvement in operations might lead to a multimillion dollar increase in profit, but achieving this improvement might require analyzing all divisions of the corporation, a gargantuan task. Likewise, it would be virtually impossible to design a new computer chip involving millions of transistors without the aid of such models.

Such large models, with all the complexity and subtlety that they can represent, would be of little value if they could not be solved. The last few decades have witnessed astonishing improvements in computer hardware and software, and these advances have made optimization models a practical tool in business, science, and engineering. It is now possible to solve problems with thousands or even millions of variables. The theory and algorithms that make this possible form a large portion of this book.

In the first part of this chapter we give some simple examples of optimization models. They are grouped in categories, where the divisions reflect the properties of the models as well as the differences in the techniques used to solve them. We include also a discussion of systems of linear equations, which are not normally considered to be optimization models. However, linear equations are often included as constraints in optimization models, and their solution is an important step in the solution of many optimization problems.

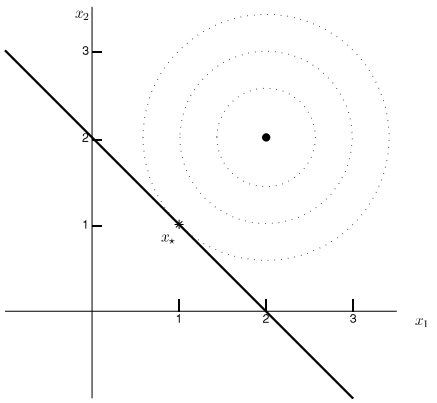


**Figure 1.1.** *Nonlinear optimization problem. The feasible set is the dark line.*

In the last section of this chapter we give some examples of applications of optimization. These examples reflect families of problems that are either in wide use, or—at the time of writing of this edition of the book—are subject of intense research. The examples reflect the tastes of the authors; by no means do they constitute a broad or representative sample of the myriad applications where optimization is in use today.

## 1.2 Optimization: An Informal Introduction

Consider the problem of finding the point on the line $x_1 + x_2 = 2$ that is closest to the point $(2, 2)^T$ (see Figure 1.1) . The problem can be written as

$$\begin{array}{ll} \text{minimize} & f(x) = (x_1 - 2)^2 + (x_2 - 2)^2 \\ \text{subject to} & x_1 + x_2 = 2. \end{array}$$

It is easy, of course, to see that the problem has an optimum at $x_\star = (1, 1)^T$.

This problem is an example of an optimization problem. Optimization problems typically minimize or maximize a function $f$ (called the *objective function*) in a set of points $S$ (called the *feasible set*). Commonly, the feasible set is defined by some constraints on the variables. In this example our objective function is the nonlinear function $f(x) = (x_1-2)^2+(x_2-2)^2$, and the feasible set $S$ is defined by a single linear constraint $x_1+x_2 = 2$. The feasible set could also be defined by multiple constraints. An example is the problem

$$\begin{array}{ll} \text{minimize} & f(x) = x_1 \\ \text{subject to} & x_1^2 \leq x_2 \\ & x_1^2 + x_2^2 \leq 2. \end{array}$$

The feasible set $S$ for this problem is shown in Figure 1.2; it is easy to see that the optimal point is $x_\star = (-1, 1)^T$. It is possible to have an *unconstrained optimization* problem where

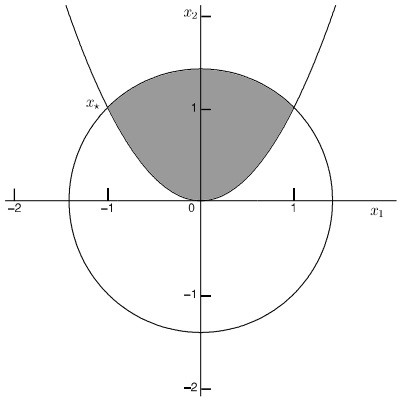


**Figure 1.2.** *Nonlinear optimization problem with inequality constraints.*

there are no constraints, as in the example

$$\text{minimize } f(x) = (e^{x_1} - 1)^2 + (x_2 - 1)^2.$$

The feasible set $S$ here is the entire two-dimensional space. The minimizer is $x_\star = (0, 1)^T$, since the function value is zero at this point and positive elsewhere.

We see from these examples that the feasible set can be defined by equality constraints or inequality constraints or no constraints at all. The functions defining the objective function and the constraints may be linear or nonlinear. The examples above are *nonlinear optimization* problems since at least some of the functions involved are nonlinear. If the objective function and the constraints are all linear, the problem is a *linear optimization problem* or *linear program*. An example is the problem

$$\begin{array}{ll} \text{maximize} & f(x) = 2x_1 + x_2 \\ \text{subject to} & x_1 + x_2 \leq 1 \\ & x_1 \geq 0,\ x_2 \geq 0. \end{array}$$

Figure 1.3 shows the feasible set. The optimal solution is clearly $x_\star = (1, 0)^T$.

Consider now the nonlinear optimization problem

$$\begin{array}{ll} \text{maximize} & f(x) = (x_1 + x_2)^2 \\ \text{subject to} & x_1 x_2 \geq 0 \\ & -2 \leq x_1 \leq 1 \\ & -2 \leq x_2 \leq 1. \end{array}$$

The feasible set is shown in Figure 1.4. The point $x_c = (1, 1)^T$ has an objective value of $f(x_c) = 4$, which is a higher objective value than any of its "nearby" feasible points. It is therefore called a *local optimizer*. In contrast the point $x_\star = (-2, -2)^T$ has an objective value $f(x_\star) = 16$ which is the best among all feasible points. It is called a *global optimizer*.

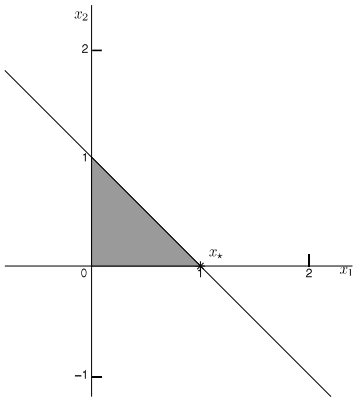


**Figure 1.3.** *Linear optimization problem. The feasible region is shaded.*

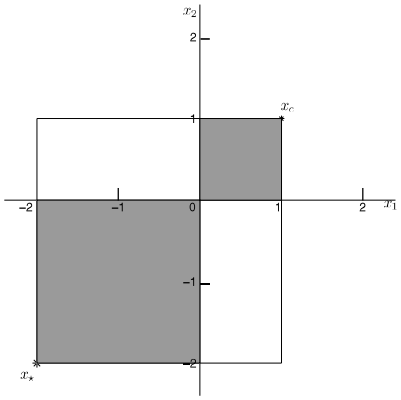


**Figure 1.4.** *Local and global solutions. The feasible region is shaded.*

The methods we consider in this book focus on finding local optima. We will usually assume that the problem functions and their first and second derivatives are continuous. We can then use derivative information at a given point to anticipate the behavior of the problem functions at "nearby" points and use this to determine whether the point is a local solution and if not, to find a better point. The derivative information cannot usually anticipate the behavior of the functions at points "farther away," and hence cannot determine whether a local solution is also the global solution. One exception is when the problem solved is a *convex optimization problem*, in which any local optimizer is also a global optimizer (see Section 2.3). Luckily, linear programs are convex so that for this important family of problems, local solutions are also global.

It may seem odd to give so much attention to finding local optimizers when they are not always guaranteed to be global optimizers. However, most global optimization algorithms seek the global optimum by finding local solutions to a sequence of subproblems generated by some methodical approximation to the original problem; the techniques described in the book are suitable for these subproblems. In addition, for some applications a local solution may be sufficient, or the user might be satisfied with an improvement on the objective value. Of course, some applications require finding a global solution. The drawback is that for a problem that is not convex (or not known to be convex), finding a global solution can require substantially more computational effort than finding a local solution.

Our book will also assume that the variables of the problems are continuous, that is, they can take a continuous range of real values. For this reason the problems we consider are also referred to as *continuous optimization problems*. Many variables such as length, volume, weight, and time are by nature continuous, and even though we cannot compute or measure them to infinite precision, it is plausible in the optimization to assume that they are continuous. On the other hand, variables such as the number of people to be hired, the number of flights to dispatch per day, or the number of new plants to be opened can assume only integer values. Problems where the variables can only take on integer values are called *discrete optimization problems* or, in the case where all problem functions are linear, *integer programming problems*. In a few applications it is sufficient to solve the problem ignoring the integrality restriction, and once a solution is obtained, to round off the variables to their nearest integer. Unfortunately rounding off of a solution does not guarantee that it is optimal, or even that it is feasible, so this approach is often inadequate.

While a discussion of discrete optimization is beyond the scope of this book, we will mention that such problems are much harder than their continuous counterparts for much the same reason global optimization is harder than local optimization. Since at a given point we only have information of the behavior of the function at "nearby points," there are no straightforward conditions that can determine whether a given feasible solution is optimal. Hence the solution process must rule out either explicitly or implicitly every other feasible solution. Thus the search for an integer solution requires the solution of a potentially large sequence of continuous optimization subproblems. Typically the first of these subproblems is a *relaxed problem*, in which the integrality requirement on each variable is relaxed (omitted) and replaced by a (continuous) constraint on the range of the variable. If, for example, a variable $x_j$ is restricted to be either 0, 1, or 2, the relaxed constraint would be $0 \leq x_j \leq 2$. Subsequent subproblems would typically include additional continuous constraints. The subproblems would be solved by continuous optimization methods such as those described in the book.

Continuous optimization is the basis for the solution of many applied problems, both discrete and continuous, convex or nonconvex. The examples in this chapter reflect just a small fraction of such applications.

## 1.3 Linear Equations

Systems of linear equations are central to almost all optimization algorithms and form a part of a great many optimization models. They are used in this section to represent a data-fitting example. A slight generalization of this example will lead to the important problem of least-squares data fitting. Linear equations are also used to represent constraints in a model.

Finally, solving systems of linear equations is an important step in the simplex method for linear programming and Newton's method for nonlinear optimization, and is a technique used to determine dual variables (Lagrange multipliers) in both settings. In this chapter we only give examples of linear equations. Techniques for their solution are discussed in Appendix A.

Our example is based on Figure 1.5. The points marked by • are assumed to lie on the graph of a quadratic function. These points, denoted by $(t_i, b_i)^T$, have the coordinates $(2, 1)^T$, $(3, 6)^T$, and $(5, 4)^T$. The quadratic function can be written as

$$b(t) = x_1 + x_2 t + x_3 t^2,$$

where $x_1$, $x_2$, and $x_3$ are three unknown parameters that determine the quadratic. The three data points define three equations of the form $b(t_i) = b_i$:

$$\begin{aligned} x_1 + x_2(2) + x_3(2)^2 &= 1 \\ x_1 + x_2(3) + x_3(3)^2 &= 6 \\ x_1 + x_2(5) + x_3(5)^2 &= 4 \end{aligned}$$

or

$$\begin{aligned} x_1 + 2x_2 + 4x_3 &= 1 \\ x_1 + 3x_2 + 9x_3 &= 6 \\ x_1 + 5x_2 + 25x_3 &= 4. \end{aligned}$$

The solution is $(x_1, x_2, x_3)^T = (-21, 15, -2)^T$, or

$$b(t) = -21 + 15t - 2t^2,$$

and is graphed in Figure 1.5.

This approach to data fitting has many applications. It is not unique to fitting data by a quadratic function. If the data were thought to have some sort of periodic component (perhaps a daily fluctuation), then a more appropriate model might be

$$b(t) = x_1 + x_2 t + x_3 \sin t,$$

and the system of equations would have the form

$$\begin{aligned} x_1 + x_2(2) + x_3(\sin 2) &= 1 \\ x_1 + x_2(3) + x_3(\sin 3) &= 6 \\ x_1 + x_2(5) + x_3(\sin 5) &= 4. \end{aligned}$$

Also, there is nothing special about having three data points and three terms in the model. If we wish to associate the data-fitting problem with a system of linear equations, then the number of data points and the number of model terms must be the same. However, through the use of least-squares models (see Section 1.5), it would be possible to have more data points than model terms. In fact, this is often the case. Least-squares techniques are also appropriate if there are measurement errors in the data (also a common occurrence).

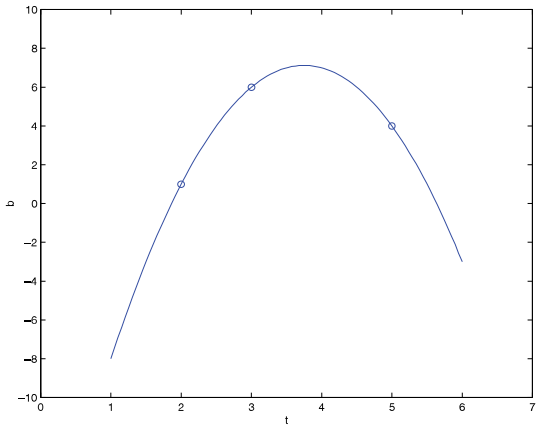


**Figure 1.5.** *Fitting a quadratic function to data.*

Let us return to the example of the quadratic model. We can write the system of equations in matrix form as

$$\begin{pmatrix} 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 5 & 25 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 6 \\ 4 \end{pmatrix},$$

or more generally,

$$\begin{pmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ 1 & t_3 & t_3^2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}.$$

If there were $n$ data points and the model were of the form

$$b(t) = x_1 + x_2 t + \cdots + x_n t^{n-1},$$

then the system would have the form

$$\begin{pmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ 1 & t_2 & \cdots & t_2^{n-1} \\ & & \vdots & \\ 1 & t_n & \cdots & t_n^{n-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}.$$

We will often denote such a system of linear equations as $Ax = b$.

For these examples the number of data points is equal to the number of variables. Equivalently the matrix $A$ has the same number of rows and columns. We refer to this as a "square" system because of the shape of the matrix $A$. It is also possible to consider problems with unequal numbers of data points and variables. Such examples, called "rectangular," are discussed in Section 1.5.

**Table 1.1.** *Cabinet data.*

| Cabinet | Wood | Labor | Revenue |
|---|---|---|---|
| Bookshelf | 10 | 2 | 100 |
| With Doors | 12 | 4 | 150 |
| With Drawers | 25 | 8 | 200 |
| Custom | 20 | 12 | 400 |

## 1.4 Linear Optimization

A linear optimization model (also knows as a "linear program") involves the optimization of a linear function subject to linear constraints on the variables. Although linear functions are simple functions, they arise frequently in economics, production planning, networks, scheduling, and other applications. We will consider several examples. Further examples are included in Section 1.7 and in Chapters 5–8. In particular, examples of network models are discussed in Section 8.2.

Suppose that a manufacturer of kitchen cabinets is trying to maximize the weekly revenue of a factory. Various orders have come in that the company could accept. They include bookcases with open shelves, cabinets with doors, cabinets with drawers, and custom-designed cabinets. Table 1.1 indicates the quantities of materials and labor required to assemble the four types of cabinets, as well as the revenue earned.

Suppose that 5000 units of wood and 1500 units of labor are available. Let $x_1, \ldots, x_4$ represent the number of cabinets of each type made ($x_1$ for bookshelves, $x_2$ for cabinets with doors, etc.). Then the corresponding linear programming model might be

$$\begin{aligned} \text{maximize} \quad & z = 100x_1 + 150x_2 + 200x_3 + 400x_4 \\ \text{subject to} \quad & 10x_1 + 12x_2 + 25x_3 + 20x_4 \leq 5000 \\ & 2x_1 + 4x_2 + 8x_3 + 12x_4 \leq 1500 \\ & x_1, x_2, x_3, x_4 \geq 0. \end{aligned}$$

This problem can easily be expanded from four products (bookshelves, cabinets with doors, cabinets without doors, etc.) to any number of products $n$, and from two resources (wood and labor) to any number of resources $m$. Denoting the unit profit from product $j$ by $c_j$, the amount available of resource $i$ by $b_i$, and the amount of resource $i$ used by a unit of product $j$ by $a_{ij}$, the problem can be written in the form

$$\begin{aligned} \text{maximize} \quad & z = \sum_{j=1}^{n} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{n} a_{ij} \leq b_i, \quad i = 1, \ldots, m \\ & x_j \geq 0, \qquad\quad j = 1, \ldots, n. \end{aligned}$$

The problem can be written in a more compact manner by introducing matrix-vector notation. Letting $x = (x_1, \ldots, x_n)^T$, $c = (c_1, \ldots, c_n)^T$, $b = (b_1, \ldots, b_m)^T$, and denoting the matrix

**Table 1.2.** *Work times (in minutes).*

| Worker | Information | Policy | Claim |
|---|---|---|---|
| 1 | 10 | 28 | 31 |
| 2 | 15 | 22 | 42 |
| 3 | 13 | 18 | 35 |
| 4 | 19 | 25 | 29 |
| 5 | 17 | 23 | 33 |

of coefficients $a_{ij}$ by $A$, the problem becomes

$$\begin{aligned} \text{maximize} \quad & z = c^T x \\ \text{subject to} \quad & Ax \leq b \\ & x \geq 0. \end{aligned}$$

This is a typical example of a linear program. Here a linear objective function is to be maximized subject to linear inequality constraints and nonnegativity constraints on the variables. In the general case, the objective of a linear program may be either maximized or minimized, the constraints may involve a combination of inequalities and equalities, and the variables may be either restricted in sign or unrestricted. Although these may appear as different forms, it is easy to convert from one form to another.

As another example, consider the assignment of jobs to workers. Suppose that an insurance office handles three types of work: requests for information, new policies, and claims. There are five workers. Based on a study of office operations, the average work times (in minutes) for the workers are known; see Table 1.2.

The company would like to minimize the overall elapsed time for handling a (long) sequence of tasks, by appropriately assigning a fraction of each type of task to each worker. Let $p_i$ be the fraction of information calls assigned to worker $i$, $q_i$ the fraction of new policy calls, and $r_i$ the fraction of claims; $t$ will represent the elapsed time. Then a linear programming model for this situation would be

$$\begin{aligned} \text{minimize} \quad & z = t \\ \text{subject to} \quad & p_1 + p_2 + p_3 + p_4 + p_5 = 1 \\ & q_1 + q_2 + q_3 + q_4 + q_5 = 1 \\ & r_1 + r_2 + r_3 + r_4 + r_5 = 1 \\ & 10p_1 + 28q_1 + 31r_1 \leq t \\ & 15p_2 + 22q_2 + 42r_2 \leq t \\ & 13p_3 + 18q_3 + 35r_3 \leq t \\ & 19p_4 + 25q_4 + 29r_4 \leq t \\ & 17p_5 + 23q_5 + 33r_5 \leq t \\ & p_i, q_i, r_i \geq 0, \ i = 1, \ldots, 5. \end{aligned}$$

The constraints in this model assure that $t$ is no less than the overall elapsed time. Since the objective is to minimize $t$, at the optimal solution $t$ will be equal to the elapsed time.

The problems we have introduced so far are small, involving only a handful of variables and constraints. Many real-life applications involve much larger problems, with possibly hundreds of thousands of variables and constraints. Section 1.7 discusses some of these applications.

## Exercise[5]

4.1. Consider the production scheduling problem of the perfume Polly named after a famous celebrity. The manufacturer of the perfume must plan production for the first four months of the year and anticipates a demand of 4000, 5000, 6000, and 4500 gallons in January, February, March, and April, respectively. At the beginning of the year the company has an inventory of 2000 gallons. The company is planning on issuing a new and improved perfume called Pollygone in May, so that all Polly produced must be sold by the end of April. Assume that the production cost for January and February is \$5 per gallon and this will rise to \$5.5 per gallon in March and April. The company can hold any amount produced in a certain month over to the next month at an inventory cost of \$1 per unit. Formulate a linear optimization model that will minimize the costs incurred in meeting the demand for Polly in the period January through April. Assume for simplicity that any amount produced in a given month may be used to fulfill demand for that month.

## 1.5 Least-Squares Data Fitting

Let us re-examine the quadratic model from Section 1.3:

$$b(t) = x_1 + x_2 t + x_3 t^2.$$

For the data points (2, 1), (3, 6), and (5, 4) we obtained the linear system

$$\begin{pmatrix} 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 5 & 25 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 6 \\ 4 \end{pmatrix}$$

with solution $x = (-21, 15, -2)^T$ so that

$$b(t) = -21 + 15t - 2t^2.$$

It is easy to check that the three data points satisfy this equation.

Suppose that the data points had been obtained from an experiment, with an observation made at times $t_1 = 2$, $t_2 = 3$, and $t_3 = 5$. If another observation were made at $t_4 = 7$, then (assuming that the quadratic model is correct) it should satisfy

$$b(7) = -21 + 15 \times 7 - 2 \times 7^2 = -14.$$

If the observed value at $t_4 = 7$ were not equal to $-14$, then the observation would not be consistent with the model.

[5] See Footnote 1 in the Preface for an explanation of the Exercise numbering within chapters.

It is common when collecting data to gather more data points than there are variables in the model. This is true in political polls where hundreds or thousands of people will be asked which candidate they plan to vote for (so that there is only one variable). It is also true in scientific experiments where repeated measurements will be made of a desired quantity. It is expected that each of the measurements will be in error, and that the observations will be used collectively in the hope of obtaining a better result than any individual measurement provides. (The collective result may only be better in the sense that the bound on its error will be smaller. Since the true value is often unknown, the actual errors cannot be measured.)

Since each of the measurements is considered to be in error, it is no longer sensible to ask that the model equation (in our case $b(t) = x_1 + x_2 t + x_3 t^2$) be solved exactly. Instead we will try to make components of the "residual vector"

$$r = b - Ax = \begin{pmatrix} b_1 - (x_1 + x_2 t_1 + x_3 t_1^2) \\ b_2 - (x_1 + x_2 t_2 + x_3 t_2^2) \\ \vdots \\ b_m - (x_1 + x_2 t_m + x_3 t_m^2) \end{pmatrix}$$

small in some sense.

The most commonly used approach is called "least squares" data fitting, where we try to minimize the sum of the squares of the components of $r$:

$$\underset{x}{\text{minimize}} \;\; r_1^2 + \cdots + r_m^2 = \sum_{i=1}^{m} [b_i - (x_1 + x_2 t_i + x_3 t_i^2)]^2.$$

Under appropriate assumptions about the errors in the observations, it can be shown that this is an optimal way of selecting the coefficients $x$.

If the fourth data point was $(7, -14)^T$, then the least-squares approach would give $x = (-21, 15, -2)^T$, since this choice of $x$ would make $r = 0$. In this case the graph of the model would pass through all four data points. However, if the fourth data point was $(7, -15)^T$, then the least-squares solution would be

$$x = \begin{pmatrix} -21.9422 \\ 15.6193 \\ -2.0892 \end{pmatrix}.$$

The corresponding residual vector would be

$$r = b - Ax = \begin{pmatrix} 0.0603 \\ -0.1131 \\ 0.0754 \\ -0.0226 \end{pmatrix}.$$

None of the residuals is zero, and so the graph of the model does not pass through any of the data points. This is typical in least-squares models.

If the residuals can be written as $r = b - Ax$, then the model is "linear." This name is used because each of the coefficients $x_j$ occurs linearly in the model. It does not mean that the model terms are linear in $t$. In fact, the model above has a quadratic term $x_3 t^2$. Other

examples of linear models would be

$$b(t) = x_1 + x_2 \sin t + x_3 \sin 2t + \cdots + x_{k+1} \sin kt$$
$$b(t) = x_1 + \frac{x_2}{1 + t^2}.$$

"Nonlinear" models are also possible. Some examples are

$$b(t) = x_1 + x_2 e^{x_3 t} + x_4 e^{x_5 t}$$
$$b(t) = x_1 + \frac{x_2}{1 + x_3 t^2}.$$

In these models there are nonlinear relationships among the coefficients $x_j$. A nonlinear least-squares model can be written in the form

$$\text{minimize } f(x) = \sum_{i=1}^{m} r_i(x)^2,$$

where $r_i(x)$ represents the residual at $t_i$. For example,

$$r_i(x) \equiv b_i - (x_1 + x_2 e^{x_3 t_i} + x_4 e^{x_5 t_i})$$

for the first nonlinear model above. We can also write this as

$$f(x) = r(x)^T r(x).$$

If the model is linear, then $r(x) = b - Ax$ and $f(x)$ can be shown to be a quadratic function. See the Exercises.

Nonlinear least squares models are examples of unconstrained minimization problems, that is, they correspond to the minimization of a nonlinear function without constraints on the variables. In fact, they are one of the most commonly encountered unconstrained minimization problems.

## Exercises

5.1. Prove that for the linear least-squares problem with $r(x) = b - Ax$, the objective $f(x) = r(x)^T r(x)$ is a quadratic function.

## 1.6 Nonlinear Optimization

A nonlinear optimization model (also referred to as a "nonlinear program") consists of the optimization of a function subject to constraints, where any of the functions may be nonlinear. This is the most general type of model that we will consider in this book. It includes all the other types of models as special cases.

Nonlinear optimization models arise often in science and engineering. For example, the volume of a sphere is a nonlinear function of its radius, the energy dissipated in an electric

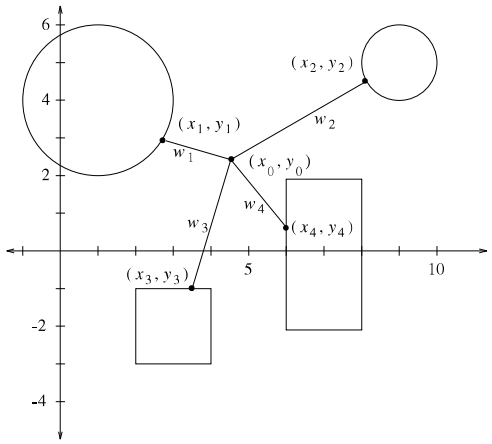


**Figure 1.6.** *Electrical connections.*

circuit is a nonlinear function of the resistances, the size of an animal population is a nonlinear function of the birth and death rates, etc. We will develop two specific examples here.

Suppose that four buildings are to be connected by electrical wires. The positions of the buildings are illustrated in Figure 1.6. The first two buildings are circular: one at $(1, 4)^T$ with radius 2, the second at $(9, 5)^T$ with radius 1. The third building is square with sides of length 2 centered at $(3, -2)^T$. The fourth building is rectangular with height 4 and width 2 centered at $(7, 0)^T$. The electrical wires will be joined at some central point $(x_0, y_0)^T$ and will connect to building $i$ at position $(x_i, y_i)^T$. The objective is to minimize the amount of wire used. Let $w_i$ be the length of the wire connecting building $i$ to $(x_0, y_0)^T$. A model for this problem is

$$\begin{aligned}
\text{minimize} \quad & z = w_1 + w_2 + w_3 + w_4 \\
\text{subject to} \quad & w_i = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2}, \ i = 1, 2, 3, 4, \\
& (x_1 - 1)^2 + (y_1 - 4)^2 \leq 4 \\
& (x_2 - 9)^2 + (y_2 - 5)^2 \leq 1 \\
& 2 \leq x_3 \leq 4 \\
& -3 \leq y_3 \leq -1 \\
& 6 \leq x_4 \leq 8 \\
& -2 \leq y_4 \leq 2.
\end{aligned}$$

We assume here for simplicity that the wires can be routed through the buildings (if necessary) at no additional cost.

The constraints in nonlinear optimization problems are often written so that the right-hand sides are equal to zero. For the above model this would correspond to using constraints of the form

$$w_i - \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} = 0, \ i = 1, 2, 3, 4,$$

and so forth. This is just a cosmetic change to the model.

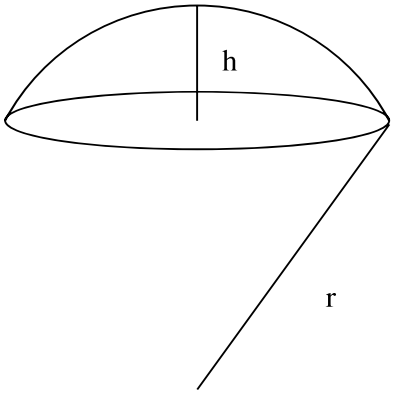


**Figure 1.7.** *Archimedes' problem.*

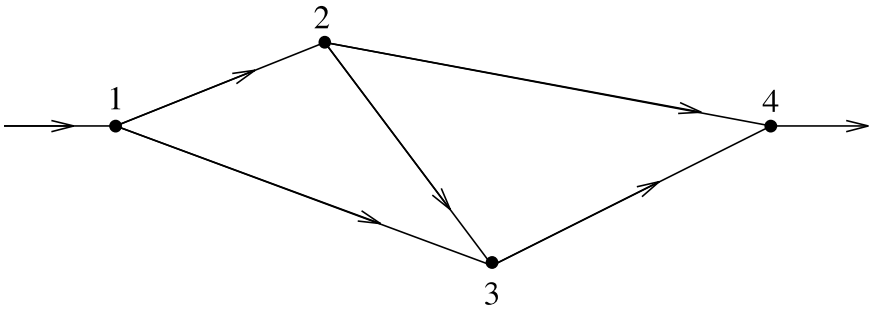


**Figure 1.8.** *Traffic network.*

As a second example we consider a problem posed by Archimedes. Figure 1.7 illustrates a portion of a sphere with radius $r$, where the height of the spherical segment is $h$. The problem is to choose $r$ and $h$ so as to maximize the volume of the segment, but where the surface area $A$ of the segment is fixed. The model is

$$\begin{array}{ll} \text{maximize} & v(r,h) = \pi h^2(r - \frac{h}{3}) \\ \text{subject to} & 2\pi rh = A. \end{array}$$

Archimedes was able to prove that the solution was a hemisphere (i.e., $h = r$).

As another illustration of how nonlinear models can arise, consider the network in Figure 1.8. This represents a set of road intersections, and the arrows indicate the direction of traffic. If few cars are on the roads, the travel times between intersections can be considered as constants, but if the traffic is heavy, the travel times can increase dramatically.

Let us focus on the travel time between a pair of intersections $i$ and $j$. Let $t_{i,j}$ be the (constant) travel time when the traffic is light, let $x_{i,j}$ be the number of cars entering the road per hour, let $c_{i,j}$ be the capacity of the road, that is, the maximum number of cars entering per hour, and let $\alpha_{i,j}$ be a constant reflecting the rate at which travel time increases as the traffic get heavier. (The constant $\alpha_{i,j}$ might be selected using data collected about the road system.)

Then the travel time between intersections $i$ and $j$ could be modeled by

$$T_{i,j}(x_{i,j}) = t_{i,j} + \alpha_{i,j}\frac{x_{i,j}}{1 - x_{i,j}/c_{i,j}}.$$

If there is no traffic on the road ($x_{i,j} = 0$), then the travel time is $t_{i,j}$. If $x_{i,j}$ approaches the capacity of the road $c_{i,j}$, then the travel time tends to $+\infty$. $T_{i,j}$ is a nonlinear function of $x_{i,j}$.

Suppose we wished to minimize the total travel time through the network for a volume of $X$ cars per hour. Then our model would be

$$\text{minimize } f(x) = \sum x_{i,j}T_{i,j}(x_{i,j})$$

subject to the constraints

$$\begin{aligned} x_{1,2} + x_{1,3} &= X \\ x_{2,3} + x_{2,4} - x_{1,2} &= 0 \\ x_{3,4} - x_{1,3} - x_{2,3} &= 0 \\ x_{2,4} + x_{3,4} &= X \\ 0 \le x_{i,j} &\le c_{ij}. \end{aligned}$$

The equations ensure that all cars entering an intersection also leave an intersection. The objective sums up the travel times for all the cars.

A potential snag with this formulation is that if the traffic volume reaches capacity on any arc ($x_{i,j} = c_{i,j}$), the objective function becomes undefined, which will cause optimization software to fail. A number of measures could be invoked to prevent this situation. One alternative is to slightly lower the upper bounds on the variables, so that $x_{i,j} \le c_{i,j} - \epsilon$, where $\epsilon$ is a small positive number. Alternatively we could increase each denominator in the objective by a small positive amount $\epsilon$, thus forcing the denominator to have a value of at least $\epsilon$ and thereby avoiding division by zero.

Our last example is the problem of finding the minimum distance from a point $r$ to the set $\{x : a^T x = b\}$. In two dimensions the points in the set $S$ define a line, and in three dimensions they define a plane; in the more general case, the set is called a *hyperplane*. The least-distance problem can be written as

$$\begin{array}{ll} \text{minimize} & f(x) = \frac{1}{2}(x - r)^T(x - r) \\ \text{subject to} & a^T x = b. \end{array}$$

(The coefficient of one half in the objective is included for convenience; it allows for simpler formulas when analyzing the problem.) Unlike most nonlinear problems this one has a closed-form solution. It is given by

$$x = r + \frac{b - a^T r}{a^T a}a.$$

(See the Exercises for Section 14.2.)

The minimum distance problem is an example of a *quadratic program*. In general, a quadratic program involves the minimization of a quadratic function subject to linear constraints. An example is the problem

$$\begin{array}{ll} \text{minimize} & f(x) = \frac{1}{2}x^T Q x \\ \text{subject to} & Ax \geq b. \end{array}$$

Quadratic programs for which the matrix $Q$ is positive definite are relatively easy to solve, compared to other nonlinear problems.

## 1.7 Optimization Applications

In this section we present a number of applications that are of current interest to practitioners or researchers. The models we present are but a few of the numerous applications where optimization is making a significant impact.

We start by presenting two problems arising in the optimization of airline operations—the crew scheduling and fleet scheduling problems. Both problems are large linear programs with the added restriction that the variables must take on integer values.

Next we discuss an approach for pattern classification known as support vector machines. Given a set of points that all belong to one of two classes, the idea is to estimate a function that will automatically classify to which of the two classes a new point belongs. In particular we discuss the case where the classifying function is linear. The resulting problem is a quadratic program. This topic is developed further in Chapter 14. Also in this section we discuss a portfolio optimization problem that attempts to balance between the competing goals of maximizing expected returns and minimizing risk in investment planning. This too is a quadratic program.

Next we will discuss two optimization problems arising from medical applications. One problem arises from planning for treatment of cancer by radiation, where the conflicting goals of providing sufficient radiation to the tumor and limiting the dosage to nearby vital organs give rise to a plethora of models which cover the spectrum from linear through quadratic to nonlinear. The other problem arises from positron emission tomography (PET) image reconstruction, where a model of the image that best fits the scan data gives rise to a linearly constrained nonlinear problem. In both applications the optimization problems can be very large and challenging to solve.

Finally we use optimization to find the shape of a hanging cable with minimum potential energy. We present several models of the problem and emphasize the importance of certain modeling issues.

### 1.7.1 Crew Scheduling and Fleet Scheduling

Consider an airline that operates 2000 flights per day serving 100 cities worldwide, with 400 aircraft of 10 different types, each requiring a flight crew. The airline must design a flight schedule that meets the passenger demand, the maintenance requirements on aircraft, and all other safety regulations and labor contract rules, while trying to be cost effective in order to maximize profit.

This planning problem is extremely complex. For this reason many airlines used a phased planning cycle that breaks the problem into smaller steps. While more manageable, the individual steps themselves can also be complex.

Arguably the most challenging of these is the *crew scheduling problem*, that assigns crews (pilots and flight attendants) to flights. Economically it is a significant problem, since the cost of crews is second only to the cost of fuel in an airline's operating expenses. Saving even 1% of this cost can save the airline hundreds of millions of dollars annually. Computationally it is a difficult problem since it involves a linear model, which is not only very large, but also involves integer variables, which necessitates multiple solutions of linear programs.

In planning the crew activities, the flight schedule is subdivided into "legs," representing a nonstop flight from one city to another. If a plane flew from, say, New York via Chicago to Los Angeles, this would be considered as two legs. A large airline would typically have hundreds of flight legs per day. The planning period might be a day, a week, or a month.

The crews themselves are certified for particular aircraft, and this restricts how personnel can be assigned to legs. In addition, there are union rules and federal laws that constrain the crew assignments.

To set up the model, the airline first specifies a set of possible crew assignments. One of these assignments might correspond to sending a crew from New York (their home city) to a sequence of cities and then back to New York. Each such round trip is called a "pairing." The number of pairings grows exponentially with the number of legs, and for a large airline, the number of pairings may easily run into the billions, even for the shorter planning period of one week.

The variables in the model are $\{ x_j \}$, where $x_j$ is 1 if a particular pairing is selected as part of the total schedule, and 0 otherwise. Let the total number of pairings be $N$. The majority of the constraints correspond to the requirement that each leg in the planning period be covered by exactly one pairing. For the $i$th leg, the constraint has the form

$$\sum_{j=1}^{N} a_{i,j} x_j = 1,$$

where the constant $a_{i,j} = 1$ if a particular pairing includes leg $i$, and zero otherwise. There is one such constraint for every leg in the schedule.

The columns of the matrix $A$ correspond to the pairings, and each pairing must represent a round trip that is technically and legally feasible. For example, if a crew flies from New York to Chicago, it cannot then immediately fly out of Denver. The pairing makes sense if it makes sense chronologically, includes minimum rests between flights, satisfies regulations on maximum flying time, and so forth. This places many restrictions on how the pairings are generated, and hence on the coefficients $a_{i,j}$. The resulting columns of $A$ are typically very sparse, with many zeros, and just a few ones, corresponding to the legs of the roundtrip.

The cost $c_j$ of a pairing is a function of the duration of the pairing, the number of flight hours, and "penalties" that may be associated with the pairing. For example, extra wages and expenses must be paid if the crew spends a night away from its home city, or it may be necessary to transport a crew from one city to another for them to complete the pairing.

The basic model has the form

$$\begin{aligned} \text{minimize} \quad & z = c^T x \\ \text{subject to} \quad & \sum_{j=1}^{N} a_{i,j} x_j = 1 \\ & x_j = 0 \text{ or } 1. \end{aligned}$$

The problem is a linear program with the additional requirement that the variables take on integer values (here—zero and one), hence it is an integer programming problem. As mentioned in Section 1.2, such problems are most commonly solved by solving a sequence of linear programs, where the integrality restrictions are relaxed and replaced by a (continuous) constraint on the range of the variable. The range should ideally be as tight as possible, yet should not exclude the optimal solution. For a zero-one problem the relaxed constraints for the first subproblem would typically be $0 \leq x_j \leq 1$ for all $j$. Subsequent problems are variants of the relaxed problem, usually with additional constraints or an adjusted objective function.

Crew scheduling problems can be very large. A major effort is required just to generate the possible pairings. Commonly, only a partial model is generated, corresponding to a subset of the possible pairings. Even so, problems with millions of variables are typical.

Linear programs of this size (even ignoring the integrality restriction) are difficult to solve. They demand all the resources of the most sophisticated software. The special structure of the matrix $A$ (and in particular its sparsity—the large number of zero entries) and the latest algorithmic techniques must be used. Many of these techniques are discussed in Part II.

The crew scheduling problem is typically the last step in an airline's schedule planning. The first step begins about several months prior to the actual service when the airline selects the optimal set of flight legs to be included in its schedule. The flight schedule lists the schedule of flight legs by departure time, destination, and arrival time.

The next step is fleet assignment, which determines which type of aircraft will fly each leg of the schedule. Airline fleets are made up of many different types of aircraft, which differ in capacity and in operational characteristics such as speed, fuel burn rates, landing weights, range, and maintenance costs. Allocating an aircraft that is too small will result in loss of revenue from passengers turned away, while allocating an aircraft that is too big will result in too many empty seats to cover the high expenses. The airline's problem is to determine the best aircraft to use for each flight leg such that capacity is matched to demand while minimizing the operating cost.

This problem is frequently represented as a *time-line network*. The network includes a line called a "time-line" for each airport, with nodes positioned along the line in chronological order at each arrival and departure time. Each flight is represented by an arc in the network. Thus for example a flight leaving Washington Dulles (IAD) at 6:00 am (Eastern Standard Time) and arriving at Denver (DEN) at 10:00 am (Eastern Standard Time) would be represented by an arc connecting the 6:00 am node on the IAD time-line to the 10:00 am node on the DEN time-line. (In practice, the arrival time is adjusted to account for the time it takes to prepare the aircraft for the next flight, but we will ignore that here.) In addition to the flight arcs we create an arc from each node on a time line to the consecutive node on the

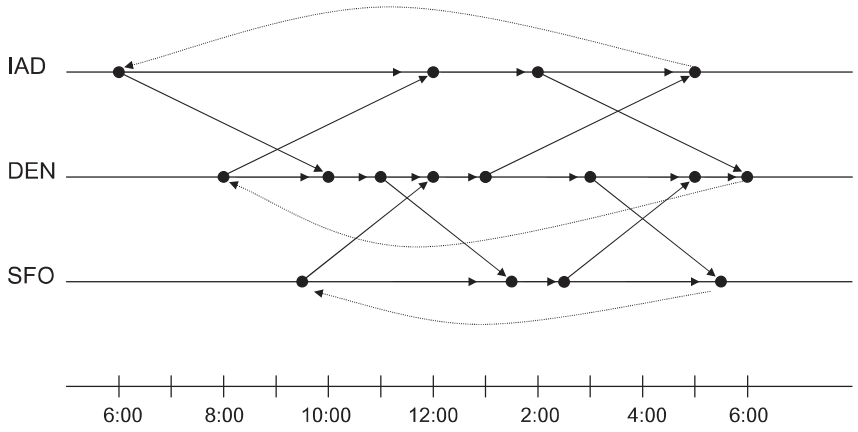


**Figure 1.9.** *Time-line network.*

time line, and (assuming the schedule is repeated daily) an arc from the last node returning to the first node. The flow on these arcs represents aircraft on the ground that are waiting for their flight.

Figure 1.9 illustrates a time-line network for an airline that has two flights a day each from IAD to DEN, DEN to IAD, DEN to SFO (San Francisco), and SFO to DEN.

Define now $x_{ij}$ to be the number of aircraft of type $i$ on arc $j$. Any feasible fleet assignment solution must satisfy the following constraints: (i) Covering constraints: each flight leg must be covered by exactly one aircraft; (ii) Flow-balance constraints: for each node of the network the total number of aircraft of type $i$ entering the node must equal the total number of aircraft of type $i$ exiting the node; (iii) Fleet size constraints: the number of aircraft used of each type must not exceed the number of aircraft available. The objective is to minimize the total cost of the assignment. The problem is by nature integer, but it is generally solved by a series of linear programs where the integrality restrictions are relaxed.

Once the fleet is assigned, the individual aircraft of the fleet must be assigned to their flights. This is known as the aircraft routing problem. The planning must take into account the required maintenance for each aircraft. To meet safety regulations, an airline might typically maintain aircraft every 40–45 hours of flying with the maximum time between checks restricted to three to four calendar days. The problem is to determine the most cost effective assignment of aircraft of a single fleet to the scheduled flights, so that all flight legs are covered and aircraft maintenance requirements are satisfied.

The last step of the planning cycle is the task of crew scheduling. Breaking down the full planning cycle into steps helps make the planning more manageable, but ultimately it leads to suboptimal schedules (see Exercise 7.2). Researchers are therefore investigating methods that combine two or more of the planning phases together for more profitable schedules.

## Exercises

7.1. Formulate the fleet scheduling problem corresponding to Figure 1.9.

7.2. Consider an airline that has scheduled the flight legs for the next month. It has done so by breaking down the planning cycle into a sequence of steps: first determine the optimal fleet for this schedule; next route the aircraft within the fleet to the flight legs; and finally assign crews for each of the flight legs. Discuss why this makes the planning more manageable but likely leads to suboptimal schedules.

### 1.7.2 Support Vector Machines

Suppose that you have a set of data points that you have classified in one of two ways: either they have a certain stated property or they do not. These data points might represent the subject titles of email messages, which are classified as either being legitimate email or spam; or they may represent medical data such as age, sex, weight, blood pressure, cholesterol levels, and genetic traits of patients that have been classified either as high risk or as low risk for a heart attack; or they may represent some features of handwritten digits such as ratio of height to width, curvature, that have been classified either as (say) zero or not zero. Suppose now that you obtain a new data point. Your goal is to determine whether this new point does or does not have the stated property. The set of techniques for doing this is broadly referred to as *pattern classification*. The main idea is to identify some rule based on the existing data (referred to as the *training data*) that characterizes the set of points that have the property, which can then be used to determine whether a new point has the property.

In its simplest form classification uses linear functions to provide the characterization. Suppose we have a set of $m$ training data $x_i \in \Re^n$ with classification $y_i$, where either $y_i = 1$ or $y_i = -1$. A two-dimensional example is shown in the left-hand side of Figure 1.10, where the two classes of points are designated by circles of different shades. Suppose it is possible to find some hyperplane $w^T x + b = 0$ which separates the positive points from the negative. Ideally we would like to have a sharp separation of the positive points from the negative. Thus we will require

$$\begin{aligned} w^T x_i + b &\geq +1 \quad \text{for } y_i = +1, \\ w^T x_i + b &\leq -1 \quad \text{for } y_i = -1. \end{aligned}$$

There is nothing special about the separation coefficients $\pm$ on the right-hand side of the above inequalities. The coefficients $w$ and $b$ of the hyperplane can always be scaled so that the separation will be $\pm$ 1.

To obtain the best results we would like the hyperplanes separating the positive points from the negative to be as far apart as possible. From basic geometric principles it can be shown that the distance between the two hyperplanes (that is, the *separation margin*) is $2/\|w\|$. Thus among all separating hyperplanes we should seek the one that maximizes this margin. This is equivalent to minimizing $w^T w$. The resulting problem is to determine the coefficients $w$ and $b$ that solve

$$\begin{aligned} \text{minimize} \quad & f(w,b) = \tfrac{1}{2} w^T w \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1, \quad i = 1, \ldots, m. \end{aligned}$$

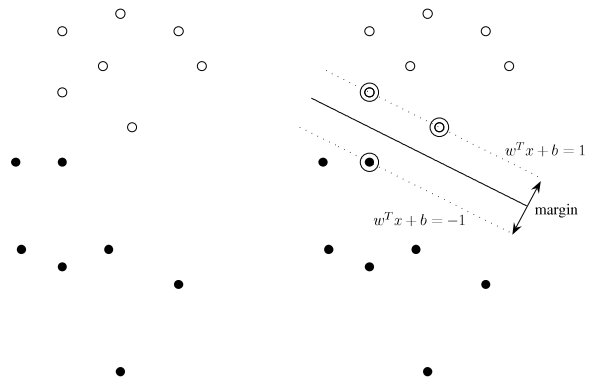


**Figure 1.10.** *Linear separating hyperplane for the separable case.*

The coefficient $\frac{1}{2}$ in the objective is included for convenience; it results in simpler formulas when analyzing the problem.

The right-hand side of Figure 1.10 shows the solution of our two-dimensional example. The training points that lie on the boundary of either of the hyperplanes are called the *support vectors*; they are highlighted by larger circles. Removal of these points from our training set would change the coefficients of the hyperplanes. Removal of the other training points would leave the coefficients unchanged. The method is called a "support vector machine" because support vectors are used for classifying data as part of a machine (computerized) learning process.

Once the coefficients $w$ and $b$ of the separating hyperplane are found from the training data, we can use the value of the function $f(x) = w^T x + b$ (our "learning machine") to predict whether a new point $\bar{x}$ has the property of interest or not, depending on the sign of $f(\bar{x})$.

So far we have assumed that the data set was separable, that is, a hyperplane separating the positive points from the negative points exists. For the case where the data set is not separable, we can refine the approach to the separable case. We will now allow the points to violate the equations of the separating hyperplane, but we will impose a penalty for the violation. Letting the nonnegative variable $\xi_i$ denote the amount by which the point $x_i$ violates the constraint at the margin, we now require

$$\begin{aligned} w^T x_i + b &\geq +1 - \xi_i \quad \text{for } y_i = +1 \\ w^T x_i + b &\leq -1 + \xi_i \quad \text{for } y_i = -1. \end{aligned}$$

A common way to impose the penalty is to add to the objective a term proportional to the sum of the violations. The added penalty term takes the form $C \sum_{i=1}^{m} \xi$ to the objective, where the larger the value of the parameter $C$, the larger the penalty for violating

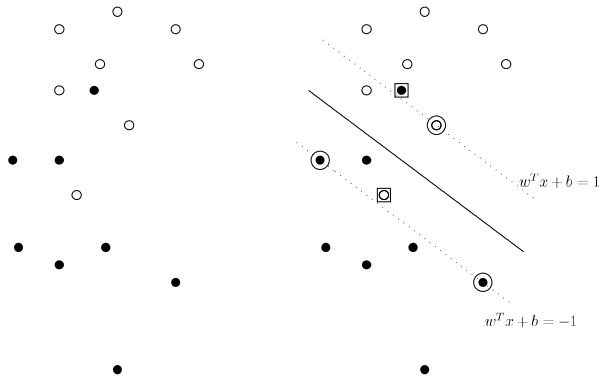


**Figure 1.11.** *Linear separating hyperplane for the nonseparable case.*

the separation. Our problem is now to find $w$, $b$, and $\xi$ that solve

$$\begin{array}{ll} \text{minimize} & f(w,b,\xi) = \frac{1}{2}w^T w + C\sum_{i=1}^m \xi_i \\ \text{subject to} & y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1,\ldots,m, \\ & \xi_i \geq 0. \end{array}$$

Figure 1.11 shows an example of the nonseparable case and the resulting separating hyperplane. We see in this example that two of the points (indicated in the figure by the extra squares) are misclassified, since they lie on the incorrect side of the hyperplane $w^T x + b = 0$.

In later chapters of this book we will see that many problems have a companion problem called the dual problem, that there are important relations between a problem and its dual, and that these relations sometimes lead to insights for solving the problem. In Section 14.8 we will discuss the dual of the problem of finding the hyperplanes with the largest separation margin. We will show that the dual problem directly identifies the support vectors, and that the dual formulation can give rise to a rich family of nonlinear classifications that are often more useful and more accurate than the linear hyperplane classification we presented here.

## Exercises

7.1. Consider two classes of data, where the points

$$(13.3),\ (0.31.5),\ (2,4.2),\ (2.2,2.9),\ (1.7,3.6),\ (3,4),\ (1,4)$$

possess a certain property and the points

$$(1.8,1.5),\ (3.4,3.6),\ (0.2,2.5),\ (1,1.3),\ (1,2.5),\ (3,1.1),\ (2,0.1)$$

do not possess this property. Use optimization software to compute the maximum margin hyperplane that separates the two classes of points. Are the classes indeed separable? What are the support vectors? Repeat the problem when the first class includes also the point (0.2, 2.5) and the second class includes the point (1.7, 3.6).

7.2. In this project we create a support vector machine for breast cancer diagnosis. We use the Wisconsin Diagnosis Breast Cancer Database (WDBC) made publicly available by Wolberg, Street, and Mangasarian of the University of Wisconsin. A link to the data base is made available on the Web page for this book, http://www.siam.org/books/ot108. There are two files: wdbc.data and wdbc.names. The file wdbc.names gives more details about the data, and you should read it to understand the context. The file wdbc.data gives $N = 569$ data vectors. Each data vector (in row form) has $n = 32$ components. The first component is the patient number, and the second is either "M" or "B" depending on whether the data is malignant or benign. You may manually change the entries "M" to "+1" and "B" to "−1". These entries are the indicators $y_i$. Elements 3 through 32 of each row $i$ form a 30-dimensional vector $x_i^T$ of observations.

(i) Use the first 500 data vectors as your training set. Use a modeling language to formulate the problem for the nonseparable case, using $C = 1000$. Solve the problem and display the separating hyperplane. Determine whether the data are indeed separable.

(ii) Use the output of the run to predict whether the remaining 69 patients have cancer. Compare your prediction to the actual patients' medical status. Evaluate the accuracy (proportion of correct predictions), the sensitivity (proportion of positive diagnoses for patients with the disease), and the specificity (the proportion of negative diagnoses for patients without the disease).

### 1.7.3 Portfolio Optimization

Suppose that an investor wishes to select a set of assets to achieve a good return on the investment while controlling risks of losses. The use of nonlinear models to manage investments began in the 1950s with the pioneering work of Nobel Prize laureate Harry Markowitz, who demonstrated how to reduce the risk of investment by selecting a *portfolio* of stocks rather than picking individual attractive stocks, and established the trade-off between reward and risk in investment portfolios.

An investment portfolio is defined by the vector $x = (x_1, \ldots, x_n)$, where $x_j$ denotes the proportion of the investment to be invested in asset $j$. Letting $\mu_j$ denote the expected rate of return of asset $j$, the expected rate of return of the portfolio is $\mu^T x$.

Let $\Sigma$ be the matrix of variances and covariances of the assets' returns. The entry $\Sigma_{j,j}$ is the variance of investment $j$. A high variance indicates high volatility or high risk; a low variance indicates stability or low risk. The entry $\Sigma_{i,j}$ is the covariance of investments $i$ and $j$. A positive value of $\Sigma_{i,j}$ indicates assets whose values usually move in the same direction, as often occurs with stocks of companies in the same industry. A negative value indicates assets whose values generally move in opposite directions—a desirable feature

in a diversified portfolio. Markowitz defined the risk of the portfolio to be its expected variance $x^T\Sigma x$.

Our optimization problem has two conflicting objectives: to maximize the return $\mu^T x$, and to minimize the risk $x^T\Sigma x$. The relative importance of these objectives will vary depending on the investor's tolerance for risk. We introduce a nonnegative parameter $\alpha$ that reflects the investor's trade-off between risk and return. The objective function in the model will be some combination of the two objectives, parameterized by $\alpha$, leading to the model

$$\text{maximize } f(x) = \mu^T x - \alpha x^T \Sigma x$$

subject to the constraints

$$\sum_i x_i = 1 \quad \text{and} \quad x \geq 0.$$

The value of $\alpha$ reflects the investor's aversion to risk. A large value indicates a reluctance to take on risk, with an emphasis on the stability of the investment. A low value indicates a high tolerance for risk with an emphasis on the expected return of the investment.

It can be difficult to choose a sensible value for $\alpha$. For this reason it is common to solve this model for a range of values of this parameter. This can reveal how sensitive the solution is to considerations of risk. The solution of the problem for any value of $\alpha$ is called *efficient* indicating that there is no other portfolio that has a larger expected return and a smaller variance.

There are of course some limitations to our model. First, we do not generally know the theoretical (joint) distribution of the assets' return and will need to estimate the mean and variance from historical data. Denoting the estimate of $\mu$ by $r$ and the estimate of $\Sigma$ by $V$, the actual problem we solve is

$$\begin{array}{ll} \text{maximize} & r^T x - \alpha x^T V x \\ \text{subject to} & \sum_i x_i = 1 \\ & x_i \geq 0. \end{array}$$

Second, investors should be aware that past performance is no indicator of future returns. Finally, we note that the matrix $V$ is *dense*; that is, it has many nonzero elements. As a result, when the number of assets is large, computations involving $V$ can be expensive thus making the optimization problem computationally difficult.

To illustrate portfolio optimization, consider an investor who is planning a portfolio based on four stocks. Data on the rates of return of the stocks in the last six periods are given in Table 1.3.

Using this information we estimate the mean of the rate of return as

$$r = (\,0.0667 \quad 0.0900 \quad 0.0717 \quad 0.0733\,),$$

and the variance as

$$V = \begin{pmatrix} 0.00019 & 0.00065 & 0.00004 & 0.00038 \\ 0.00065 & 0.00883 & 0.00218 & 0.00327 \\ 0.00004 & 0.00218 & 0.00125 & 0.00063 \\ 0.00308 & 0.00327 & 0.00063 & 0.00162 \end{pmatrix}.$$

**Table 1.3.** *Past rates of return of stocks.*

| Period | Stock 1 | Stock 2 | Stock 3 | Stock 4 |
|---|---|---|---|---|
| 1 | 0.08 | 0.05 | 0.01 | 0.08 |
| 2 | 0.06 | 0.17 | 0.09 | 0.12 |
| 3 | 0.07 | 0.05 | 0.10 | 0.07 |
| 4 | 0.04 | −0.07 | 0.04 | −0.01 |
| 5 | 0.08 | 0.12 | 0.08 | 0.09 |
| 6 | 0.07 | 0.22 | 0.11 | 0.09 |

**Table 1.4.** *Optimal portfolio for selected values of* $\alpha$.

| $\alpha$ | Stock 1 | Stock 2 | Stock 3 | Stock 4 | Mean | Variance |
|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 0.090 | $8.8 \times 10^{-3}$ |
| 2 | 0.12 | 0.65 | 0.23 | 0 | 0.083 | $4.5 \times 10^{-3}$ |
| 5 | 0.57 | 0.19 | 0.24 | 0 | 0.072 | $8.0 \times 10^{-4}$ |
| 10 | 0.71 | 0.04 | 0.25 | 0 | 0.069 | $2.6 \times 10^{-4}$ |
| 100 | 0.87 | 0 | 0.13 | 0 | 0.067 | $1.7 \times 10^{-4}$ |

The solution of the optimization problem for a selection of values of the parameter $\alpha$ is given in Table 1.4. Figure 1.12 plots the rate of return against the variance of the optimized portfolios for a continuous range of values of $\alpha$. The curved line is called the *efficient frontier* since it depicts the collection of all efficient points. The figure also shows the rate of return and variance obtained when allocating the entire portfolio to one stock only. In this example, a person who has a high tolerance for risk may choose to invest entirely in Stock 2, whereas a person who is extremely cautious may choose to invest entirely in Stock 1. Investing only in Stock 3, or only in Stock 4, or half in Stock 1 and half in Stock 2 are not recommended strategies for anyone, since they are dominated by strategies that have both higher return and lower risk.

## Exercises

7.1. How would the formulation to the problem change if a risk-free asset (such as government treasury bills at a fixed rate of return) is also being considered?

7.2. An investor wants to put together a portfolio consisting of the 30 stocks used to determine the Dow Jones industrial average. Use 25 weekly returns ending on the last Friday of last month to find the optimal portfolio. Experiment with different values of the parameter $\alpha$ and plot the corresponding points on the efficient frontier. You will need access to a nonlinear optimization solver. You may need to use a modeling language to formulate the problem for input to the solver.

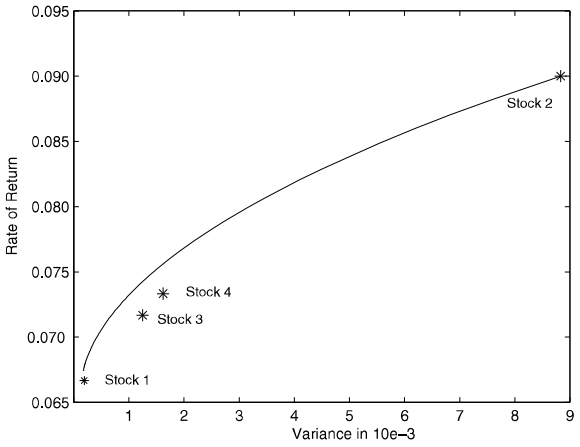


**Figure 1.12.** *Efficient frontier.*

### 1.7.4 Intensity Modulated Radiation Treatment Planning

Radiotherapy is the treatment of cancerous tissues with external beams of radiation. As a beam of radiation passes through the body, energy is deposited at points along its path, and as this happens the beam intensity gradually decreases (this is called attenuation). The radiation dosage is the amount of energy deposited locally per unit mass. High doses of radiation can kill cancerous cells, but will also damage nearby healthy cells. If vital organs receive too much radiation, serious complications may arise. Some limited damage to healthy cells may be tolerable however, since normal cells repair themselves more effectively than cancerous cells. If the radiation dosage is limited, the surrounding organs can continue to function and may eventually recover. The goal of the radiation treatment planning is to design a treatment that will kill the cancer in its entirety but limit the damage to surrounding healthy tissue.

To keep the radiation levels of normal healthy tissue low, the treatment typically uses several beams of radiation delivered from different angles. Intensity modulated radiation therapy (IMRT) is an important recent advance that allows each beam to be broken into hundreds (or possibly thousands) of beamlets of varying intensity. This is achieved using a set of metallic leaves (called collimators) that can sequentially move from open to closed position, thus filtering the radiation in a way that not only allows for the modulation of the intensity of the beam, but also enables control of its shape. This enables more accurate radiation treatment. This is particularly important in cases where the tumor has an unusual shape as is the case when it is wrapped around the spinal cord, or when it is close to a vital structure such as the optic nerve.

A simplified example of the desired goals for treatment of a hypothetical prostate cancer patient is given in Table 1.5. Radiation dosage is measured in a unit call Gray (Gy). One Gy is equal to one Joule of energy deposited in one kilogram. The planning target volume (PTV) describes a region large enough to incorporate the diseased organ, the