AN Introduction TO Optimization

Edwin K. P. Chong
and
Stanislaw H. Żak

Wiley-Interscience Series in
Discrete Mathematics and Optimization

An Introduction to Optimization

EDWIN K. P. CHONG

and

STANISLAW H. ŻAK

A Wiley-Interscience Publication
JOHN WILEY & SONS, INC.
New York · Chichester · Brisbane · Toronto · Singapore

Library of Congress Cataloging in Publication Data

Chong, Edwin K. P.
An introduction to optimization / Edwin K P Chong and Stanislaw H Zak
p. cm. — (Wiley-Interscience series in discrete mathematics and optimization)
Includes bibliographical references (p –) and index
ISBN 0-471-08949-4 (cloth : acid-free)
1. Mathematical optimization. I. Zak Stanislaw H
II. Title. III. Series.
QA402.5.Z35 1995
519.3-dc20 95 6111
CIP

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

Preface

Optimization is central to any problem involving decision making, whether in engineering or in economics. The task of decision making entails choosing between various alternatives. This choice is governed by our desire to make the "best" decision. The measure of goodness of the alternatives is described by an objective function or performance index. Optimization theory and methods deal with selecting the best alternative in the sense of the given objective function.

The area of optimization has received enormous attention in recent years, primarily because of the rapid progress in computer technology, including the development and availability of user-friendly software, high-speed and parallel processors, and artificial neural networks. A clear example of this phenomenon is the wide accessibility of optimization software tools such as MATLAB[1] and the many other commercial software packages.

There are currently several excellent graduate textbooks on optimization theory and methods [e.g., 4, 19, 22, 28, 51, 52, 59, 76], as well as undergraduate textbooks on the subject with an emphasis on engineering design [e.g., 2, 62]. However, there is a need for an introductory textbook on optimization theory and methods at a senior undergraduate or beginning graduate level. The present text was written with this goal in mind. The material is an outgrowth of our lecture notes for a one-semester course in optimization methods for seniors and first-year graduate students at Purdue University, West Lafayette. In our presentation, we assume a working knowledge of basic linear algebra and multivariable calculus. For the reader's convenience, a part of this book (Part I) is devoted to a review of the required mathematical background material. The many figures throughout the text complement the written presentation of the material. We also include a variety of exercises at the end of each chapter. A solutions manual with complete solutions to the exercises is available from the publisher to instructors who adopt this text. Some of the exercises require using MATLAB. The student edition of MATLAB is sufficient for almost all the MATLAB exercises included in the text. The MATLAB source listings for the MATLAB exercises are also included in the solutions manual.

The purpose of the book is to give the reader a working knowledge of optimization theory and methods. To accomplish this goal, we include numerous

[1] MATLAB is a registered trademark of The MathWorks, Inc.

examples that illustrate the theory and algorithms discussed in the text. However, it is not our intention to provide a cookbook of the most recent numerical techniques for optimization; rather, our goal is to equip the reader with sufficient background for further study of advanced topics in optimization.

The field of optimization is a very active research area. In recent years, various new approaches to optimization have been proposed. In this text, we have tried to reflect at least some of the flavor of recent activity in the area. For example, in our treatment of linear programming, we discuss not only the classical simplex method, but also the more recent methods of Khachiyan and Karmarkar for solving linear programs. There has also been a recent surge of applications of optimization methods to a variety of new problems. A prime example of this is the use of descent algorithms for the training of feedforward neural networks. An entire chapter in the book is devoted to this topic. The area of neural networks is an active area of ongoing research, and many books have been devoted to this subject. The topic of neural network training fits perfectly into the framework of unconstrained optimization methods. Therefore, the chapter on feedforward neural networks provides not only an example of application of unconstrained optimization methods, it also gives the reader an accessible introduction to what is currently a topic of wide interest. We also include a discussion of genetic algorithms, a topic becoming of increasing importance in the study of complex adaptive systems.

The material in this book is organized into four independent parts. Part I contains a review of some basic definitions, notations, and relations from linear algebra, geometry, and calculus that we use frequently throughout the book. In Part II, we consider unconstrained optimization problems. We first discuss some theoretical foundations of unconstrained optimization, including necessary and sufficient conditions for minimizers and maximizers. This is followed by a treatment of various iterative optimization algorithms, together with their properties. A discussion of genetic algorithms is included in this part. We also analyze the least-squares optimization problem and the associated recursive least-squares algorithm. Parts III and IV are devoted to constrained optimization. Part III deals with linear programming problems, which form an important class of constrained optimization problems. We give examples and analyze properties of linear programs and then discuss the simplex method for solving linear programs. We also provide a brief treatment of dual linear programming problems. We wrap up Part III by presenting non-simplex algorithms for solving linear programs, namely, Khachiyan's and Karmarkar's methods. In Part IV, we treat nonlinear constrained optimization. Here, as in Part II, we first present some theoretical foundations of nonlinear constratined optimization problems. We then discuss different algorithms for solving constrained optimization problems with equality as well as inequality constraints.

We are grateful to several people for their help during the course of writing this book. In particular, we thank Dennis Goodman of Lawrence Livermore Laboratories for his comments on early versions of Part II, and for making available to us his lecture notes on nonlinear optimization. We thank Moshe Kam of Drexel

University for pointing out some useful references on non-simplex methods. We are grateful to Ed Silverman and Russell Quong of Purdue University for their valuable remarks on Part I. We also thank the students of EE 580 for their many helpful comments and suggestions. In particular, we are grateful to Christopher Taylor for his diligent proofreading of early versions of the book.

E. K. P. CHONG
S. H. ŻAK

West Lafayette, Indiana

Contents

Part I

MATHEMATICAL REVIEW

1

Methods of Proof and Some Notation

1.1. METHODS OF PROOF

Consider two statements, "A" and "B," which could be either true or false. For example, let "A" be the statement "John is an engineering student," and let "B" be the statement "John is taking a course on optimization." We can combine these statements to form other statements, like "A and B" or "A or B." In our example, "A and B" means "John is an engineering student, and he is taking a course on optimization." We can also form statements like "not A," "not B," "not (A and B)," and so on. For example, "not A" means "John is not an engineering student." The truth or falsity of the combined statements depends on the truth or falsity of the original statements, "A" and "B." This relationship is expressed by means of truth tables (e.g., see Tables 1.1 and 1.2). From Tables 1.1 and 1.2, it is easy to see that the statement "not (A and B)" is equivalent to "(not A) or (not B)" (see Exercise 1.3). This is called *DeMorgan's Law.*

In proving statements, it is convenient to express a combined statement by a *conditional,* such as "A implies B," which we denote "A $\Rightarrow$ B." The conditional "A $\Rightarrow$ B" is simply the combined statement "(not A) or B," and is often also read "A only if B," "if A then B," "A is sufficient for B," or "B is necessary for A."

We can combine two conditional statements to form a *biconditional* statement of the form "A $\Leftrightarrow$ B," which simply means "(A $\Rightarrow$ B) and (B $\Rightarrow$ A)." The statement "A $\Leftrightarrow$ B" reads "A if, and only if, B," or "A is equivalent to B," or "A is necessary and sufficient for B." Truth tables for conditional and biconditional statements are given in Table 1.3.

It is easy to verify, using the truth table, that the statement "A $\Rightarrow$ B" is equivalent to the statement "(not B) $\Rightarrow$ (not A)." The latter is called the *contrapositive* of the former.

If we take the contrapositive to DeMorgan's Law, we obtain the assertion that "not (A or B)" is equivalent to "(not A) and (not B)."

Most statements we deal with have the form "A $\Rightarrow$ B." To prove such a statement, we may use one of the following three different techniques:

1. The direct method

Table 1.1 Truth Table for "A and B" and "A or B"

A	B	A and B	A or B
F	F	F	F
F	T	F	T
T	F	F	T
T	T	T	T

Table 1.2 Truth Table for "not A"

A	Not A
F	T
T	F

Table 1.3 Truth Table for Conditionals and Biconditionals

A	B	$A \Rightarrow B$	$A \Leftarrow B$	$A \Leftrightarrow B$
F	F	T	T	T
F	T	T	F	F
T	F	F	T	F
T	T	T	T	T

2. Proof by contraposition
3. Proof by contradiction or *reductio ad absurdum.*

In the case of the *direct method,* we start with "A," then deduce a chain of various consequences to end with "B."

A useful method for proving statements is *proof by contraposition,* based on the equivalence of the statements "$A \Rightarrow B$" and "(not B)$\Rightarrow$(not A)." We start with "not B," then deduce various consequences to end with "not A" as a conclusion.

Another method of proof that we use is *proof by contradiction,* based on the equivalence of the statements "$A \Rightarrow B$" and "not (A and (not B))." Here we begin with "A and (not B)" and derive a contradiction.

Occasionally, we will use the *Principle of Induction* to prove statements. This principle may be stated as follows. Assume that a given property of positive integers satisfies the following conditions:

The number 1 possesses this property;

If the number n possesses this property, then the number $n + 1$ possesses it too.

The Principle of Induction states that under these assumptions any positive integer possesses the property.

The Principle of Induction is easily understood using the following intuitive argument. If the number 1 possesses the given property, then the second condition implies that the number 2 possesses the property. But, then again, the second condition implies that the number 3 possesses this property, and so on. The Principle of Induction is a formal statement of this intuitive reasoning.

1.2. NOTATION

Throughout, we use the following notation. If X is a set, then we write $x \in X$ to mean that x is an element of X. When an object x is not an element of a set X, then we write $x \notin X$. We also use the "curly bracket notation" for sets, writing down the first few elements of a set followed by three dots. For example, $\{x_1, x_2, x_3, \ldots\}$ is the set containing the elements x_1, x_2, x_3, and so on. Alternatively, we can explicitly display the law of formation. For example, $\{x : x \in \mathbb{R}, x > 5\}$ reads "the set of all x such that x is real and x is greater than 5." The colon following x reads "such that." An alternative notation for the same set is $\{x \in \mathbb{R} : x > 5\}$.

If X and Y are sets, then we write $X \subset Y$ to mean that every element of X is also an element of Y. In this case, we say that X is a *subset* of Y. If X and Y are sets, then we denote by $X \setminus Y$ the set of all points in X that are not in Y. Note that $X \setminus Y$ is a subset of X. The notation $f : X \to Y$ means "f is a function from the set X into the set Y." The symbol $:=$ denotes arithmetic assignment. Thus, a statement of the form $x := y$ means "x becomes y." The symbol $\triangleq$ means "equals by definition."

Throughout the text, we mark the end of theorems, lemmas, propositions, and corollaries using the symbol □. We mark the end of proofs, definitions, and examples by ■.

EXERCISES

1.1 Construct the truth table for the statement "(not B)$\Rightarrow$(not A)," and use it to show that this statement is equivalent to the statement "A$\Rightarrow$B."

1.2 Construct the truth table for the statement "not (A and (not B))," and use it to show that this statement is equivalent to the statement "A$\Rightarrow$B."

1.3 Prove DeMorgan's Law by constructing the appropriate truth tables.

1.4 Prove that for any statements A and B, we have "A$\Leftrightarrow$(A and B)" or "(A and (not B))." This is useful because it allows us to prove a statement A by proving the two separate cases "(A and B)," and "(A and (not B))." For example, to prove that $|x| \geqslant x$ for any $x \in \mathbb{R}$, we separately prove the cases "$|x| \geqslant x$ and $x \geqslant 0$," and "$|x| \geqslant x$ and $x < 0$." Proving the two cases turns out to be easier than directly proving the statement $|x| \geqslant x$ (see Section 2.4 and Exercise 2.4).

1.5 Suppose you are shown four cards, laid out in a row. Each card has a letter on one side and a number on the other. On the visible side of the cards are printed the symbols:

$$S \quad 8 \quad 3 \quad A$$

Determine which cards you should turn over to decide if the following rule is true or false: "If there is a vowel on one side of the card, then there is an even number on the other side."

2

Real Vector Spaces and Matrices

2.1. REAL VECTOR SPACES

We define a column n vector to be an array of n numbers, denoted by

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}.$$

The number a_i is called the ith component of the vector $\boldsymbol{a}$. Denote by $\mathbb{R}$ the set of real numbers, and by $\mathbb{R}^n$ the set of column n-vectors with real components. We call $\mathbb{R}^n$ an n-dimensional *real vector space.* We commonly denote elements of $\mathbb{R}^n$ by lower case bold letters, for example, $\boldsymbol{x}$. The components of $\boldsymbol{x} \in \mathbb{R}^n$ are denoted by $x_1, \ldots, x_n$.

We define a *row* n-vector as

$$[a_1, a_2, \ldots, a_n].$$

The *transpose* of a given column vector $\boldsymbol{a}$ is a row vector with corresponding elements, denoted $\boldsymbol{a}^T$. For example, if

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix},$$

then

$$\boldsymbol{a}^T = [a_1, a_2, \ldots, a_n].$$

Equivalently, we may write $\boldsymbol{a} = [a_1, a_2, \ldots, a_n]^T$. Throughout the text, we adopt