NONLINEAR AND DYNAMIC PROGRAMMING

G. HADLEY

NONLINEAR AND DYNAMIC PROGRAMMING

by

G. HADLEY

University of Chicago

and

Universidad de Los Andes
Bogotá, Colombia

ADDISON-WESLEY PUBLISHING COMPANY, INC.

READING, MASSACHUSETTS · PALO ALTO · LONDON

PREFACE

This work is intended as a sequel to the author's *Linear Programming*, and it concentrates on a study of the theory and computational aspects of nonlinear programming problems. The first chapter surveys in simple terms those features of nonlinear programming problems which make them much more difficult to solve than linear ones. Chapter 2 attempts to summarize the most important mathematical background needed and introduces the notation to be used throughout the text. By the inclusion of this chapter it is hoped that the current work can be used independently of the author's *Linear Programming* or *Linear Algebra* text. Chapter 3 concentrates on classical optimization methods based on the calculus, mainly the method of Lagrange multipliers. It also develops the most important properties of convex and concave functions that will be needed in the remainder of the text. No attempt is made to study the calculus of variations in this chapter since an entire book would be required to develop this subject adequately. Indeed, the only references to the calculus of variations which appear in this text are brief sections in the chapters on dynamic programming. Chapter 4 discusses approximate methods for finding either a local or global optimum (depending on the nature of the problem) for nonlinear programming problems. Chapter 5 is devoted to stochastic programming problems and Chapter 6 to the Kuhn-Tucker theory. In Chapter 7 quadratic programming problems are studied and in Chapter 8 integer linear programming is discussed. Chapter 9 is concerned with gradient methods for solving programming problems, and the remaining two chapters deal with dynamic programming.

The subject matter of nonlinear programming is much broader and much more difficult to unify than is that concerned only with linear programming. For this reason, the current text may appear to be somewhat lacking in cohesiveness and may seem to be concerned with a set of somewhat unrelated special methods for solving particular types of problems. This is true to a considerable extent, because there does not at present exist any unifying theory for all of what might be considered nonlinear programming and because, in addition, computational algorithms exist only for solving very special types of problems. The development of these algorithms depends in a crucial way on the special characteristics of the problems. A very large number of computational schemes have been suggested for solving a variety of the problems considered here. It was out of the question to discuss every method that has been proposed for solving a particular type of problem, and hence the author was forced to make a choice in selecting the ones to be discussed. In many cases, this had to be done on an arbitrary basis, because no data exist for comparing the computational efficiency of various schemes. In some cases, a method may have been omitted simply because the author was not aware of its existence. The field of nonlinear programming is currently in a state of rapid change. In selecting material for this text, an attempt was made to include those methods which would be more or less of lasting relevance. However, it is quite possible that many of the computational algorithms discussed will be superseded by improved ones in the not too distant future.

The mathematical background needed for a study of this work varies considerably from chapter to chapter. For most of the chapters a rudimentary knowledge of linear algebra and linear programming is needed (say, roughly, the first eight or nine chapters of the author's *Linear Programming*). However, a large part of the material on dynamic programming could be read without a knowledge of either linear algebra or linear programming. Some knowledge of calculus is needed in Chapters 3 and 6, but not elsewhere to any significant extent. Chapter 2 attempts to summarize most of the necessary mathematical background referred to above. There remains one background area which Chapter 2 does not attempt to summarize. This is the subject matter of elementary probability theory. Discussions involving the use of probability theory appear in Chapters 3, 5, 10, and 11.

The author is indebted once again to Jackson E. Morris, who has so generously supplied the excellent quotations appearing at the beginning of each chapter. The reviewers, Robert Dorfman and Stuart Dreyfus, provided a number of useful suggestions for which the author is grateful. The Graduate School of Business, University of Chicago, generously provided the secretarial assistance for having the manuscript typed and reproduced.

Bogotá, Colombia G.H.
June 1964

CONTENTS

CHAPTER 1

INTRODUCTION

Each venture
Is a new beginning, a raid on the inarticulate
With shabby equipment always deteriorating.

T. S. Eliot, *East Coker*

1–1 Programming problems. Any problem which seeks to maximize or minimize a numerical function of one or more variables (or functions) when the variables (or functions) can be independent or related in some way through the specification of certain constraints may be referred to as an optimization problem. Optimization problems have long been of interest to mathematicians, physical scientists, and engineers. The possibility of using the methods of the differential calculus and the calculus of variations to solve certain types of optimization problems arising in geometry and physics has been known and applied since the middle of the eighteenth century. In the last fifteen years there has been a remarkable growth of interest in a new class of optimization problems, often referred to as programming problems, which are usually not amenable to solution by the classical methods of the calculus. Programming problems can frequently be classified in a broad context as problems in economics, rather than problems in geometry or physics as were the classical optimization problems. Often a programming problem can be considered to be one concerned with the allocation of scarce resources—men, machines, and raw materials—to the manufacture of one or more products in such a way that the products meet certain specifications, while at the same time some objective function such as profit or cost is maximized or minimized. Programming problems have attracted such wide interest because they do not occur in theoretical economics only, but also arise, in the form of important practical problems, in industry, commerce, government, and the military.

The general programming problem can be formulated as follows. It is desired to determine values for n variables $x_1, \ldots, x_n$ which satisfy the m inequalities or equations

$$g_i(x_1, \ldots, x_n)\{\leq, =, \geq\}b_i, \qquad i = 1, \ldots, m, \tag{1–1}$$

and, in addition, maximize or minimize the function

$$z = f(x_1, \ldots, x_n). \tag{1–2}$$

The restrictions (1–1) are called the *constraints,* and (1–2) is called the *objective function.* In (1–1) the $g_i(x_1, \ldots, x_n)$ are assumed to be specified functions, and the b_i are assumed to be known constants. Furthermore, in (1–1), one and only one of the signs $\leq, =, \geq$ holds for each constraint, but the sign may vary from one constraint to another. The values of m and n need not be related in any way, that is, m can be greater than, less than, or equal to n. We shall allow m to be zero, so that we include cases where there are no constraints (1–1). Usually, some or all of the variables are restricted to be non-negative. In addition, it may be required that some or all of the variables are allowed to take on only certain discrete values, such as integral values. Unless otherwise specified, (1–1) and (1–2) will be interpreted as a problem in which it is desired to find numerical values for the n variables $x_1, \ldots, x_n$ which optimize (1–2) subject to (1–1) and any non-negativity and/or integrality requirements. In certain cases, however, the variables x_j will be functions of one or more parameters, and the problem will then be one of determining a set of functions $x_1, \ldots, x_n$ which optimize (1–2) subject to (1–1) and any non-negativity and/or integrality requirements on the x_j.

The real impetus for the growth of interest in and the practical applications of programming problems came in 1947, when George Dantzig devised the simplex algorithm for solving the general linear programming problem. If in (1–1) and (1–2),

$$g_i(x_1, \ldots, x_n) = \sum_{j=1}^{n} a_{ij}x_j, \qquad i = 1, \ldots, m, \tag{1–3}$$

and

$$f(x_1, \ldots, x_n) = \sum_{j=1}^{n} c_j x_j, \tag{1–4}$$

where the a_{ij} and c_j are known constants, the programming problem is said to be linear provided that there are no other restrictions except perhaps the requirement that some or all variables must be non-negative. Usually, in the formulation of the general linear programming problem, it is specified that *each* variable must be non-negative, i.e.,

$$x_j \geq 0, \qquad j = 1, \ldots, n, \tag{1–5}$$

since this form is most convenient when making numerical computations. Any problem in which some of the variables are unrestricted in sign may easily be transformed to one in which all variables are non-negative. Thus a linear programming problem seeks to determine non-negative values of the n variables x_j which satisfy the m constraints

$$\textstyle\sum_j a_{ij}x_j \{\leq, =, \geq\} b_i,$$

and which maximize or minimize the linear function $z = \sum_j c_j x_j$. We shall not consider a problem of the above form to be linear unless all x_j values which satisfy the constraints and the non-negativity restrictions are allowable, i.e., it is not permissible to impose an additional restriction, for example, that the variables can only assume integral values. All programming problems that are not linear in the sense defined above will be called nonlinear.

In this text we shall be concerned almost exclusively with the solution of nonlinear programming problems. It will be assumed that the reader is familiar with linear programming to the extent covered in the first nine chapters of the author's *Linear Programming* [13].* This is an essential prerequisite, since many of the techniques for the solution of nonlinear problems involve in one way or another the use of a simplex-type algorithm. Unfortunately, nonlinear programming problems are almost always much more difficult to solve than linear ones. Indeed, computational procedures have been devised for solving only a very small subset of all nonlinear programming problems. We shall study those that can be solved and the techniques available for solving them. No attempt will be made to study in detail the great variety of practical problems that can be formulated as nonlinear programming problems. It might be noted, however, that most practical problems which have been formulated as linear programming problems are in reality nonlinear ones for which the nonlinearities were ignored or approximated in some way. Although no attempt will be made to cover formally the applications of nonlinear programming to practical situations, a number of practical applications will be considered in the discussion of examples and in the problems.

1–2 Special cases of interest. Most of the following chapters will either be concerned with a study of techniques for solving very specialized types of nonlinear programming problems, or will be devoted to examining the types of nonlinear programming problems that can be solved with some particular computational technique. It seems desirable at the outset to review briefly the special types of nonlinear programming problems that will receive the greatest attention, and the general types of computational techniques that have been found useful in solving such problems. This section will deal with the special types of problems to be considered in detail later, and the next section will be concerned with computational techniques. In the discussion to follow it will be assumed that the variables are *not* restricted to integral or, more generally, discrete values unless it is indicated specifically that they are so restricted.

* Numbers in brackets refer to bibliographical references.

The class of nonlinear programming problems which has been studied most extensively is that where the constraints are linear and the objective function is nonlinear. The general problem of this kind can be written in the abbreviated format

$$\sum_{j=1}^{n} a_{ij}x_j \{\leq, =, \geq\} b_i, \qquad i = 1, \ldots, m, \tag{1-6}$$

$$x_j \geq 0, \qquad j = 1, \ldots, n, \tag{1-7}$$

$$\max \text{ or } \min z = f(x_1, \ldots, x_n). \tag{1-8}$$

Equations (1–6) through (1–8) should be read: Find non-negative values of the n variables x_j which satisfy the constraints (1–6) and which maximize or minimize the objective function $z = f(x_1, \ldots, x_n)$. For convenience, the variables were required to be non-negative in the above formulation. Problems in which some or all variables are allowed to be unrestricted in sign may be easily reduced to this case by a simple transformation that will be introduced later.

Even when attention is restricted to problems involving linear constraints, computational techniques for finding optimal solutions have not been devised except in cases where the objective function has very special properties. There are two special cases of (1–6) through (1–8) that will be of particular interest to us. In the first, the objective function can be written as a sum of n functions, each of which is a function of only a single variable, i.e.,

$$z = f(x_1, \ldots, x_n) = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n). \tag{1-9}$$

To guarantee that an optimal solution can be found, additional restrictions must be placed on the $f_j(x_j)$. These will be discussed later. When the objective function can be written in the form (1–9), it is said to be separable. Occasionally, when studying problems with separable objective functions and a very small number of linear constraints, we shall also consider cases where the variables are restricted to taking on only integral values.

In the second case, the objective function can be written as the sum of a linear form plus a quadratic form, so that

$$\begin{aligned} z = f(x_1, \ldots, x_n) &= \sum_{j=1}^{n} c_j x_j + \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_i x_j \\ &= c_1x_1 + \cdots + c_nx_n + d_{11}x_1^2 + d_{12}x_1x_2 + \cdots + d_{1n}x_1x_n \\ &\qquad + \cdots + d_{nn}x_n^2. \end{aligned} \tag{1-10}$$

Such a nonlinear programming problem is referred to as a quadratic programming problem. To be certain that an optimal solution can be found in this case, the d_{ij} must satisfy certain restrictions which need not be considered now. An entire chapter, Chapter 7, will be devoted to studying the theory of, and techniques for, solving quadratic programming problems.

For a variety of reasons that will become clear later, problems with nonlinear constraints tend to be much more difficult to solve than those with linear constraints. We shall devote parts of several chapters to studying problems in which the constraints may be nonlinear. Most of our attention will be limited, however, to cases in which the constraints are separable. This means that the $g_i(x_1, \ldots, x_n)$ in (1–1) must be capable of being written

$$g_i(x_1, \ldots, x_n) = g_{i1}(x_1) + \cdots + g_{in}(x_n). \tag{1–11}$$

To ensure that we can obtain an optimal solution to problems with nonlinear constraints of the form (1–11), very stringent restrictions must be placed both on the $g_{ij}(x_j)$ and on the objective function. Although it is profitable to study the special case of a nonseparable objective function which is the sum of a linear and a quadratic form, this is not true in the corresponding case for nonseparable constraints, and we shall not attempt to do so.

There is one other class of problems which may have nonlinear constraints that we shall spend some time studying. Problems in this class are referred to as classical optimization problems. To obtain the form of this kind of problem, let us return to the general nonlinear programming problem (1–1) and (1–2) and imagine that (1) no inequalities appear in the constraints, (2) there are no non-negativity or discreteness restrictions on the variables, (3) $m < n$, and (4) the functions $g_i(x_1, \ldots, x_n)$ and $f(x_1, \ldots, x_n)$ are continuous and possess partial derivatives at least through second order. Such a programming problem can be represented as

$$\begin{gathered} g_i(x_1, \ldots, x_n) = b_i, \qquad i = 1, \ldots, m, \\ \max \text{ or } \min z = f(x_1, \ldots, x_n). \end{gathered} \tag{1–12}$$

A problem of this kind will be called a *classical optimization problem*. Problems such as (1–12) can be solved, at least in principle, by means of the classical optimization techniques based on the calculus. While there are problems (1–12) which can actually be solved numerically in this manner, one usually encounters computational difficulties of such a magnitude that it becomes mandatory to attempt some other method of solution. Indeed, we shall not even classify the classical techniques as computational devices, but instead as theoretical tools. It is important, however, to have some familiarity with the classical techniques, because in many areas they form the basis for the theoretical analyses employed. For example, much of the standard theory of production and consumer behavior in economics is based on classical optimization methods. Our main reason, then, for studying problems of the form (1–12) as a special class will be to develop the theory of classical optimization methods rather than to develop numerical procedures for solving problems of this type. Classical optimization techniques will be studied in Chapter 3.

The classical techniques can be generalized to handle cases in which the variables are required to be non-negative and the constraints may be inequalities, but again these generalizations are primarily of theoretical value and do not usually constitute computational procedures. Nonetheless, we shall see that these theoretical results will be very useful. In one particular case, that of quadratic programming, they will indirectly provide a computational method, i.e., they will provide another problem (which can be solved by the methods to be considered in the next section) whose solution yields an optimal solution to the quadratic programming problem. Chapter 6 will be devoted to a study of these theoretical generalizations of classical optimization techniques.

Another class of nonlinear programming problems that will be of interest to us is obtained from the general linear programming problem by imposing the additional requirement that the variables can take on only integral values. Such problems are frequently referred to as integer linear programming problems. They can be represented mathematically as

$$\sum_{j=1}^{n} a_{ij}x_j \;\{\le, =, \ge\}\; b_i, \qquad i = 1, \ldots, m,$$

$$x_j \ge 0, \qquad j = 1, \ldots, n; \qquad \text{some or all } x_j \text{ integers}, \tag{1–13}$$

$$\max \text{ or } \min z = \sum_{j=1}^{n} c_j x_j.$$

If all x_j are required to be integers, the problem is called an all integer problem. Otherwise, we shall refer to it as a mixed integer-continuous variable problem. Chapter 8 will be concerned entirely with developing means for solving integer linear programming problems and showing that a wide variety of interesting problems can be formulated in this manner.

One other special type of nonlinear programming problem will be studied in some detail. This will be a type of stochastic sequential decision problem which is frequently encountered in production planning and inventory control. We shall not attempt to provide its mathematical form now. It will, however, exemplify the case where the variables x_j are functions of other parameters so that, instead of determining a set of numerical values for the x_j, one must determine a set of functions.

This completes the summary of the special types of problems that will occupy most of our attention in the remainder of this work. Other kinds of programming problems will be considered, but they will not be discussed as extensively as the special classes of problems considered in this section. We must restrict ourselves in large measure to special classes of problems, because the computational procedures for finding optimal solutions rely heavily on the special features and structures of the problems.

1–3 Computational procedures. The simplex algorithm for solving the general linear programming problem is an iterative procedure which yields an exact optimal solution in a finite number of steps. For the nonlinear programming problems to be studied in this text, we shall not always be able to devise computational procedures which yield exactly an optimal solution in a finite number of steps. One must often settle for procedures which provide only an approximate optimal solution or which may require an infinite number of steps for convergence.

One of the most powerful techniques for solving nonlinear programming problems is to transform the problem by some means into a form which permits application of the simplex algorithm (or one of the simplex-type algorithms). Thus the simplex algorithm turns out to be one of the most powerful computational devices for solving nonlinear programming problems as well as for solving linear programming problems. The nature of the "transformation" required to change a nonlinear programming problem into a form permitting the use of the simplex method varies widely with the type of problem being studied. In certain cases no approximations are needed to obtain a problem to which the simplex method can be applied, whereas in others approximations must be made. However, these approximations may be made as accurate as one desires (at the expense of increased computational effort).

Another useful computational technique for solving certain types of nonlinear programming problems is *dynamic programming*. The title of this text might suggest that dynamic programming refers to a special class of programming problems that are in some way distinct from nonlinear ones. Indeed there is some justification in this assumption, since the term dynamic programming is often used to refer to programming problems where changes occur over time and hence time must be considered explicitly. We shall not use dynamic programming in this sense. Instead, we shall take dynamic programming to mean the computational method involving recurrence relations which has been developed to a considerable extent by Richard Bellman. This technique evolved as a result of studying programming problems in which changes over time were important, and this is why it was given the name "dynamic programming." However, the technique can be applied to problems in which time is in no way relevant. Hence a different name would be desirable, but the term "dynamic programming" has now become so firmly established that change would prove difficult.

It should be pointed out that we are using the term "dynamic programming" in a rather narrow sense, although one will frequently find it used to describe a special type of computational procedure. The analysis of a very broad class of functional equations is also often considered to be a part of dynamic programming, in which case the procedure becomes an analytical as well as a computational tool. Indeed, we shall even discuss briefly the analysis of certain functional equations in our discussion of dynamic programming, so that we are not treating it as a strictly computational technique. Basically, however, we shall consider the term to refer to a special kind of computational procedure.

The final computational algorithm which will be used is referred to as the *gradient method*. Like the simplex method, it is an iterative technique in which we move at each step from one feasible solution to another in such a way that the value of the objective function is improved. It differs from the simplex method in that it is not an adjacent extreme point technique. In general, gradient methods may require an infinite number of iterations for convergence.

1–4 Difficulties introduced by nonlinearities. Before entering into a detailed discussion of any particular class of nonlinear programming problems, we shall examine some of the characteristics of nonlinear phenomena that can make it much more difficult to solve nonlinear programming problems than it is to solve linear ones. Recall that linear programming problems have the following properties:

(a) The set of feasible solutions [i.e., the set of all n-tuples $[x_1, \ldots, x_n]$ which satisfy the constraints (1–6) and the non-negativity restrictions (1–7)] is a convex set. This convex set has a finite number of corners which are usually referred to as extreme points.

(b) The set of all n-tuples $[x_1, \ldots, x_n]$ which yield a specified value of the objective function is a hyperplane. Furthermore, the hyperplanes corresponding to different values of the objective function are parallel.

(c) A local maximum or minimum is also the absolute (global) maximum or minimum of the objective function over the set of feasible solutions, i.e., there do not exist local optima of the objective function different from the global optimum. A feasible solution yields the absolute maximum of the objective function if the value of z for this feasible solution is at least as great as that for any other feasible solution, whereas, roughly speaking, a feasible solution yields a local maximum of the objective function if the value of z for this feasible solution is greater than the value of z for nearby feasible solutions.

(d) If the optimal value of the objective function is bounded, at least one of the extreme points of the convex set of feasible solutions will be an optimal solution. Furthermore, starting at any extreme point of the con-