Nonlinear Programming

13

Numerous mathematical-programming applications, including many introduced in previous chapters, are cast naturally as linear programs. Linear programming assumptions or approximations may also lead to appropriate problem representations over the range of decision variables being considered. At other times, though, nonlinearities in the form of either nonlinear objectivefunctions or nonlinear constraints are crucial for representing an application properly as a mathematical program. This chapter provides an initial step toward coping with such nonlinearities, first by introducing several characteristics of nonlinear programs and then by treating problems that can be solved using simplex-like pivoting procedures. As a consequence, the techniques to be discussed are primarily algebra-based. The final two sections comment on some techniques that do not involve pivoting.

As our discussion of nonlinear programming unfolds, the reader is urged to reflect upon the linear-programming theory that we have developed previously, contrasting the two theories to understand why the nonlinear problems are intrinsically more difficult to solve. At the same time, we should try to understand the similarities between the two theories, particularly since the nonlinear results often are motivated by, and are direct extensions of, their linear analogs. The similarities will be particularly visible for the material of this chapter where simplex-like techniques predominate.

13.1 NONLINEAR PROGRAMMING PROBLEMS

A general optimization problem is to select n decision variables $x_1, x_2, \ldots, x_n$ from a given feasible region in such a way as to optimize (minimize or maximize) a given objective function

$$f(x_1, x_2, \ldots, x_n)$$

of the decision variables. The problem is called a *nonlinear programming problem* (NLP) if the objective function is nonlinear and/or thefeasible region is determined by nonlinear constraints. Thus, in maximization form, the general nonlinear program is stated as:

$$\text{Maximize } f(x_1, x_2, \ldots, x_n),$$

subject to:

$$\begin{aligned} g_1(x_1, x_2, \ldots, x_n) &\leq b_1, \\ &\vdots \\ g_m(x_1, x_2, \ldots, x_n) &\leq b_m, \end{aligned}$$

where each of the constraint functions g_1 through g_m is given. A special case is the linear program that has been treated previously. The obvious association for this case is

$$f(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{n} c_j x_j,$$

and

$$g_i(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{n} a_{ij} x_j \qquad (i = 1, 2, \ldots, m).$$

Note that nonnegativity restrictions on variables can be included simply by appending the additional constraints:

$$g_{m+i}(x_1, x_2, \ldots, x_n) = -x_i \leq 0 \qquad (i = 1, 2, \ldots, n).$$

Sometimes these constraints will be treated explicitly, just like any other problem constraints. At other times, it will be convenient to consider them implicitly in the same way that nonnegativity constraints are handled implicitly in the simplex method.

For notational convenience, we usually let x denote the vector of n decision variables $x_1, x_2, \ldots, x_n$ — that is, $x = (x_1, x_2, \ldots, x_n)$ — and write the problem more concisely as

$$\text{Maximize } f(x),$$

subject to:

$$g_i(x) \leq b_i \qquad (i = 1, 2, \ldots, m).$$

As in linear programming, we are not restricted to this formulation. To minimize $f(x)$, we can of course maximize $-f(x)$. Equality constraints $h(x) = b$ can be written as two inequality constraints $h(x) \leq b$ and $-h(x) \leq -b$. In addition, if we introduce a slack variable, each inequality constraint is transformed to an equality constraint. Thus sometimes we will consider an alternative equality form:

$$\text{Maximize } f(x),$$

subject to:

$$\begin{aligned} h_i(x) &= b_i \qquad (i = 1, 2, \ldots, m) \\ x_j &\geq 0 \qquad (j = 1, 2, \ldots, n). \end{aligned}$$

Usually the problem context suggests either an equality or inequality formulation (or a formulation with both types of constraints), and we will not wish to force the problem into either form.

The following three simplified examples illustrate how nonlinear programs can arise in practice.

Portfolio Selection An investor has \$5000 and two potential investments. Let x_j for $j = 1$ and $j = 2$ denote his allocation to investment j in thousands of dollars. From historical data, investments 1 and 2 have an expected annual return of 20 and 16 percent, respectively. Also, the total risk involved with investments 1 and 2, as measured by the variance of total return, is given by $2x_1^2 + x_2^2 + (x_1 + x_2)^2$, so that risk increases with total investment and with the amount of each individual investment. The investor would like to maximize his expected return and at the same time minimize his risk. Clearly, both of these objectives cannot, in general, be satisfied simultaneously. There are several possible approaches. For example, he can minimize risk subject to a constraint imposing a lower bound on expected return. Alternatively, expected return and risk can be combined in an objective function, to give the model:

$$\text{Maximize } f(x) = 20x_1 + 16x_2 - \theta[2x_1^2 + x_2^2 + (x_1 + x_2)^2],$$

subject to:

$$\begin{aligned} g_1(x) &= x_1 + x_2 \leq 5, \\ x_1 &\geq 0, \quad x_2 \geq 0, \qquad \text{(that is, } g_2(x) = -x_1, \qquad g_3(x) = -x_2). \end{aligned}$$

The nonnegative constant θ reflects his tradeoff between risk and return. If $\theta = 0$, the model is a linear program, and he will invest completely in the investment with greatest expected return. For very large θ, the objective contribution due to expected return becomes negligible and he is essentially minimizing his risk.

Water Resources Planning In regional water planning, sources emitting pollutants might be required to remove waste from the water system. Let x_j be the pounds of Biological Oxygen Demand (an often-used measure of pollution) to be removed at source j.

One model might be to minimize total costs to the region to meet specified pollution standards:

$$\text{Minimize } \sum_{j=1}^{n} f_j(x_j),$$

subject to:

$$\sum_{j=1}^{n} a_{ij}x_j \geq b_i \quad (i = 1, 2, \ldots, m)$$
$$0 \leq x_j \leq u_j \qquad (j = 1, 2, \ldots, n),$$

where

$f_j(x_j) =$ Cost of removing x_j pounds of Biological Oxygen Demand at source j,
$b_i =$ Minimum desired improvement in water quality at point i in the system,
$a_{ij} =$ Quality response, at point i in the water system, caused by removing one pound of Biological Oxygen Demand at source j,
$u_j =$ Maximum pounds of Biological Oxygen Demand that can be removed at source j.

Constrained Regression A university wishes to assess the job placements of its graduates. For simplicity, it assumes that each graduate accepts either a government, industrial, or academic position. Let

$$N_j = \text{Number of graduates in year } j \quad (j = 1, 2, \ldots, n),$$

and let G_j, I_j, and A_j denote the number entering government, industry, and academia, respectively, in year j ($G_j + I_j + A_j = N_j$).

One model being considered assumes that a given fraction of the student population joins each job category each year. If these fractions are denoted as λ_1, λ_2, and λ_3, then the predicted number entering the job categories in year j is given by the expressions

$$\hat{G}_j = \lambda_1 N_j,$$
$$\hat{I}_j = \lambda_2 N_j,$$
$$\hat{A}_j = \lambda_3 N_j.$$

A reasonable performance measure of the model's validity might be the difference between the actual number of graduates G_j, I_j, and A_j entering the three job categories and the predicted numbers $\hat{G}_j$, $\hat{I}_j$, and $\hat{A}_j$, as in the least-squares estimate:

$$\text{Minimize } \sum_{j=1}^{n} [(G_j - \hat{G}_j)^2 + (I_j - \hat{I}_j)^2 + (A_j - \hat{A}_j)^2],$$

subject to the constraint that all graduates are employed in one of the professions. In terms of the fractions entering each profession, the model can be written as:

$$\text{Minimize } \sum_{j=1}^{n} [(G_j - \lambda_1 N_j)^2 + (I_j - \lambda_2 N_j)^2 + (A_j - \lambda_3 N_j)^2],$$

subject to:

$$\lambda_1 + \lambda_2 + \lambda_3 = 1,$$
$$\lambda_1 \geq 0, \quad \lambda_2 \geq 0, \quad \lambda_3 \geq 0.$$

This is a nonlinear program in three variables λ_1, λ_2, and λ_3.

There are alternative ways to approach this problem. For example, the objective function can be changed to:

$$\text{Minimize} \sum_{j=1}^{n} \left[|G_j - \hat{G}_j| + |I_j - \hat{I}_j| + |A_j - \hat{A}_j| \right].^{\dagger}$$

This formulation is appealing since the problem now can be transformed into a linear program. Exercise 28 (see also Exercise 20) from Chapter 1 illustrates this transformation.

The range of nonlinear-programming applications is practically unlimited. For example, it is usually simple to give a nonlinear extension to any linear program. Moreover, the constraint $x = 0$ or 1 can be modeled as $x(1 - x) = 0$ and the constraint x integer as $\sin(\pi x) = 0$. Consequently, *in theory* any application of integer programming can be modeled as a nonlinear program. We should not be overly optimistic about these formulations, however; later we shall explain why nonlinear programming is not attractive for solving these problems.

13.2 LOCAL vs. GLOBAL OPTIMUM

Geometrically, nonlinear programs can behave much differently from linear programs, even for problems with linear constraints. In Fig. 13.1, the portfolio-selection example from the last section has been plotted for several values of the tradeoff parameter θ. For each fixed value of θ, contours of constant objective values are concentric ellipses. As Fig. 13.1 shows, the optimal solution can occur:

a) at an interior point of the feasible region;
b) on the boundary of the feasible region, which is not an extreme point; or
c) at an extreme point of the feasible region.

As a consequence, procedures, such as the simplex method, that search only extreme points may not determine an optimal solution.

Figure 13.2 illustrates another feature of nonlinear-programming problems. Suppose that we are to minimize $f(x)$ in this example, with $0 \leq x \leq 10$. The point $x = 7$ is optimal. Note, however, that in the indicated dashed interval, the point $x = 0$ is the best feasible point; i.e., it is an optimal feasible point in the local vicinity of $x = 0$ specified by the dashed interval.

The latter example illustrates that a solution optimal in a local sense need not be optimal for the overall problem. Two types of solution must be distinguished. A global optimum is a solution to the overall optimization problem. Its objective value is as good as any other point in the feasible region. A local optimum, on the other hand, is optimal only with respect to feasible solutionsclose to that point. Points far removed from a local optimum play no role in its definition and may actually be preferred to the local optimum. Stated more formally,

Definition. Let $x = (x_1, x_2, \ldots, x_n)$ be a feasiblesolution to a maximization problem with objective function $f(x)$. We call x

1. A *global maximum* if $f(x) \geq f(y)$ for every feasible point $y = (y_1, y_2, \ldots, y_n)$;

† $|\ \ |$ denotes absolute value; that is, $|x| = x$ if $x \geq 0$ and $|x| = -x$ if $x < 0$.

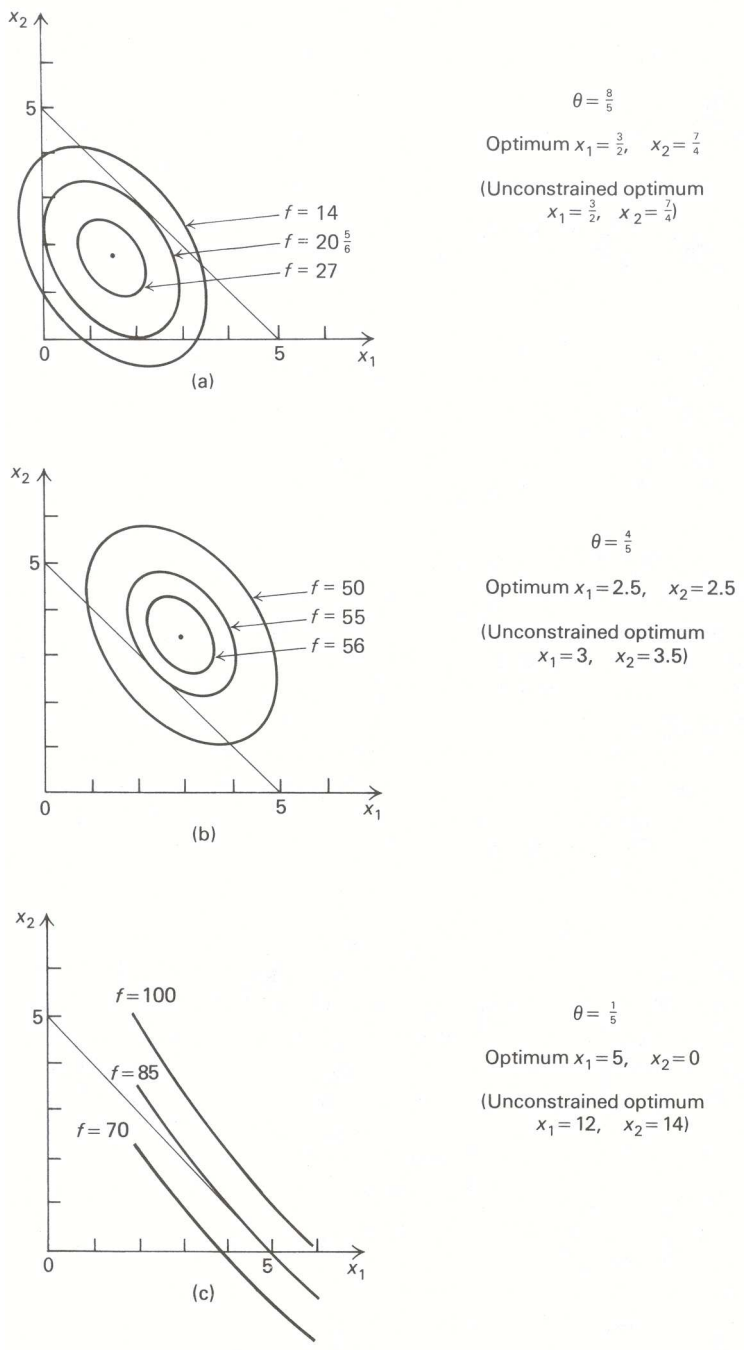

Figure 13.1 Portfolio-selection example for various values of θ. (Lines are contours of constant objective values.)

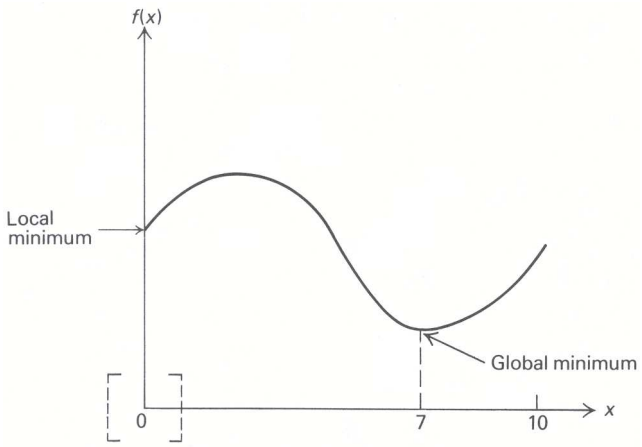

Figure 13.2 Local and global minima.

2. A *local maximum* if $f(x) \geq f(y)$ for every feasible point $y = (y_1, y_2, \ldots, y_n)$ sufficiently close to x. That is, if there is a number $\epsilon > 0$ (possibly quite small) so that, whenever each variable y_j is within ϵ of x_j — that is, $x_j - \epsilon \leq y_j \leq x_j + \epsilon$ —and y is feasible, then $f(x) \geq f(y)$.

Global and local minima are defined analogously. The definition of local maximum simply says that if we place an n-dimensional box (e.g., a cube in three dimensions) about x, whose side has length 2ϵ, then $f(x)$ is as small as $f(y)$ for every feasible point y lying within the box. (Equivalently, we can use n-dimensional spheres in this definition.) For instance, if $\epsilon = 1$ in the above example, the one-dimensional box, or interval, is pictured about the local minimum $x = 0$ in Fig. 13.2.

The concept of a local maximum is extremely important. As we shall see, most general-purpose nonlinear-programming procedures are near-sighted and can do no better than determine local maxima. We should point out that, since every global maximum is also a local maximum, the overall optimization problem can be viewed as seeking the best local maxima.

Under certain circumstances, local maxima and minima are known to be global. Whenever a function ''curves upward'' as in Fig. 13.3(a), a local minimum will be global. These functionsare called *convex*. Whenever a function ''curves downward'' as in Fig. 13.3(b) a local maximum will be a global maximum. These functions are called *concave*.† For this reason we usually wish to minimize convex functions and maximize concave functions. These observations are formalized below.

13.3 CONVEX AND CONCAVE FUNCTIONS

Because of both their pivotal role in model formulation and their convenient mathematical properties, certain functional forms predominate in mathematical programming. Linear functions are by far the most important. Next in importance are functions which are convex or concave. These functions are so central to the theory that we take some time here to introduce a few of their basic properties.

An essential assumption in a linear-programming model for profit maximization is constant returns to scale for each activity. This assumption implies that if the level of one activity doubles, then that activity's profit contribution also doubles; if the first activity level changes from x_1 to $2x_1$, then profit increases proportionally from say \$20 to \$40 [i.e., from c_1x_1 to $c_1(2x_1)$]. In many instances, it is realistic to assume constant returns to scale over the range of the data. At other times, though, due to economies of scale, profit might increase disproportionately, to say \$45; or, due to diseconomies of scale (saturation effects), profit may be only \$35. In the former case, marginal returns are increasing with the activity level, and we say that the profit function

† As a mnemonic, the ''A'' in concAve reflects the shape of these functions.

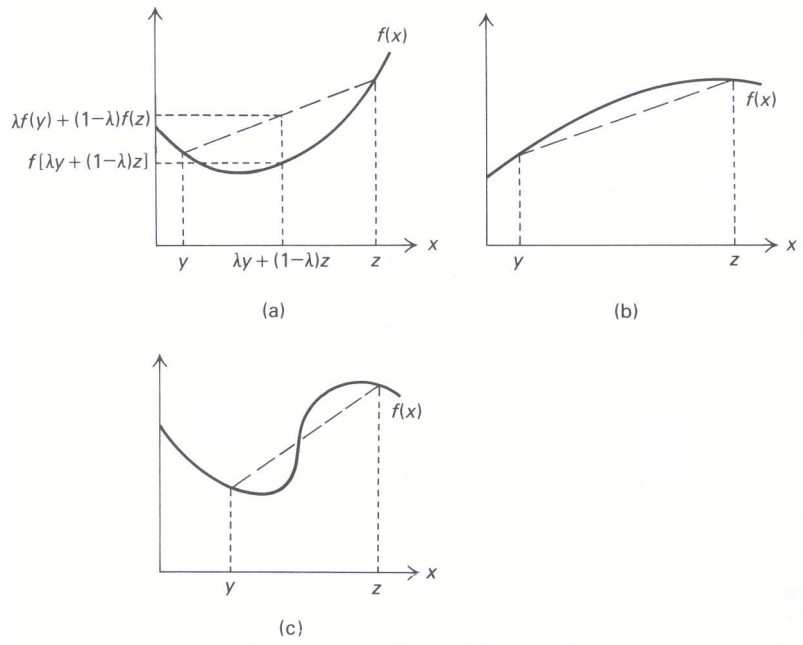

Figure 13.3 a) Convex function b) concave function (c) nonconvex, nonconcave function.

is *convex* (Fig. 13.3(a)). In the second case, marginal returns are decreasing with the activity level and we say that the profit function is *concave* (Fig.13.3(b)). Of course, marginal returns may increase over parts of the data range and decrease elsewhere, giving functions that are neither convex nor concave (Fig. 13.3(c)).

An alternative way to view a convex function is to note that linear interpolation overestimates its values. That is, for any points y and z, the line segment joining $f(y)$ and $f(z)$ lies above the function (see Fig. 13.3). More intuitively, convex functions are ''bathtub like'' and hold water. Algebraically,

Definition. A function $f(x)$ is called *convex* if,for every y and z and every $0 \leq \lambda \leq 1$,

$$f[\lambda y + (1-\lambda)z] \leq \lambda f(y) + (1-\lambda) f(z).$$

It is called *strictly convex* if, for every two distinct points y and z and every $0 < \lambda < 1$,

$$f[\lambda y + (1-\lambda)z] < \lambda f(y) + (1-\lambda) f(z).$$

The lefthand side in this definition is the function evaluation on the line joining x and y; the righthand side is the linear interpolation. Strict convexity corresponds to profit functions whose marginal returns are strictly increasing.

Note that although we have pictured f above to be a function of one decision variable, this is not a restriction. If $y = (y_1, y_2, \ldots, y_n)$ and $z = (z_1, z_2, \ldots, z_n)$, we must interpret $\lambda y + (1-\lambda)z$ only as weighting the decision variables one at a time, i.e., as the decision vector $(\lambda y_1 + (1-\lambda)z_1, \ldots, \lambda y_n + (1-\lambda)z_n)$.

Concave functions are simply the negative of convex functions. In this case, linear interpolation underestimates the function. The definition above is altered by reversing the direction of the inequality. Strict concavity is defined analogously. Formally,

Definition. A function $f(x)$ is called *concave* if,for every y and z and every $0 \leq \lambda \leq 1$,

$$f[\lambda y + (1-\lambda)z] \geq \lambda f(y) + (1-\lambda) f(z).$$

It is called *strictly concave* if, for every y and z and every $0 < \lambda < 1$,

$$f[\lambda y + (1-\lambda)z] > \lambda f(y) + (1-\lambda) f(z).$$

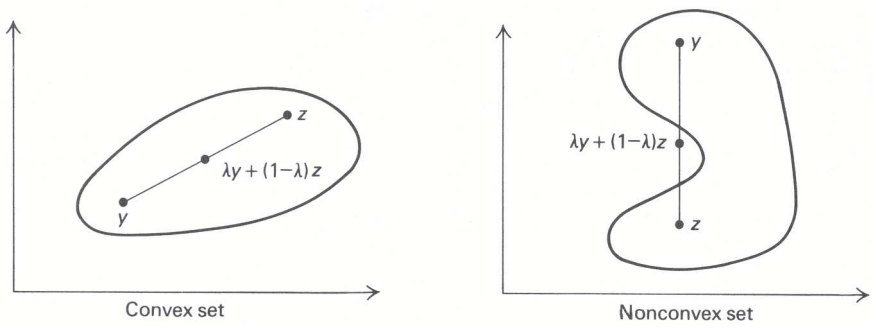

Figure 13.4 Convex and nonconvex sets.

We can easily show that a linear function is both convex and concave. Consider the linear function:

$$f(x) = \sum_{j=1}^{n} c_j x_j,$$

and let $0 \leq \lambda \leq 1$. Then

$$\begin{aligned} f(\lambda y + (1-\lambda)z) &= \sum_{j=1}^{n} c_j(\lambda y_j + (1-\lambda)z_j) \\ &= \lambda \left[\sum_{j=1}^{n} c_j y_j\right] + (1-\lambda)\left[\sum_{j=1}^{n} c_j z_j\right] \\ &= \lambda f(y) + (1-\lambda) f(z). \end{aligned}$$

These manipulations state, quite naturally, that linear interpolation gives exact values for f and consequently, from the definitions, that a linear function is both convex and concave. This property is essential, permitting us to either maximize or minimize linear functions by computationally attractive methods such as the simplex method for linear programming.

Other examples of convex functions are x^2, x^4, e^x, e^{-x} or $-\log x$. Multiplying each example by minus one gives aconcave function. The definition of convexity implies that the sum of convex functions is convex and that any nonnegative multiple of a convex function also is convex. Utilizing this fact, we can obtain a large number of convex functions met frequently in practice by combining these simple examples, giving, for instance,

$$2x^2 + e^x, \qquad e^x + 4x,$$

or

$$-3 \log x + x^4.$$

Similarly, we can easily write several concave functions by multiplying these examples by minus one.

A notion intimately related to convex and concave functions is that of a *convex set*. These sets are ''fat,'' in the sense that, whenever y and z are contained in the set, every point on the line segment joining these points is also in the set (see Fig. 13.4). Formally,

> ***Definition.*** A set of points C is called *convex* if, for all λ in the interval $0 \leq \lambda \leq 1$, $\lambda y + (1-\lambda)z$ is contained in C whenever x and y are contained in C.

Again we emphasize that y and z in this definition are decision vectors; in the example, each of these vectors has two components.

We have encountered convex sets frequently before, since the feasible region for a linear program is convex. In fact, the feasible region for a nonlinear program is convex if it is specified by less-than-or-equal-to equalities with convex functions. That is, if $f_i(x)$ for $i = 1, 2, \ldots, m$, are convex functions and if the points $x = y$ and $x = z$ satisfy the inequalities

$$f_i(x) \leq b_i \quad (i = 1, 2, \ldots, m),$$

then, for any $0 \leq \lambda \leq 1$, $\lambda y + (1 - \lambda)z$ is feasible also, since the inequalities

$$\begin{array}{ccccc} f_i(\lambda y + (1-\lambda)z) & \leq & \lambda f_i(y) + (1-\lambda) f_i(z) & \leq & \lambda b_i + (1-\lambda)b_i = b_i \\ \uparrow & & & & \\ \text{Convexity} & & \uparrow & & \\ & & \text{Feasibility of } y \text{ and } z & & \end{array}$$

hold for every constraint. Similarly, if the constraints are specified by greater-than-or-equal-to inequalities and the functions are concave, then the feasible region is convex. In sum, for convex feasible regions we want convex functions for less-than-or-equal-to constraints and concave functions for greater-than-or-equal-to constraints. Since linear functions are both convex and concave, they may be treated as equalities.

An elegant mathematical theory, which is beyond the scope of this chapter, has been developed for convex and concave functions and convex sets. Possibly the most important property for the purposes of nonlinear programming was previewed in the previous section. Under appropriate assumptions, a local optimumcan be shown to be a global optimum.

Local Minimum and Local Maximum Property

1. A local $\begin{Bmatrix}\text{minimum}\\ \text{maximum}\end{Bmatrix}$ of a $\begin{Bmatrix}\text{convex}\\ \text{concave}\end{Bmatrix}$ function on a convex feasible region is also a global $\begin{Bmatrix}\text{minimum}\\ \text{maximum}\end{Bmatrix}$.
2. A local $\begin{Bmatrix}\text{minimum}\\ \text{maximum}\end{Bmatrix}$ of a strictly $\begin{Bmatrix}\text{convex}\\ \text{concave}\end{Bmatrix}$ function on a convex feasible region is the unique global $\begin{Bmatrix}\text{minimum}\\ \text{maximum}\end{Bmatrix}$.

We can establish this property easily by reference to Fig. 13.5. The argument is for convex functions; the concave case is handled similarly. Suppose that y is a local minimum. If y is not a global minimum, then, by definition, there is a feasible point z with $f(z) < f(y)$. But then if f is convex, the function must lie on or below the dashed linear interpolation line. Thus, in any box about y, there must be an x on the line segment joining y and z, with $f(x) < f(y)$. Since the feasible region is convex, this x is feasible and we have contradicted the hypothesis that y is a local minimum. Consequently, no such point z can exist and any local minimum such as y must be a global minimum.

To see the second assertion, suppose that y is a local minimum. By Property 1 it is also a global minimum. If there is another global minimum z (so that $f(z) = f(y)$), then $\frac{1}{2}x + \frac{1}{2}z$ is feasible and, by the definition of strict convexity,

$$f\left(\tfrac{1}{2}x + \tfrac{1}{2}z\right) < \tfrac{1}{2} f(y) + \tfrac{1}{2} f(z) = f(y).$$

But this states that $\frac{1}{2}x + \frac{1}{2}z$ is preferred to y, contradicting our premise that y is a global minimum. Consequently, no other global minimum such as z can possibly exist; that is, y must be the unique global minimum.

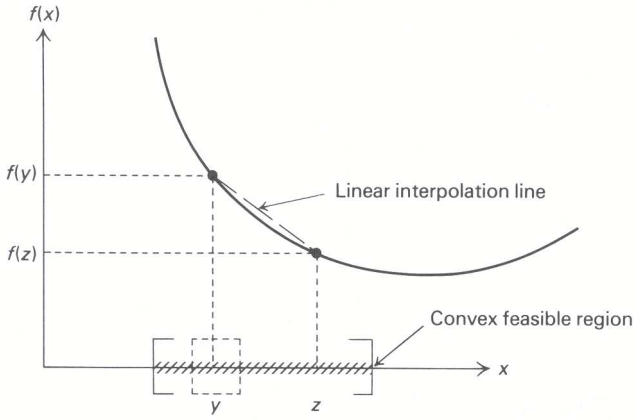

Figure 13.5 Local minima are global minima for convex function

13.4 PROBLEM CLASSIFICATION

Many of the nonlinear-programming solution procedures that have been developed do not solve the general problem

$$\text{Maximize } f(x),$$

subject to:

$$g_i(x) \leq b_i \qquad (i = 1, 2, \ldots, m),$$

but rather some special case. For reference, let us list some of these special cases:

1. Unconstrained optimization:
$$f \text{ general}, \quad m = 0 \text{ (noconstraints)}.$$

2. Linear programming:
$$f(x) = \sum_{j=1}^{n} c_j x_j, \quad g_i(x) = \sum_{j=1}^{n} a_{ij} x_j \quad (i = 1, 2, \ldots, m),$$
$$g_{m+i}(x) = -x_i \qquad (i = 1, 2, \ldots, n).$$

3. Quadratic programming:
$$f(x) = \sum_{j=1}^{n} c_j x_j + \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij} x_i x_j \qquad \text{(Constraints of case 2),}$$
(q_{ij} are given constants).

4. Linear constrained problem:
$$f(x) \text{ general}, \qquad g_i(x) = \sum_{j=1}^{n} a_{ij} x_j \qquad (i = 1, 2, \ldots, m),$$
(Possibly $x_j \geq 0$ will be included as well).

5. Separable programming:

$$f(x) = \sum_{j=1}^{n} f_j(x_j), \qquad g_i(x) = \sum_{j=1}^{n} g_{ij}(x_j) \qquad (i = 1, 2, \ldots, m);$$

   i.e., the problem ''separates'' into functions of single variables. The functions f_j and g_{ij} are given.

6. Convex programming:

   f is a concave function. (In a minimization problem, f would be a convex function.) The functions $g_i (i = 1, 2, \ldots, m)$ are all convex.

Note that cases 2, 3, and 4 are successive generalizations. In fact linear programming is a special case of every other problem type except for case 1.

13.5 SEPARABLE PROGRAMMING

Our first solution procedure is for separable programs, which are optimization problems of the form:

$$\text{Maximize } \sum_{j=1}^{n} f_j(x_j),$$

subject to:

$$\sum_{j=1}^{n} g_{ij}(x_j) \leq 0 \qquad (i = 1, 2, \ldots, m),$$

where each of the functions f_j and g_{ij} is known. These problems are called separable because the decision variables appear separately, one in each function g_{ij} in the constraints and one in each function f_j in the objective function.

Separable problems arise frequently in practice, particularly for time-dependent optimization. In this case, the variable x_j usually corresponds to an activity level for time period j and the separable model assumes that the decisions affecting resource utilization and profit (or cost) are additive over time. The model also arises when optimizing over distinct geographical regions, an example being the water-resources planning formulation given in Section 13.1.

Actually, instead of solving the problem directly, we make an appropriate approximation so that linear programming can be utilized. In practice, two types of approximations, called the *δ-method* and the *λ-method*, are often used. Since we have introduced the δ-method when discussing integer programming, we consider the λ-method in this section.

The general technique is motivated easily by solving a specific example. Consider the portfolio-selection problem introduced in Section 13.1. Taking $\theta = 1$, that problem becomes:

$$\text{Maximize } f(x) = 20x_1 + 16x_2 - 2x_1^2 - x_2^2 - (x_1 + x_2)^2,$$

subject to:

$$x_1 + x_2 \leq 5,$$
$$x_1 \geq 0, \qquad x_2 \geq 0.$$

As stated, the problem is not separable, because of the term $(x_1 + x_2)^2$ in the objective function. Letting $x_3 = x_1 + x_2$, though, we can re-express it in separable form as:

$$\text{Maximize } f(x) = 20x_1 + 16x_2 - 2x_1^2 - x_2^2 - x_3^2,$$

subject to:

$$x_1 + x_2 \le 5,$$
$$x_1 + x_2 - x_3 = 0,$$
$$x_1 \ge 0, \qquad x_2 \ge 0, \qquad x_3 \ge 0.$$

The objective function is now written as $f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3)$, where

$$f_1(x_1) = 20x_1 - 2x_1^2,$$
$$f_2(x_2) = 16x_2 - x_2^2,$$

and

$$f_3(x_3) = -x_3^2.$$

Thus it is separable. Clearly, the linear constraints are also separable.

To form the approximation problem, we approximate each nonlinear term by a piecewise-linear curve, as pictured in Fig. 13.6. We have used three segments to approximate the function f_1 and two segments to approximate the functions f_2 and f_3. Note that the constraints imply that

$$x_1 \le 5, \qquad x_2 \le 5, \qquad \text{and} \qquad x_3 \le 5,$$

so that we need not extend the approximation beyond these bounds on the variables.

The dashed approximation curves for $f_1(x_1)$, $f_2(x_2)$, and $f_3(x_3)$ are determined by linear approximation between breakpoints. For example, if $1 \le x_1 \le 3$, then the approximation f_1^{a} for f_1 is given by weighting the function's values at $x_1 = 1$ and $x_1 = 3$; that is, as

$$f_1^{\mathrm{a}}(x_1) = 18\lambda_1 + 42\lambda_2,$$

where the nonnegative variables λ_1 and λ_2 express x_1 as a weighted combination of 1 and 3; thus,

$$x_1 = 1\lambda_1 + 3\lambda_2, \quad \lambda_1 + \lambda_2 = 1.$$

For instance, evaluating the approximation at $x_1 = 1.5$ gives

$$f_1^{\mathrm{a}}(1.5) = 18\left(\tfrac{3}{4}\right) + 42\left(\tfrac{1}{4}\right) = 24,$$

since

$$1.5 = 1\left(\tfrac{3}{4}\right) + 3\left(\tfrac{1}{4}\right).$$

The overall approximation curve $f_1^{\mathrm{a}}(x_1)$ for $f_1(x_1)$ is expressed as:

$$f_1^{\mathrm{a}}(x_1) = 0\lambda_0 + 18\lambda_1 + 42\lambda_2 + 50\lambda_3, \tag{1}$$

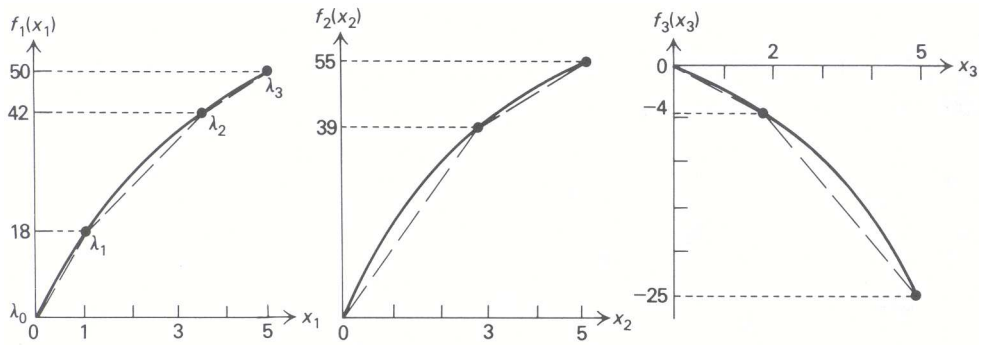

Figure 13.6 Approximating separable functions.

where

$$x_1 = 0\lambda_0 + 1\lambda_1 + 3\lambda_2 + 5\lambda_3, \tag{2}$$
$$\lambda_0 + \lambda_1 + \lambda_2 + \lambda_3 = 1,$$
$$\lambda_j \geq 0 \quad (j = 1, 2, 3, 4),$$

with the provision that the λ_j variables satisfy the following restriction:

Adjacency Condition. At most two λ_j weights arepositive. If two weights are positive, then they are adjacent, i.e., of the form λ_j and λ_{j+1}. A similar restriction applies to each approximation.

Figure 13.7 illustrates the need for the adjacency condition. If the weights $\lambda_0 = \frac{1}{3}$ and $\lambda_2 = \frac{2}{3}$, then the approximation (1) gives

$$f_1^{\mathrm{a}}(x_1) = 0\left(\tfrac{1}{3}\right) + 42\left(\tfrac{2}{3}\right) = 28,$$
$$x_1 = 0\left(\tfrac{1}{3}\right) + 3\left(\tfrac{2}{3}\right) = 2,$$

as shown in Fig. 13.7(a) by the light curve joining λ_0 and λ_2. In contrast, at $x_1 = 2$ the approximation curve gives $f_1^{\mathrm{a}}(2) = 30$.

An essential point to note here is that, for *concave* objective functions, the adjacency condition will always be enforced by the maximization and can be ignored. This property is easy to see geometrically by considering Fig 13.7(a). Suppose that the weights λ_0 and λ_2 are positive. By concavity, the function value of 18 at $x_1 = 1$ associated with the intermediate weight λ_1 lies above the line segment joining λ_0 and λ_2 in the figure. Consequently, the approximation curve must also lie above this line segment. The maximization, therefore, will select the dashed approximation curve with only the adjacent weights λ_0 and λ_1, or λ_1 and λ_2, positive, rather than any solution with both λ_0 and λ_2 positive. A similar argument applies if three or more weights are positive. For example, if λ_0, λ_2, and λ_3 are all positive, then the additional weight λ_3 can be viewed as weighting a point on the line segment joining λ_0 and λ_2 with the point at λ_3. Again, concavity implies that this point lies below the approximation curve and will not be accepted by the maximization. Note, however, that for the nonconcave function of Fig. 13.7(b), nonadjacent weights λ_0 and λ_2 are actually preferred to the approximation curve. Consequently, for nonconcave functions some effort must be expended to ensure that the adjacency condition is satisfied.

Returning to the portfolio-selection example, we can write the approximation problem:

$$\text{Maximize } z = f_1^{\mathrm{a}}(x_1) + f_2^{\mathrm{a}}(x_2) + f_3^{\mathrm{a}}(x_3),$$

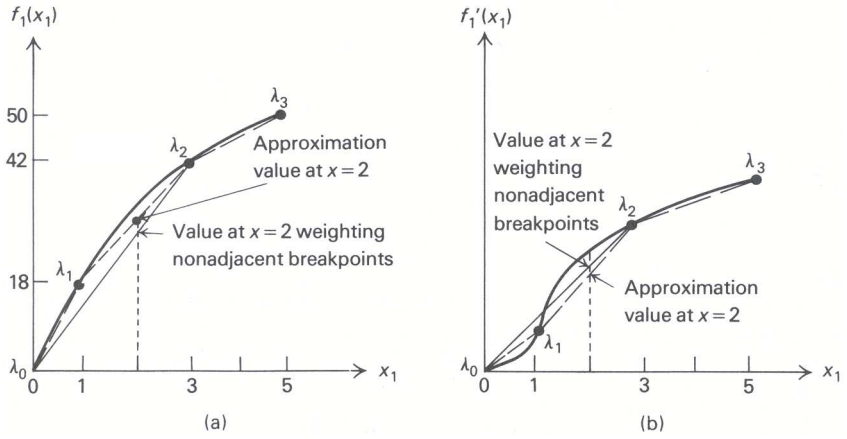

Figure 13.7 Need for the adjacency condition

subject to:

$$\begin{aligned} x_1 + \quad x_2 \qquad\quad &\le 5, \\ x_1 + \quad x_2 - \quad x_3 &= 0, \\ x_1 \ge 0, \;\; x_2 \ge 0, \quad x_3 &\ge 0, \end{aligned}$$

in terms of weighting variables λ_{ij}. Here we use the first subscript i to denote the weights to attach to variable i. The weights $\lambda_0, \lambda_1, \lambda_2$, and λ_3 used above for variable x_1 thus become $\lambda_{10}, \lambda_{11}, \lambda_{12}$, and λ_{13}. The formulation is:

$$\text{Maximize } z =$$

$$0\lambda_{10} + 18\lambda_{11} + 42\lambda_{12} + 50\lambda_{13} + 0\lambda_{20} + 39\lambda_{21} + 55\lambda_{22} - 0\lambda_{30} - 4\lambda_{31} - 25\lambda_{32},$$

subject to:

$$\begin{aligned} 0\lambda_{10} + 1\lambda_{11} + 3\lambda_{12} + 5\lambda_{13} + 0\lambda_{20} + 3\lambda_{21} + 5\lambda_{22} &\le 5, \\ 0\lambda_{10} + 1\lambda_{11} + 3\lambda_{12} + 5\lambda_{13} + 0\lambda_{20} + 3\lambda_{21} + 5\lambda_{22} - 0\lambda_{30} - 2\lambda_{31} - 5\lambda_{32} &= 0, \\ \lambda_{10} + \lambda_{11} + \lambda_{12} + \lambda_{13} &= 1 \\ \lambda_{20} + \lambda_{21} + \lambda_{22} &= 1 \\ \lambda_{30} + \lambda_{31} + \lambda_{32} &= 1 \end{aligned}$$

$$\lambda_{ij} \ge 0, \quad \text{for all } i \text{ and } j.$$

Since each of the functions $f_1(x_1)$, $f_2(x_2)$, and $f_3(x_3)$ is concave, the adjacency condition can be ignored and the problem can be solved as a *linear program*. Solving by the simplex method gives an optimal objective value of 44 with $\lambda_{11} = \lambda_{12} = 0.5$, $\lambda_{21} = 1$, and $\lambda_{32} = 1$ as the positive variables in the optimal solution. The corresponding values for the original problem variables are:

$$x_1 = (0.5)(1) + (0.5)(3) = 2, \qquad x_2 = 3, \qquad \text{and} \qquad x_3 = 5.$$

This solution should be contrasted with the true solution

$$x_1 = \tfrac{7}{3}, \qquad x_2 = \tfrac{8}{3}, \qquad x_3 = 5, \quad \text{and} \qquad f(x_1, x_2, x_3) = 46\tfrac{1}{3},$$

which we derive in Section 13.7.

Note that the approximation problem has added several λ variables and that one weighting constraint in (2) is associated with each x_j variable. Fortunately, these weighting constraints are of a special generalized upper-bounding type, which add little to computational effort and keep the number of effective constraints essentially unchanged. Thus, the technique can be applied to fairly large nonlinear programs, depending of course upon the capabilities of available linear-programming codes.

Once the approximation problem has been solved, we can obtain a better solution by introducing more breakpoints. Usually more breakpoints will be added near the optimal solution given by the original approximation.

Adding a single new breakpoint at $x_1 = 2$ leads to an improved approximation for this problem with a linear-programming objective value of 46 and

$$x_1 = 2, \qquad x_2 = 3, \qquad \text{and} \qquad x_3 = 5.$$

In this way, an approximate solution can be found as close as desired to the actual solution.

General Procedure

The general problem must be approached more carefully, since linear programming can give nonadjacent weights. The procedure is to express each variable* in terms of breakpoints, e.g., as above

$$x_1 = 0\lambda_{10} + 1\lambda_{11} + 3\lambda_{12} + 5\lambda_{13},$$

* Variables that appear in the model in only a linear fashion should not be approximated and remain as x_j variables.

and then use these breakpoints to approximate the objective function and each constraint, giving the approximation problem:

$$\text{Maximize } \sum_{j=1}^{n} f_j^{\mathrm{a}}(x_j),$$

$$\text{subject to :} \tag{3}$$

$$\sum_{j=1}^{n} g_{ij}^{\mathrm{a}}(x_j) \le b_i \qquad (i = 1, 2, \ldots, m).$$

If each original function $f_j(x_j)$ is concave and each $g_{ij}(x_j)$ convex,† then the λ_{ij} version is solved as a linear program. Otherwise, the simplex method is modified to enforce the adjacency condition. A natural approach is to apply the simplex method as usual, except for a modified rule that maintains the adjacency condition at each step. The alteration is:

> ***Restricted-Entry Criterion.*** Use the simplex criterion, butdo not introduce a λ_{ik} variable into the basis unless there is only one λ_{ij} variable currently in the basis and it is of the form $\lambda_{i,k-1}$ or $\lambda_{i,k+1}$, i.e.,it is adjacent to λ_{ik}.

Note that, when we use this rule, the optimal solution may contain a nonbasic variable λ_{ik} that would ordinarily be introduced into the basis by the simplex method (since its objective cost in the canonical form is positive), but is not introduced because of the restricted-entry criterion. If the simplex method would choose a variable to enter the basis that is unacceptable by the restricted-entry rule, then we choose the next best variable according to the greatest positive reduced cost.

An attractive feature of this procedure is that it can be obtained by making very minor modifications to any existing linear-programming computer code. As a consequence, most commercial linear-programming packages contain a variation of this separable-programming algorithm. However, the solution determined by this method in the general case can only be shown to be a localoptimum to the approximation problem (3).

Inducing Separability

Nonseparable problems frequently can be reduced to a separable form by a variety of formulation tricks. A number of such transformations are summarized in Table 13.1.

Table 13.1 Representative transformations

Term	*Substitution*	*Additional constraints*	*Restriction*
x_1x_2	$x_1x_2 = y_1^2 - y_2^2$	$y_1 = \frac{1}{2}(x_1 + x_2)$ $y_2 = \frac{1}{2}(x_1 - x_2)$	None
x_1x_2	$x_1x_2 = y_1$	$\log y_1 = \log x_1 + \log x_2$	$x_1 > 0, x_2 > 0$
$x_1^{x_2}$	$x_1^{x_2} = y_1$	$y_1 = 10^{y_2x_2}$* $x_1 = 10^{y_2}$	$x_1 > 0$
$2^{x_1+x_2^2}$	$2^{x_1+x_2^2} = y_1$	$\log y_1 = (\log 2)(x_1 + x_2^2)$	None

* The term y_2x_2 should now be separated by the first transformation, followed by an application of the last transformation to separate the resulting power-of-10 term.

† Because the constraints are written as ($\le$), the constraints should be convex; they should be concave for ($\ge$) inequalities. Similarly, for a minimization problem, the objective functions $f_j(x_j)$ should be convex.

To see the versatility of these transformations, suppose that the nonseparable term $x_1x_2^2/(1+x_3)$ appears either in the objective function or in a constraint of a problem. Letting $y_1 = 1/(1+x_3)$, the term becomes $x_1x_2^2y_1$. Now if $x_1 > 0, x_2^2 > 0$, and $y_1 > 0$ over the feasible region, thenletting $y_2 = x_1x_2^2y_1$, the original term is replaced by y_2 and the separable constraints

$$y_1 = \frac{1}{1 + x_3}$$

and

$$\log y_2 = \log x_1 + \log x_2^2 + \log y_1$$

are introduced. If the restrictions $x_1 > 0,\ x_2^2 > 0,\ y_1 > 0$ arenot met, we may let $y_2 = x_1x_2^2$, substitute y_1y_2 for the original term, and append the constraints:

$$y_1 = \frac{1}{1 + x_3}, \quad y_2 = x_1x_2^2.$$

The nonseparable terms y_1y_2 and $x_1x_2^2$ can now be separated using the first transformation in Table 13.1 (for the last expression, let x_2^2 replace x_2 in the table).

Using the techniques illustrated by this simple example, we may, in theory, state almost any optimization problem as a separable program. Computationally, though, the approach is limited since the number of added variables and constraints may make the resulting separable program too large to be manageable.

13.6 LINEAR APPROXIMATIONS OF NONLINEAR PROGRAMS

Algebraic procedures such as pivoting are so powerful for manipulating linear equalities and inequalities that many nonlinear-programming algorithms replace the given problem by an approximating linear problem. Separable programming is a prime example, and also one of the most useful, of these procedures. As in separable programming, these nonlinear algorithms usually solve several linear approximations by letting the solution of the last approximation suggest a new one.

By using different approximation schemes, this strategy can be implemented in several ways. This section introduces three of these methods, all structured to exploit the extraordinary computational capabilities of the simplex method and the wide-spread availability of its computer implementation.

There are two general schemes for approximating nonlinear programs. The last section used linear approximation for separable problems by weighting selected values of each function. This method is frequently referred to as *inner linearization* since, as shown in Fig. 13.8 when applied to a convex programming problem (i.e., constraints $g_i(x) \geq 0$ with g_i concave, or $g_i(x) \leq 0$ with g_i convex), the feasible region for the approximating problem lies inside that of the original problem. In contrast, other approximation schemes use slopes to approximate each function. These methods are commonly referred to as *outer linearizations* since, for convex-programming problems, the feasible region for the approximating problem encompasses that of the original problem. Both approaches are illustrated further in this section.

Frank–Wolfe Algorithm

Let $x^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ be any feasible solution to a nonlinear program with linear constraints:

$$\text{Maximize } f(x_1, x_2, \ldots, x_n),$$

subject to:

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i \qquad (i = 1, 2, \ldots, m),$$

$$x_j \geq 0 \qquad (j = 1, 2, \ldots, n).$$

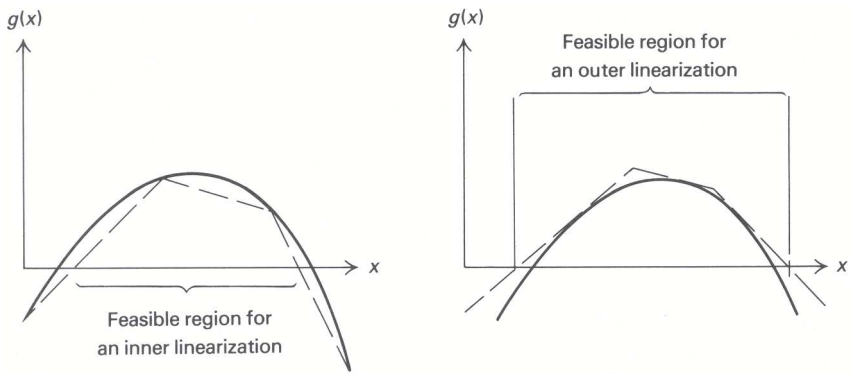

Figure 13.8 Inner and outer linearizations of $g(x) \geq 0$.

Here x^0 might be determined by phase I of the simplex method. This algorithm forms a linear approximation at the point x^0 by replacing the objective function with its current value plus a linear correction term; that is, by the linear objective

$$f(x^0) + \sum_{j=1}^{n} c_j(x_j - x_j^0),$$

where c_j is the slope, or partial derivative, of f with respect to x_j, evaluated at the point x^0. Since $f(x^0)$, c_j, and x_j^0 are fixed, maximizing this objective function is equivalent to maximizing:

$$z = \sum_{j=1}^{n} c_j x_j.$$

The linear approximation problem is solved, giving an optimal solution $y = (y_1, y_2, \ldots, y_n)$. At this point the algorithm recognizes that, although the linear approximation problem indicates that the objective improves steadily from x^0 to y, the *nonlinear* objective might not continue to improve from x^0 to y. Therefore, the algorithm uses a procedure to determine the maximum value for $f(x_1, x_2, \ldots, x_n)$ along the line segment joining x^0 to y. Special methods for performing optimization along the line segment are discussed in Section 13.9. For now, let us assume that there is a method to accomplish this line-segment maximization for us.

Letting $x^1 = (x_1^1, x_2^1, \ldots, x_n^1)$ denote the optimal solution of the line-segment optimization, we repeat the procedure by determining a new linear approximation to the objective function with slopes c_j evaluated at x^1. Continuing in this way, we determine a sequence of points $x^1, x^2, \ldots, x^n, \ldots$; any point $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ that these points approach in the limit is an optimal solution to the original problem.

Let us illustrate the method with the portfolio-selection example from Section 13.1:

$$\text{Maximize } f(x) = 20x_1 + 16x_2 - 2x_1^2 - x_2^2 - (x_1 + x_2)^2,$$

subject to:

$$\begin{aligned} x_1 + x_2 &\leq 5, \\ x_1 \geq 0, \qquad x_2 &\geq 0. \end{aligned}$$

The partial derivatives of the objective function at any point $x = (x_1, x_2)$ are given by:

$$\begin{aligned} c_1 &= 20 - 4x_1 - 2(x_1 + x_2) = 20 - 6x_1 - 2x_2, \\ c_2 &= 16 - 2x_2 - 2(x_1 + x_2) = 16 - 2x_1 - 4x_2. \end{aligned}$$

Suppose that the initial point is $x^0 = (0, 0)$. At this point $c_1 = 20$ and $c_2 = 16$ and the linear approximation uses the objective function $20x_1 + 16x_2$. The optimal solution to this linear program is $y_1 = 5$, $y_2 = 0$, and the line-segment optimization is made on the line joining $x^0 = (0, 0)$ to $y = (5, 0)$; that is, with $0 \le x_1 \le 5$, $x_2 = 0$. The optimal solution can be determined to be $x_1 = 3\frac{1}{3}$, $x_2 = 0$, so that the procedure is repeated from $x^1 = (3\frac{1}{3}, 0)$.

The partial derivatives are now $c_1 = 0, c_2 = 9\frac{1}{3}$, and the solution to the resulting linear program of maximizing $0x_1 + 9\frac{1}{3}x_2$ is $y_1 = 0$, $y_2 = 5$. The line segment joining x^1 and y is given by

$$x_1 = \theta y_1 + (1 - \theta)x_1^1 = 3\tfrac{1}{3}(1 - \theta),$$
$$x_2 = \theta y_2 + (1 - \theta)x_2^1 = 5\theta,$$

as θ varies between 0 and 1. The optimal value of θ over the line segment is $\theta = \frac{7}{15}$, so that the next point is:

$$x_1^2 = \left(\tfrac{10}{3}\right)\left(\tfrac{8}{15}\right) = 1\tfrac{7}{9} \quad \text{and} \quad x_2^2 = 2\tfrac{1}{3}.$$

Figure 13.9 illustrates these two steps of the algorithm and indicates the next few points x^4, x^5, and x^6 that it generates.

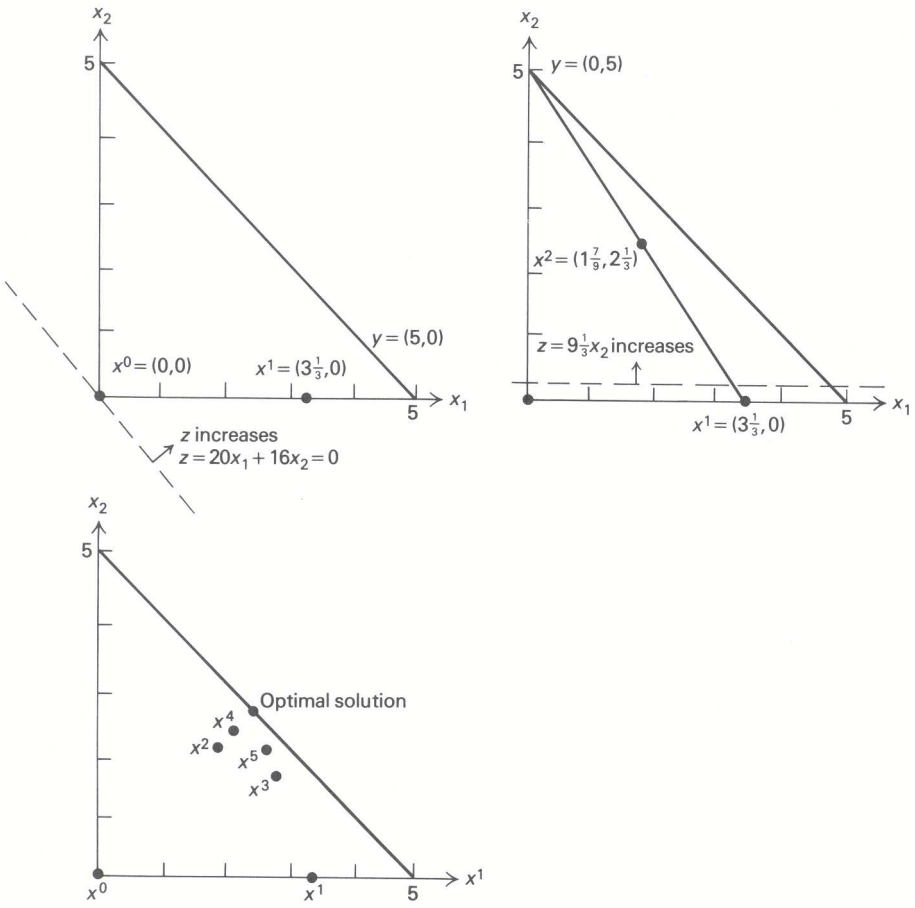

Figure 13.9 Example of the Frank–Wolfe algorithm.

The Frank–Wolfe algorithm is convenient computationally because it solves linear programs with the same constraints as the original problem. Consequently, any structural properties of these constraints are

available when solving the linear programs. In particular, network-flow techniques or large-scale system methods can be used whenever the constraints have the appropriate structure. Also, note from Fig. 13.9 that the points $x^1, x^2, x^3, \ldots$ oscillate. The even-numbered points $x^2, x^4, x^6, \ldots$ lie on one line directed toward the optimal solution, and the odd-numbered points $x^1, x^3, x^5, \ldots$, lie on another such line. This general tendency of the algorithm slows its convergence. One approach for exploiting this property to speed convergence is to make periodic optimizations along the line generated by every other point x^{k+2} and x^k for some values of k.

MAP (Method of Approximation Programming)

The Frank–Wolfe algorithm can be extended to general nonlinear programs by making linear approximations to the constraints as well as the objective function. When the constraints are highly nonlinear, however, the solution to the approximation problem can become far removed from feasible region since the algorithm permits large moves from any candidate solution. The Method of Approximation Programming (MAP) is a simple modification of this approach that limits the size of any move. As a result, it is sometimes referred to as a *small-step procedure*.

Let $x^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ be any candidate solution to the optimization problem:

$$\text{Maximize } f(x_1, x_2, \ldots, x_n),$$

subject to:

$$g_i(x_1, x_2, \ldots, x_n) \leq 0 \qquad (i = 1, 2, \ldots, m).$$

Each constraint can be linearized, using its current value $g_i(x^0)$ plus a linear correction term, as:

$$\hat{g}_i(x) = g_i(x^0) + \sum_{j=1}^{n} a_{ij}(x_j - x_j^0) \leq 0,$$

where a_{ij} is the partial derivative of constraint g_i with respect to variable x_j evaluated at the point x^0. This approximation is a linear inequality, which can be written as:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i^0 \equiv \sum_{j=1}^{n} a_{ij} x_j^0 - g_i(x^0),$$

since the terms on the righthand side are all constants.

The MAP algorithm uses these approximations, together with the linear objective-function approximation, and solves the linear-programming problem:

$$\text{Maximize } z = \sum_{j=1}^{n} c_j x_j,$$

subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i^0 \qquad (i = 1, 2, \ldots, m), \tag{4}$$

$$x_j^0 - \delta_j \leq x_j \leq x_j^0 + \delta_j \quad (j = 1, 2, \ldots, n).$$

The last constraints restrict the step size; they specify that the value for x_j can vary from x_j^0 by no more than the user-specified constant δ_j.

When the parameters δ_j are selected to be small, the solution to this linear program is not far removal from x^0. We might expect then that the additional work required by the line-segment optimization of the

Frank–Wolfe algorithm is not worth the slightly improved solution that it provides. MAP operates on this premise, taking the solution to the linear program (4) as the new point x^1. The partial-derivative data a_{ij}, b_i, and c_j is recalculated at x^1, and the procedure is repeated. Continuing in this manner determines points $x^1, x^2, \ldots, x^k, \ldots$ and as in the Frank–Wolfe procedure, any point $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ that these points approach in the limit is considered a solution.

Steps of the MAP Algorithm

STEP (0): Let $x^0 = (x_1^0, x_2^0, \ldots, x_n^0)$ be any candidate solution, usually selected to be feasible or near-feasible. Set $k = 0$.

STEP (1): Calculate c_j and $a_{ij} (i = 1, 2, \ldots, m)$, the partial derivatives of the objective function and constraints evaluated at $x^k = (x_1^k, x_2^k, \ldots, x_n^k)$. Let $b_i^k = a_{ij} x^k - g_i(x^k)$.

STEP (2): Solve the linear-approximation problem (4) with b_i^k and x_j^k replacing b_i^0 and x_j^0, respectively. Let $x^{k+1} = (x_1^{k+1}, x_2^{k+1}, \ldots, x_n^{k+1})$ be its optimal solution. Increment k to $k + 1$ and return to STEP 1.

Since many of the constraints in the linear approximation merely specify upper and lower bounds on the decision variables x_j, the bounded-variable version of the simplex method is employed in its solution. Also, usually the constants δ_j are reduced as the algorithm proceeds. There are many ways to implement this idea. One method used frequently in practice, is to reduce each δ_j by between 30 and 50 percent at each iteration.

To illustrate the MAP algorithm, consider the problem:

$$\text{Maximize } f(x) = [(x_1 - 1)^2 + x_2^2],$$

subject to:

$$\begin{aligned} g_1(x) &= \quad x_1^2 + 6x_2 - \ 36 \le 0, \\ g_2(x) &= -4x_1 + \ x_2^2 - 2x_2 \le 0, \end{aligned}$$

$$x_1 \ge 0, \quad x_2 \ge 0.$$

The partial derivatives evaluated at the point $x = (x_1, x_2)$ are given by:

$$\begin{aligned} c_1 &= 2x_1 - 2, & c_2 &= 2x_2, \\ a_{11} &= 2x_1, & a_{12} &= 6, \\ a_{21} &= -4, & a_{22} &= 2x_2 - 2. \end{aligned}$$

Since linear approximations of any linear function gives that function again, no data needs to be calculated for the linear constraints $x_1 \ge 0$ and $x_2 \ge 0$.

Using these relationships and initiating the procedure at $x^0 = (0, 2)$ with $\delta_1 = \delta_2 = 2$ gives the linear-approximation problem:

$$\text{Maximize } z = -2x_1 + 4x_2,$$

subject to:

$$\begin{aligned} 0x_1 + 6x_2 &\le \quad 0(0) + 6(2) - (-24) = 36, \\ -4x_1 + 2x_2 &\le -4(0) + 2(2) - \quad 0 \quad = 4. \end{aligned}$$

$$-2 \le x_1 \le 2, \qquad 0 \le x_2 \le 4.$$

The righthand sides are determined as above by evaluating $a_{i1}x_1^0 + a_{i2}x_2^0 - g_i(x^0)$.

The feasible region and this linear approximation are depicted in Fig. 13.10. Geometrically, we see that the optimalsolution occurs at $x_1^1 = 1, x_2^1 = 4$. Using this point and reducing both δ_1 and δ_2 to 1 generates the new approximation:

$$\text{Maximize } z = 0x_1 + 8x_2,$$

subject to:

$$\begin{aligned} 2x_1 + 6x_2 &\leq \quad 2(1) + 6(4) - (-11) = 37, \\ -4x_1 + 6x_2 &\leq -4(1) + 6(4) - \quad (4) = 16, \\ 0 \leq x_1 &\leq 2, \qquad 3 \leq x_2 \leq 5. \end{aligned}$$

The solution indicated in Fig. 13.10 occurs at $x_1^2 = 2, \quad x_2^2 = 4$.

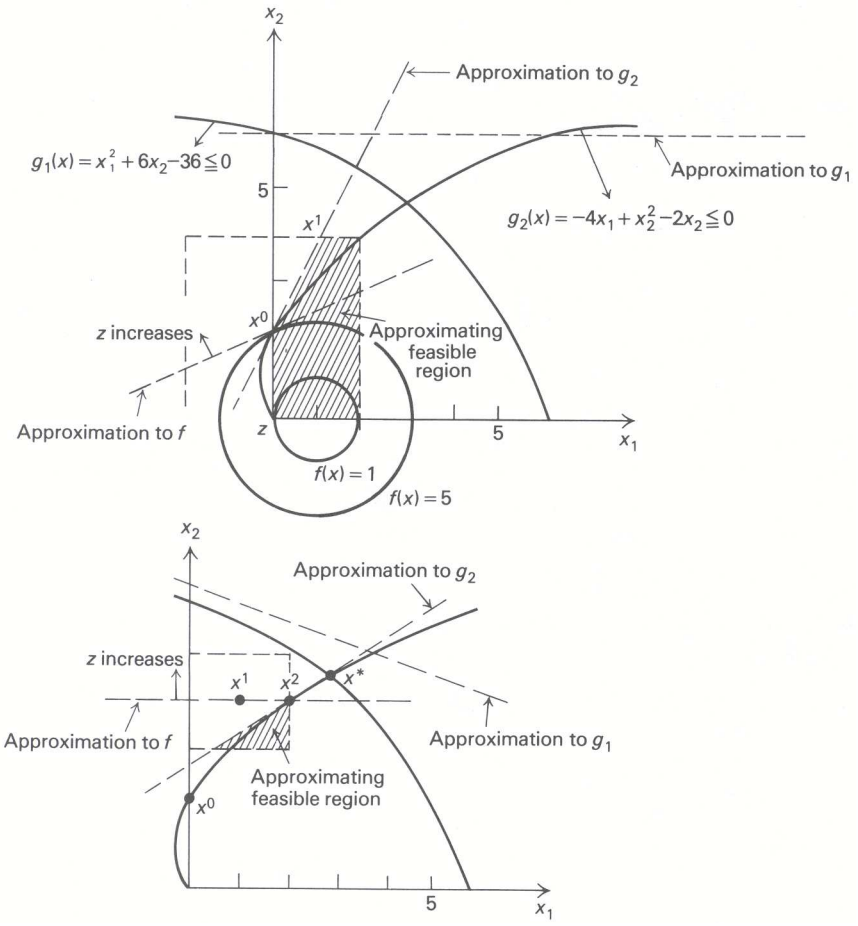

Figure 13.10 Example of the MAP algorithm.

If the procedure is continued, the points $x^3, x^4, \ldots$ that it generates approach the optimal solution x^* shown in Fig. 13.10. As a final note, let us observe that the solution x^1 is not feasible for the linear program that was constructed by making linear approximations at x^1. Thus, in general, both Phase I and Phase II of the simplex method may be required to solve each linear-programming approximation.

Generalized Programming

This algorithm is applied to the optimization problem of selecting decision variables $x_1, x_2, \ldots, x_n$ from a region C to:

$$\text{Maximize } f(x_1, x_2, \ldots, x_n),$$

subject to:

$$g_i(x_1, x_2, \ldots, x_n) \leq 0 \qquad (i = 1, 2, \ldots, m).$$

The procedure uses inner linearization and extends the decomposition and column-generation algorithms introduced in Chapter 12. When the objective function and constraints g_i are linear and C consists of the feasible points for a linear-programming subproblem, generalized programming reduces to the decomposition method.

As in decomposition, the algorithm starts with k candidate solutions $x^j = (x_1^j, x_2^j, \ldots, x_n^j)$ for $j = 1, 2, \ldots, k$, all lying the region C. Weighting these points by $\lambda_1, \lambda_2, \ldots, \lambda_k$ generates the candidate solution:

$$x_i = \lambda_1 x_i^1 + \lambda_2 x_i^2 + \cdots + \lambda_k x_i^k \quad (i = 1, 2, \ldots, n). \tag{5}$$

Any choices can be made for the weights as long as they are nonnegative and sum to one. The ''best'' choice is determined by solving the linear-approximation problem in the variables $\lambda_1, \lambda_2, \ldots, \lambda_k$:

$$\begin{array}{lll}
\text{Maximize } \lambda_1 f(x^1) + \lambda_2 f(x^2) + \cdots + \lambda_k f(x^k), & & \textit{Optimal shadow prices} \\
\text{subject to:} & & \\
\lambda_1 g_1(x^1) + \lambda_2 g_1(x^2) + \cdots + \lambda_k g_1(x^k) \leq 0, & & y_1 \\
\quad\vdots & \quad\vdots \quad\vdots & \quad\vdots \\
\lambda_1 g_m(x^1) + \lambda_2 g_m(x^2) + \cdots + \lambda_k g_m(x^k) \leq 0, & & y_m \\
\lambda_1 \quad + \lambda_2 \quad + \quad \lambda_k = 1. & & \sigma \\
\lambda_j \geq 0 \quad (j = 1, 2, \ldots, k). & &
\end{array} \tag{6}$$

The coefficients $f(x^j)$ and $g_i(x^j)$ of the weights λ_j in this linear program are fixed by our choice of the candidate solutions $x^1, x^2, \ldots, x^k$. In this problem, the original objective function and constraints have been replaced by linear approximations. When x is determined from expression (5) by weighting the points $x^1, x^2, \ldots, x^k$ by the solution of problem (6), the linear approximations at x are given by applying the same weights to the objective function and constraints evaluated at these points; that is,

$$f^a(x) = \lambda_1 f(x^1) + \lambda_2 f(x^2) + \cdots + \lambda_k f(x^k)$$

and

$$g_i^a(x) = \lambda_1 g_i(x^1) + \lambda_2 g_i(x^2) + \cdots + \lambda_k g_i(x^k).$$

The approximation is refined by applying column generation to this linear program. In this case, a new column with coefficients $f(x^{k+1}), g_1(x^{k+1}), \ldots, g_m(x^{k+1})$ is determined by the pricing-out operation:

$$v = \text{Max } [f(x) - y_1 g_1(x) - y_2 g_2(x) - \cdots - y_m g_m(x)],$$

subject to:

$$x \in C.$$

This itself is a nonlinear programming problem but without the g_i constraints. If $v - \sigma \leq 0$, then no new column improves the linear program and the procedure terminates. If $v - \sigma > 0$, then the solution x^{k+1}

giving v determines a new column to be added to the linear program (6) and the procedure continues.

The optimal weights $\lambda_1^*, \lambda_2^*, \ldots, \lambda_k^*$ to the linear program provide a candidate solution

$$x_i^* = \lambda_1^* x_i^1 + \lambda_2^* x_i^2 + \cdots + \lambda_k^* x_i^k$$

to the original optimization problem. This candidate solution is most useful when C is a convex set and each constraint is convex, for then x_i^* is a weighted combination of points $x^1, x^2, \ldots, x^k$ in C and thus belongs to C; and, since linear interpolation overestimates convex functions,

$$g_i(x^*) \leq \lambda_1^* g_i(x^1) + \lambda_2^* g_i(x^2) + \cdots + \lambda_k^* g_i(x^k).$$

The righthand side is less than or equal to zero by the linear-programming constraints; hence, $g_i(x^*) \leq 0$ and x^* is a feasible solution to the original problem.

As an illustration of the method, consider the nonlinear program:

$$\text{Maximize } f(x) = x_1 - (x_2 - 5)^2 + 9,$$

subject to:

$$g(x) = x_1^2 + x_2^2 - 16 \leq 0,$$
$$x_1 \geq 0, \quad x_2 \geq 0.$$

We let C be the region with $x_1 \geq 0$ and $x_2 \geq 0$, and start with the three points $x^1 = (0, 0)$, $x^2 = (5, 0)$, and $x^3 = (0, 5)$ from C. The resulting linear approximation problem:

	Optimal shadow prices
Maximize $z = -16\lambda_1 - 11\lambda_2 + 9\lambda_3$, subject to:	
$-16\lambda_1 + 9\lambda_2 + 9\lambda_3 \leq 0$,	$y = 1$
$\lambda_1 + \lambda_2 + \lambda_3 = 1$,	$\sigma = 0$

$$\lambda_1 \geq 0, \quad \lambda_2 \geq 0, \quad \lambda_2 \geq 0,$$

is sketched in Fig. 13.11, together with the original problem. The feasible region for the approximation problem is given by plotting the points defined by (5) that correspond to feasible weights in this linear program.

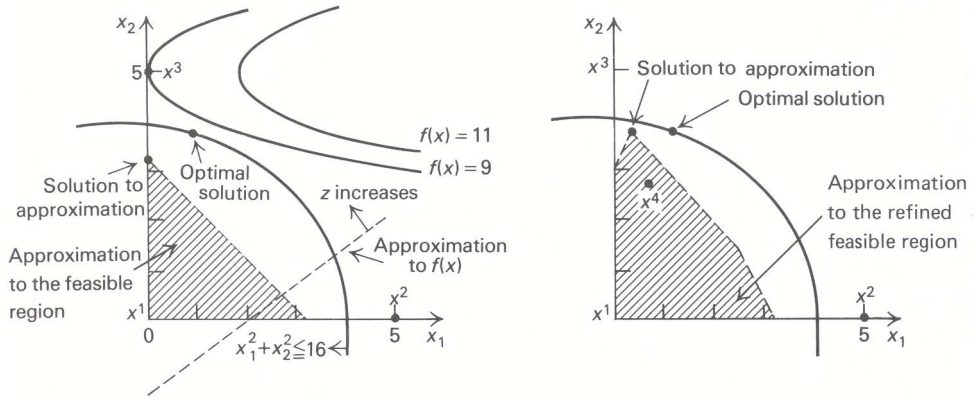

Figure 13.11 An example of generalized programming.

The solution to the linear program is:

$$\lambda_1^* = 0.36, \qquad \lambda_2^* = 0, \qquad \text{and} \qquad \lambda_3^* = 0.64,$$

or

$$x^* = 0.36(0, 0) + 0.64(0, 5) = (0, 3.2).$$

The nonlinear programming subproblem is:

$$\text{Maximize } [f(x) - yg(x)] = x_1 - (x_2 - 5)^2 - yx_1^2 - yx_2^2 + 16y + 9,$$

subject to:

$$x_1 \geq 0, \qquad x_2 \geq 0.$$

For $y > 0$ the solution to this problem can be shown to be $x_1 = 1/(2y)$ and $x_2 = 5/(1 + y)$, by setting the partial derivatives of $f(x) - yg(x)$, with respect to x_1 and x_2, equal to zero.

In particular, the optimal shadow price $y = 1$ gives the new point $x^4 = \left(\frac{1}{2}, 2\frac{1}{2}\right)$. Since $f(x^4) = 3\frac{1}{4}$ and $g(x^4) = -9\frac{1}{2}$, the updated linear programming approximation is:

Maximize $z = -16\lambda_1 - 11\lambda_2 + 9\lambda_3 + 3\frac{1}{4}\lambda_4$, *Optimal shadow prices*

subject to:

$$-16\lambda_1 + 9\lambda_2 + 9\lambda_3 - 9\tfrac{1}{2}\lambda_4 \leq 0, \qquad y = \tfrac{23}{74}$$

$$\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1, \qquad \sigma = \tfrac{459}{74}$$

$$\lambda_j \geq 0 \quad (j = (1, 2, 3, 4).$$

The optimal solution has λ_3 and λ_4 basic with $\lambda_3^* = \frac{19}{37}$, $\lambda_4^* = \frac{18}{37}$. The corresponding value for x is:

$$\begin{aligned} x^* &= \lambda_3^*(0, 5) + \lambda_4^* \left(\tfrac{1}{2}, 2\tfrac{1}{2}\right) \\ &= \tfrac{19}{37}(0, 5) + \tfrac{18}{37}\left(\tfrac{1}{2}, 2\tfrac{1}{2}\right) = \left(\tfrac{9}{37}, 3\tfrac{29}{37}\right). \end{aligned}$$

As we continue in this way, the approximate solutions will approach the optimal solution $x_1 = 1.460$ and $x_2 = 3.724$ to the original problem.

We should emphasize that the generalized programming is unlike decomposition for linear programs in that it does not necessarily determine the optimal solution in a finite number of steps. This is true because nonlinearity does not permit a finite number of extreme points to completely characterize solutions to the subproblem.

13.7 QUADRATIC PROGRAMMING

Quadratic programming concerns the maximization of a quadratic objective function subject to linear constraints, i.e., the problem:

$$\text{Maximize } f(x) = \sum_{j=1}^{n} c_j x_j + \tfrac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} q_{jk} x_j x_k,$$

subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \qquad (i = 1, 2, \ldots, m),$$

$$x_j \geq 0 \qquad (j = 1, 2, \ldots, n).$$

The data c_j, a_{ij}, and b_i are assumed to be known, as are the additional q_{jk} data. We also assume the symmetry condition $q_{jk} = q_{kj}$. This condition is really no restriction, since q_{jk} can be replaced by $\frac{1}{2}(q_{jk} + q_{kj})$. The symmetry condition is then met, and a straightforward calculation shows that the old and new q_{jk} coefficients give the same quadratic contribution to the objective function. The factor $\frac{1}{2}$ has been included to

simplify later calculations; if desired, it can be incorporated into the q_{jk} coefficients. Note that, if every $q_{jk} = 0$, then the problem reduces to a linear program.

The first and third examples of Section 13.1 show that the quadratic-programming model arises in constrained regression and portfolio selection. Additionally, the model is frequently applied to approximate problems of maximizing general functions subject to linear constraints.

In Chapter 4 we developed the optimality conditions for linear programming. Now we will indicate the analogous results for quadratic programming. The motivating idea is to note that, for a linear objective function

$$f(x) = \sum_{j=1}^{n} c_j x_j,$$

the derivative (i.e., the slope or marginal return) of f with respect to x_j is given by c_j, whereas, for a quadratic program, the slope at a point is given by

$$c_j + \sum_{k=1}^{n} q_{jk} x_k.$$

The quadratic optimality conditions are then stated by replacing every c_j in the linear-programming optimality conditions by $c_j + \sum_{k=1}^{n} q_{jk} x_k$. (See Tableau 1.)

Tableau 1 Optimality conditions

	Linear program	*Quadratic program*
Primal feasibility	$\sum_{j=1}^{n} a_{ij} x_j \le b_i,$ $x_j \ge 0$	$\sum_{j=1}^{n} a_{ij} x_j \le b_i,$ $x_j \ge 0$
Dual feasibility	$\sum_{i=1}^{m} y_i a_{ij} \ge c_j,$ $y_i \ge 0$	$\sum_{i=1}^{m} y_i a_{ij} \ge c_j + \sum_{k=1}^{n} q_{jk} x_k,$ $y_i \ge 0$
Complementary slackness	$y_i \left[b_i - \sum_{j=1}^{n} a_{ij} x_j \right] = 0,$ $\left[\sum_{i=1}^{m} y_i a_{ij} - c_j \right] x_j = 0$	$y_i \left[b_i - \sum_{j=1}^{n} a_{ij} x_j \right] = 0,$ $\left[\sum_{i=1}^{m} y_i a_{ij} - c_j - \sum_{k=1}^{n} q_{jk} x_k \right] x_j = 0$

Note that the primal and dual feasibility conditions for the quadratic program are linear inequalities in nonnegative variables x_j and y_i. As such, they can be solved by the Phase I simplex method. A simple modification to that method will permit the complementary-slackness conditions to be maintained as well. To discuss the modification, let us introduce slack variables s_i for the primal constraints and surplus variables v_j for the dual constraints; that is,

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

$$v_j = \sum_{i=1}^{m} y_i a_{ij} - c_j - \sum_{k=1}^{n} q_{jk} x_k.$$

Then the complementary slackness conditions become:

$$y_i s_i = 0 \qquad (i = 1, 2, \ldots, m),$$

and

$$v_j x_j = 0 \qquad (j = 1, 2, \ldots, n).$$

The variables y_i and s_i are called *complementary*, as are the variables v_j and x_j. With this notation, the technique is to solve the primal and dual conditions by the Phase I simplex method, but not to allow any complementary pair of variables to appear in the basis both at the same time. More formally, the Phase I simplex method is modified by the:

Restricted-Entry Rule. Never introduce avariable into the basis if its complementary variable is already a member of the basis, even if the usual simplex criterion says to introduce the variable.

Otherwise, the Phase I procedure is applied as usual. If the Phase I procedure would choose a variable to enter the basis that is unacceptable by the restricted-entry rule, then we choose the next best variable according to the greatest positive reduced cost in the Phase I objective function.

An example should illustrate the technique. Again, the portfolio-selection problem of Section 13.1 will be solved with $\theta = 1$; that is,

$$\text{Maximize } f(x) = 20x_1 + 16x_2 - 2x_1^2 - x_2^2 - (x_1 + x_2)^2,$$

subject to:

$$x_1 + x_2 \leq 5,$$
$$x_1 \geq 0, \qquad x_2 \geq 0.$$

Expanding $(x_1 + x_2)^2$ as $x_1^2 + 2x_1x_2 + x_2^2$ and incorporating the factor $\frac{1}{2}$, we rewrite the objective function as:

$$\text{Maximize } f(x) = 20x_1 + 16x_2 + \tfrac{1}{2}(-6x_1x_1 - 4x_2x_2 - 2x_1x_2 - 2x_2x_1),$$

so that $q_{11} = -6$, $q_{12} = -2$, $q_{21} = -2$, $q_{22} = -4$, and the optimality conditions are:

$$s_1 = 5 - x_1 - x_2, \qquad \text{(Primal constraint)}$$
$$\left.\begin{aligned} v_1 &= y_1 - 20 + 6x_1 + 2x_2 \\ v_2 &= y_1 - 16 + 2x_1 + 4x_2 \end{aligned}\right\} \qquad \text{(Dual constraints)}$$
$$x_1, \quad x_2, \quad s_1, \quad v_1, \quad v_2, \quad y_1 \geq 0, \qquad \text{(Nonnegativity)}$$
$$y_1 s_1 = 0, \quad v_1 x_1 = 0, \quad v_2 x_2 = 0. \qquad \text{(Complementary slackness)}$$

Letting s_1 be the basic variable isolated in the first constraint and adding artificial variables a_1 and a_2 in the second and third constraints, the Phase I problem is solved in Table 13.2.

For this problem, the restricted-entry variant of the simplex Phase I procedure has provided the optimal solution. It should be noted, however, that the algorithm will *not* solve every quadratic program. As an example, consider the problem:

$$\text{Maximize } f(x_1, x_2) = \frac{1}{4}(x_1 - x_2)^2,$$

subject to:

$$x_1 + x_2 \leq 5,$$
$$x_1 \geq 0, \qquad x_2 \geq 0.$$

As the reader can verify, the algorithm gives the solution$x_1 = x_2 = 0$. But this solution is not even a local optimum, since increasing either x_1 or x_2 increases the objective value.

It can be shown, however, that the algorithm does determine an optimal solution if $f(x)$ is strictly concave for a maximization problem or strictly convex for a minimization problem. For the quadratic problem, $f(x)$ is strictly concave whenever

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i q_{ij} \alpha_j < 0 \qquad \text{for every choice of } \alpha_1, \alpha_2, \ldots, \alpha_n,$$

such that some $\alpha_j \neq 0$. In this case, the matrix of coefficients (q_{ij}) is called *negative definite*. Thus the algorithm will always work for a number of important applications including the least-square regression and portfolio-selection problems introduced previously.

Table 13.2 Solving a quadratic program.

	Basic variables	*Current values*	x_1	x_2	s_1	y_1^*	v_1	v_2	a_1	a_2
	s_1	5	1	1	1	0	0	0		
←	a_1	20	⑥	2		1	−1	0	1	
	a_2	16	2	4		1	0	−1		1
	$(-w)$	36	8	6		2	−1	−1		

↑

	Basic variables	*Current values*	x_1	x_2	s_1	y_1^*	v_1^*	v_2	a_1	a_2
←	s_1	$\frac{5}{3}$		⑤/3	1	$-\frac{1}{6}$	$\frac{1}{6}$	0	$-\frac{1}{6}$	
	x_1	$\frac{10}{3}$	1	$\frac{1}{3}$		$\frac{1}{6}$	$-\frac{1}{6}$	0	$\frac{1}{6}$	
	a_2	$\frac{28}{3}$		$\frac{10}{3}$		$\frac{2}{3}$	$\frac{1}{3}$	−1	$-\frac{1}{3}$	1
	$(-w)$	$\frac{28}{3}$		$\frac{10}{3}$		$\frac{2}{3}$	$\frac{1}{3}$	0	$-\frac{4}{3}$	

↑

	Basic variables	*Current values*	x_1	x_2	s_1	y_1	v_1^*	v_2^*	a_1	a_2
	x_2	$\frac{5}{2}$		1	$\frac{3}{2}$	$-\frac{1}{4}$	$\frac{1}{4}$	0	$-\frac{1}{4}$	
	x_1	$\frac{5}{2}$	1		$-\frac{1}{2}$	$\frac{1}{4}$	$-\frac{1}{4}$	0	$\frac{1}{4}$	
←	a_2	1			−5	②/2	$-\frac{1}{2}$	−1	$\frac{1}{2}$	1
	$(-w)$	1			−5	$\frac{3}{2}$	$-\frac{1}{2}$	0	$-\frac{1}{2}$	

↑

Basic variables	*Current values*	x_1	x_2	s_1^*	y_1	v_1^*	v_2^*	a_1	a_2
x_2	$\frac{8}{3}$		1	$\frac{2}{3}$		$\frac{1}{6}$	$-\frac{1}{6}$	$-\frac{1}{6}$	$\frac{1}{6}$
x_1	$\frac{7}{3}$	1		$\frac{1}{3}$		$-\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$-\frac{1}{6}$
y_1	$\frac{2}{3}$			$\frac{10}{3}$	1	$\frac{1}{3}$	$-\frac{2}{3}$	$\frac{1}{3}$	$\frac{2}{3}$
$(-w)$	0			0		0	1	−1	−1

* Starred variables cannot be introduced into the basis since their complementary variable is in the basis

13.8 UNCONSTRAINED MINIMIZATION AND SUMT*

Conceptually, the simplest type of optimization is unconstrained. Powerful solution techniques have been developed for solving these problems, which are based primarily upon calculus, rather than upon algebra and pivoting, as in the simplex method. Because the linear-programming methods and unconstrained-optimization techniques are so efficient, both have been used as the point of departure for constructing more general-purpose nonlinear-programming algorithms. The previous sections have indicated some of the algorithms using the linear-programming-based approach. This section briefly indicates the nature of the unconstrained-optimization approaches by introducing algorithms for unconstrained maximization and showing how they might be used for problems with constraints.

Unconstrained Minimization

Suppose that we want to maximize the function $f(x)$ of n decision variables $x = (x_1, x_2, \ldots, x_n)$ and that this function is differentiable. Let $\partial f/\partial x_j$ denote the partialderivative of f with respect to x_j, defined by

* This section requires some knowledge of differential calculus.

$$\frac{\partial f}{\partial x_j} = \lim_{\theta \to 0} \frac{f(x+\theta u_j) - f(x)}{\theta}, \tag{7}$$

where u_j is a decision vector $u_j = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$, with all zeros except for the jth component, which is 1. Thus, $x + \theta u_j$ corresponds to the decisions $(x_1, x_2, \ldots, x_{j-1}, x_j + \theta, x_{j+1}, \ldots, x_n)$, in which only the jth decision variable is changed from the decision vector x.

Note that if $\partial f/\partial x_j > 0$, then

$$\frac{f(x + \theta u_j) - f(x)}{\theta} > 0 \qquad \text{for } \theta > 0 \text{ small enough,}$$

and therefore $x + \theta u_j$ is preferred to x for a maximization problem. Similarly, if $\partial f/\partial x_j < 0$, then

$$\frac{f(x + \theta u_j) - f(x)}{\theta} < 0 \qquad \text{for } \theta < 0 \text{ small enough,}$$

and $x + \theta u_j$ is again preferred to x. Therefore, at any given point with at least one partial derivative $\partial f/\partial x_j > 0$, we can improve the value of f by making a one dimensional search

$$\underset{\theta \geq 0}{\text{Maximize}} \; f(x_1, x_2, \ldots, x_{j-1}, x_j + \theta, x_{j+1}, \ldots, x_n),$$

in which only the value of x_j is varied. Similarly, if $\partial f/\partial x_j < 0$, we can alter x_j to $x_j - \theta$ and search on $\theta \geq 0$ to improve the function's value. The one-dimensional search can be made by using any of a number of procedures that are discussed in the next section. One algorithm using this idea, called *cyclic coordinate ascent*, searches by optimizing in turn with respect to x_1, then x_2, then x_3 and so on, holding all other variables constant at stage j when optimizing with respect to variable x_j. After optimizing with x_n, the method starts over again with x_1.

In fact, there is a large class of algorithms known as *ascent algorithms* that implement the ''uphill movement'' philosophy of cyclic coordinate ascent. Suppose that we consider moving in a general direction $d = (d_1, d_2, \ldots, d_n)$ instead of in a coordinate, or unit, direction. Then, instead of considering the partial derivative, as in (7), we consider the *directional* derivative. The directional derivative, which indicates how the function varies as we move away from x in the direction $d$is defined by:

$$\lim_{\theta \to 0} \frac{f(x + \theta d) - f(x)}{\theta} = \frac{\partial f}{\partial x_1} d_1 + \frac{\partial f}{\partial x_2} d_2 + \cdots + \frac{\partial f}{\partial x_n} d_n. \tag{8}$$

The directional derivative is just the slope of the function $f(x)$ in the direction d and reduces to the definition of the partial derivative $\partial f/\partial x_j$ in Eq. (7) when the direction is taken to be $d = u_j$. Just as in the case of partial derivatives, if the directional derivative in Eq. (8) is positive, then f increases in the direction d; that is,

$$f(x_1 + \theta d_1, x_2 + \theta d_2, \ldots, x_n + \theta d_n) > f(x_1, x_2, \ldots, x_n)$$

for $\theta > 0$ small enough. At any given point x, the ascent algorithms choose an increasing direction d (i.e., such that Eq. (8) is positive), and then select the next point $\bar{x}_i = x_i + \bar{\theta} d_i$ as the solution $\theta = \bar{\theta}$ to the one-dimensional problem

$$\underset{\theta \geq 0}{\text{Maximize}} \; f(x_1 + \theta d_1, x_2 + \theta d_2, \ldots, x_n + \theta d_n).$$

From $\bar{x}$, the ascent algorithms select a new direction and solve another one-dimensional problem, and then continue by repeating this procedure. Cyclic coordinate ascent is a special case in which, at each step, all but one d_j, say d_i, is set to zero; d_i is set to $d_i = +1$ if $\partial f/\partial x_i > 0$ and $d_i = -1$ if $\partial f/\partial x_i < 0$.

One natural choice for the direction d for this general class of algorithms is:

$$d_1 = \frac{\partial f}{\delta x_1}, \quad d_2 \frac{\partial f}{\partial x_2}, \quad \ldots, \quad d_n = \frac{\partial f}{\partial x_n},$$

since then Eq. (8) is positive as long as $\partial f/\partial x_j \neq 0$ for some variable x_j. This choice for d is known as *Cauchy's method*, or *steepest ascent*.

To illustrate the ascent algorithms, consider the objective function

$$f(x_1, x_2) = 20x_1 + 16x_2 - 2x_1^2 - x_2^2 - (x_1 + x_2)^2$$

introduced earlier in this chapter for a portfolio-selection problem. The partial derivatives of this function are:

$$\frac{\partial f}{\partial x_1} = 20 - 6x_1 - 2x_2 \quad \text{and} \quad \frac{\partial f}{\partial x_2} = 16 - 2x_1 - 4x_2.$$

Figure 13.12 shows the result of applying the first few iterations of both cyclic coordinate ascent and steepest ascent for maximizing this function, starting at the point $x_1 = x_2 = 0$.

Cyclic coordinate ascent first increases x_1 from the starting point $x_1 = x_2 = 0$ since $\partial f/\partial x_1 = 20 > 0$. Holding $x_2 = 0$ and setting $\partial f/\partial x_1 = 0$ gives the solution to the first one-dimensional problem at ② as $x_1 = 3\frac{1}{3}$ and $x_2 = 0$. At this point $\partial f/\partial x_2 = 9\frac{1}{3} > 0$ and we increase x_2, holding $x_1 = 3\frac{1}{3}$. The optimum to this one-dimensional problem occurs at point ③ with $\partial f/\partial x_2 = 0$ and $x_1 = 3\frac{1}{3}$, $x_2 = 2\frac{1}{3}$. We now decrease x_1 since $\partial f/\partial x_1 = -4\frac{2}{3}$, generating the next point $x_1 = 2.56$ and $x_2 = 2\frac{1}{3}$. The last point ⑤ shown has $x_1 = 2.56$ and $x_2 = 2.72$. Continuing in this way from this point, the algorithm approaches the optimal solution $x_1 = 2.4$ and $x_2 = 2.8$.

Since the partial derivatives at $x_1 = x_2 = 0$ are $\partial f/\partial x_1 = 20$ and $\partial f/\partial x_2 = 16$, the first direction for steepest ascent is $d_1 = 20$ and $d_2 = 16$, giving the one-dimensional problem:

$$\underset{\theta \geq 0}{\text{Maximize}}\ f(0 + 20\theta, 0 + 16\theta) = 20(20\theta) + 16(16\theta) - 2(20\theta)^2 - (16\theta)^2 - (20\theta + 16\theta)^2 = 656\theta - 2352\theta^2.$$

Setting the derivative with respect to θ equal to zero gives the optimal solution

$$\bar{\theta} = \frac{656}{2(2352)} = 0.1395$$

and the next point

$$\bar{x}_1 = 0 + 20\bar{\theta} = 2.79 \quad \text{and} \quad x_2 = 0 + 16\bar{\theta} = 2.23.$$

At this point, the partial derivatives of f are

$$\frac{\partial f}{\partial x_1} = 20 - 6x_1 - 2x_2 = 20 - 6(2.79) - 2(2.23) = -1.2,$$

and

$$\frac{\partial f}{\partial x_2} = 16 - 2x_1 - 4x_2 = 16 - 2(2.79) - 4(2.23) = 1.5.$$

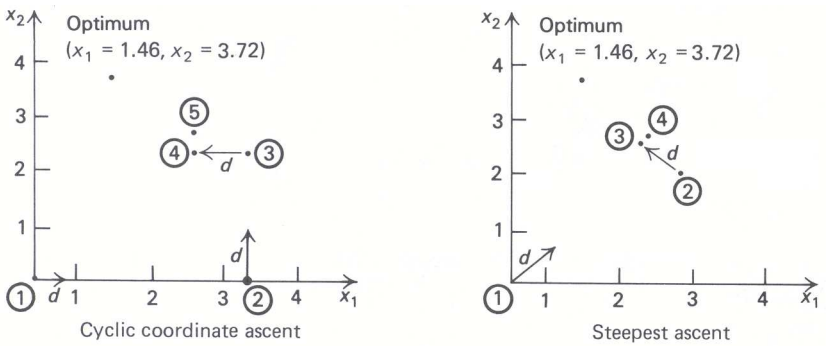

Figure 13.12 Ascent algorithms.

Therefore, the next one-dimensional optimization is in the direction $d_1 = -1.2$ and $d_2 = 1.5$ from the last point; that is,

$$\underset{\theta \geq 0}{\text{Maximize}}\ f(2.79 - 1.2\theta, 2.23 + 1.5\theta).$$

The optimal choice for θ is $\bar{\theta} = 0.354$, giving

$$x_1 = 2.79 - 1.2(0.354) = 2.37 \quad \text{and} \quad x_2 = 2.23 + 1.5(0.354) = 2.76$$

as the next point. Evaluating derivatives gives the direction $d_1 = 0.26$ and $d_2 = 0.22$ and the point ④with $x_1 = 2.40$ and $x_2 = 2.79$, which is practically the optimal solution $x_1 = 2.4$ and $x_2 = 2.8$.

As we have noted, if any partial derivative is nonzero, then the current solution can be improved by an ascent algorithm. Therefore any optimal solution must satisfy the *(first order) optimality conditions*

$$\frac{\partial f}{\partial x_j} = 0 \quad \text{for} \quad j = 1, 2, \ldots, n. \tag{9}$$

As Fig. 13.13 illustrates, these conditions can be satisfied for nonoptimal points as well. Nevertheless, solving the system (9) permits us to generate potential solutions. In fact, we have used this observation above by setting $\partial f/\partial\theta$ to zero to find the solution to the one-dimensional problems. Usually, though, we cannot easily solve for a point that gives zero derivatives, and must rely on numerical methods such as those given in the next section.

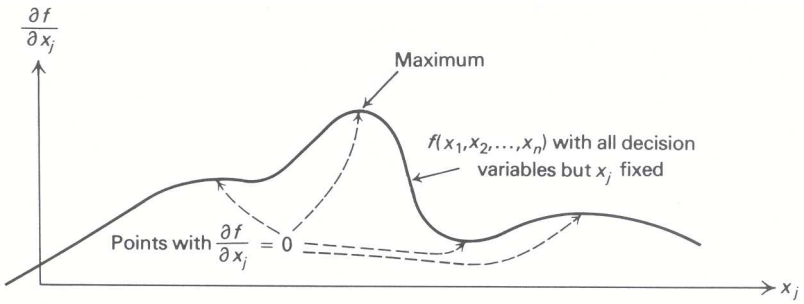

Figure 13.13 Partial derivatives are zero at maximum points.

Since the partial derivatives of the objective function are zero at an optimal solution, ''first-order'' methods like those described in this section, which rely only upon first-derivative information, may encounter numerical difficulties near an optimal solution. Also, in general, first-order methods do not converge to an optimal solution particularly fast. Other methods that use curvature, or second-derivative, information (or more often, approximations to the inverse of the matrix of second partial derivatives) overcome these difficulties, at the expense of more computational work at each step. The *Newton–Raphson* algorithm, which is described in the next section for the special case of one-dimensional optimization, is one popular example of these methods. An entire class of methods known as *conjugate direction* algorithms also use second-derivative information. Instead of reviewing these more advanced methods here, we show how unconstrained algorithms can be used to solve constrained optimization problems.

SUMT (Sequential Unrestrained Maximization Technique)

In principle, any optimization problem can be converted into an unconstrained optimization, as illustrated by the example

$$\text{Minimize } x^2,$$

subject to:

$$\begin{aligned} x &\leq 4, \\ x &\geq 1. \end{aligned} \tag{10}$$

Suppose that we let $P(x)$ denote a penalty for being infeasible, given by:

$$P(x) = \begin{cases} +\infty & \text{if } x \text{ is infeasible (that is, } x > 4 \text{ or } x > 1), \\ 0 & \text{if } x \text{ is feasible (that is, } 1 \leq x \leq 4). \end{cases}$$

Then the constrained optimization problem (10) can be restated in unconstrained form as

$$\text{Minimize } \{x^2 + P(x)\},$$

since the objective function with the penalty term agrees with the original objective function for any feasible point and is $+\infty$ for every infeasible point.

Although this conceptualization in terms of penalties is useful, the method cannot be implemented easily because of numerical difficulties caused by the $+\infty$ terms. In fact, even if a large penalty $M > 0$ is used to replace the $+\infty$ penalty, the method is difficult to implement, since the objective function is discontinuous (i.e., jumps) at the boundary of the feasible region; for example, with $M = 1000$, the term $x^2 + P(x)$ would equal $x^2 + 1000$ to the left of $x = 1$ and only x^2 at $x = 1$. Consequently, we cannot use the algorithms for unconstrained optimization presented in this section, which require differentiability.

We can overcome this difficulty by approximating the penalty term by a smooth function and refining the approximation sequentially by a method known as the *Sequential Unconstrained Maximization Technique*, or SUMT. In this method, instead of giving every infeasible point the same penalty, such as $+\infty$ or a large constant M, we impose a penalty that increases the more a point becomes infeasible.

The curve with $r = 1$ in Fig. 13.14 shows a penalty term used frequently in practice, in which the penalty $P(x)$ grows quadratically as points move farther from the feasible region. In this case, the penalty is zero for feasible points $1 \leq x \leq 4$. To the left of $x = 1$, the penalty is the quadratic term $(1 - x)^2$, and to the right of $x = 4$, the penalty is the quadratic term $(x - 4)^2$, and to the right of $x = 4$, the penalty is the quadratic term $(x - 4)^2$; that is,

$$P(x) = \begin{cases} (1 - x)^2 & \text{if} \quad x \leq 1, \\ 0 & \text{if} \quad 1 \leq x \leq 4, \\ (x - 4)^2 & \text{if} \quad x \geq 4, \end{cases} \tag{11}$$

which is stated compactly as:

$$P(x) = \text{Max } (1 - x, 0)^2 + \text{Max } (x - 4, 0)^2.$$

Note that when $1 \leq x \leq 4$, both maximum terms in this expression are zero and no penalty is incurred; when

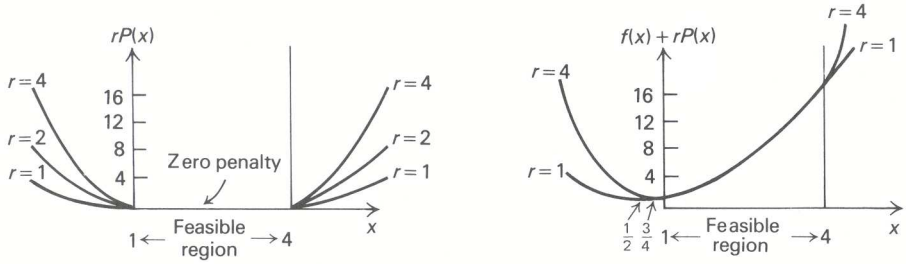

Figure 13.14 Imposing penalty terms sequentially.

$x \leq 1$ or $x \geq 4$, the last expression reduces to the appropriate penalty term in (11).

As Fig. 13.14 illustrates, the infeasible point $x = \frac{1}{2}$ solves the problem

$$\text{Minimize } \{f(x) + P(x)\} = \text{ Minimize } \{x^2 + P(x)\},$$

since the quadratic penalty term does not impose a stiff enough penalty for near-feasible points. Note, however, that if the penalty term $P(x)$ is replaced by $2P(x),\ 3P(x),\ 4P(x)$, or, in general, $rP(x)$ for $r > 1$, the penalty increases for any infeasible point. As the penalty scale-factor r becomes very large, the penalty associated with any particular infeasible point also becomes large, so that the solution to the modified penalty problem

$$\text{Minimize } \{f(x) + rP(x)\}$$

is driven closer and closer to the feasible region. The example problem illustrates this behavior. From Fig. 13.14 we see that, for any $r > 1$, the solution to the penalty problem occurs to the left of $x = 1$, where the penalty term is $r(1 - x)^2$. In this region, the penalty problem of minimizing $x^2 + rP(x)$ reduces to:

$$\text{Minimize } \{x^2 + r(1 - x)^2\}.$$

Setting the first derivative of this objective function to zero gives

$$2x - 2r(1 - x) = 0$$

or

$$x = \frac{r}{r + 1}$$

as the optimal solution. At this point, the objective value of the penalty problem is:

$$x^2 + r \text{ Max } (1 - x, 0)^2 + r \text{ Max } (4 - x, 0)^2 = \left(\frac{r}{r+1}\right)^2 + r\left(1 - \frac{r}{r+1}\right)^2 + r(0)^2$$
$$= \frac{r}{r+1}.$$

Consequently, as r approaches $+\infty$, both the optimal solution $r/(r + 1)$ and the optimal value $r/(r + 1)$ to the penalty problem approach the optimal values $x^* = 1$ and $f(x^*) = 1$ to the original problem.

Although the penalty function cannot always be visualized as nicely as in this simple example, and the computations may become considerably more involved for more realistic problems, the convergence properties exhibited by this example are valid for any constrained problem. The general problem

$$z^* = \text{Min } f(x),$$

subject to:

$$g_i(x) \leq b_i \quad \text{for } i = 1, 2, \ldots, m,$$

is converted into a sequence of unconstrained penalty problems

$$\text{Minimize } \{f(x) + rP(x)\} \tag{12}$$

as r increases to $+\infty$. The penalty term

$$P(x) = \sum_{i=1}^{m} \text{Max } (g_i(x) - b_i, 0)^2$$

introduces a quadratic penalty $[g_i(x) - b_i]^2$ for any constraint i that is violated; that is, $g_i(x) > b_i$. If x^r denotes the solution to the penalty problem (12) when the penalty scale factor is r, then any point x^* that

these points approach in the limit solves the original constrained problem. Moreover, the optimal objective values $f(x^r) + r\,P(x^r)$ to the penalty problems approach the optimal value z^* to the constrained optimization problem.

The general theory underlying penalty-function approaches to constrained optimization permits many variations to the methodology that we have presented, but most are beyond the scope of our introduction to the subject. For example, the absolute value, or any of several other functions of the terms Max $(g_i(x) - b_i, 0)$, can be used in place of the quadratic terms. When equality constraints $h_i(x) = b_i$ appear in the problem formulation, the term $r(h_i(x) - b_i)^2$ is used in the penalty function.

In addition, *barrier methods* can be applied, in which the penalty terms are replaced by barrier terms:

$$\frac{1}{r}\sum_{i=1}^{m}\frac{1}{(g_i(x) - b_i)^2}. \tag{13}$$

In contrast to the SUMT procedure, these methods always remain within the feasible region. Since the barrier term $1/(g_i(x) - b_i)^2$ becomes infinite as x approaches the boundary of the constraint $g_i(x) \le b_i$, where $g_i(x) = b_i$, if the method starts with an initial feasible point, the minimization will not let it cross the boundary and become infeasible. As r becomes large, the barrier term decreases near the boundary and the terms (13) begin to resemble the penalty function with $P(x) = 0$ when x is feasible and $P(x) = +\infty$ when x is infeasible. Figure 13.15 illustrates this behavior when the barrier method is applied to our example problem with $r = 1$ and $r = 4$.

To conclude this section, let us see how the standard penalty procedure, without any of these variations, performs on the portfolio-selection problem that has been solved by several other methods in this chapter. The problem formulation

$$\text{Maximize } \{20x_1 + 16x_2 - 2x_1^2 - x_2^2 - (x_1 + x_2)^2\},$$

subject to:

$$\begin{aligned} x_1 + \quad x_2 &\le 5, \\ x_1 \qquad\; &\ge 0 \qquad (\text{or } -x_1 \le 0) \\ x_2 &\ge 0 \qquad (\text{or } -x_2 \le 0) \end{aligned}$$

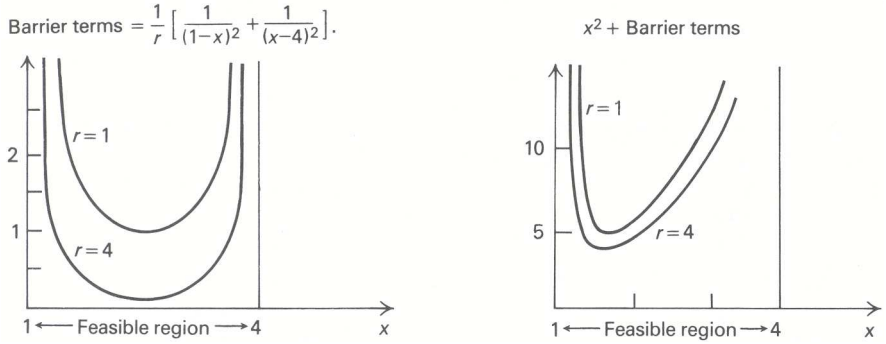

Figure 13.15 The barrier method.

leads to the penalty problem*:

$$\text{Max } [20x_1 + 16x_2 - 2x_1^2 - x_2^2 - (x_1 + x_2)^2 - r \text{ Max } (x_1 + x_2 - 5, 0)^2 - r \text{ Max } (-x_1, 0)^2 - r \text{ Max } (-x_2, 0)^2].$$

We can find the solution to this problem for any value of r by setting to zero the partial derivatives of the terms in braces with respect to both x_1 and x_2, that is, by solving the equations.†

$$20 - 6x_1 - 2x_2 - 2r \text{ Max } (x_1 + x_2 - 5, 0) - 2r \text{ Max } (-x_1, 0) = 0,$$
$$16 - 2x_1 - 4x_2 - 2r \text{ Max } (x_1 + x_2 - 5, 0) - 2r \text{ Max } (-x_2, 0) = 0.$$

The solution to these equations is given by

$$x_1 = \frac{7r^2 + 33r + 36}{3r^2 + 14r + 15}$$

and

$$x_2 = \frac{8r + 14}{3r + 5},$$

as can be verified by substitution in the equations. Figure 13.16 shows how the solutions approach the optimal solution $x_1^* = 2\frac{1}{3},\ x_2^* = 2\frac{2}{3}$, and the optimal objective value $46\frac{1}{3}$ as r increases.

r	x_1	x_2	*Optimal value for the penalty problem*
1	2.375	2.75	46.39
2	2.36	2.73	46.38
5	2.35	2.70	46.36
10	2.34	2.69	46.35
100	2.334	2.669	46.34
$+\infty$	2.333...	2.666...	46.33

Figure 13.16 Penalty method for portfolio selection.

13.9 ONE-DIMENSIONAL OPTIMIZATION

Many algorithms in optimization are variations on the following general approach: given a current feasible solution $x = (x_1, x_2, \ldots, x_n)$, find a direction of improvement $d = (d_1, d_2, \ldots, d_n)$ and determine the next feasible solution $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ as the point that optimizes the objective function along the line segment $\bar{x}_i = x_i + \theta d_i$ pointing away from x in the direction d. For a maximization problem, the new point is the solution to the problem

$$\underset{\theta \geq 0}{\text{Maximize}}\ f(x_1 + \theta d_1, x_2 + \theta d_2, \ldots, x_n + \theta d_n).$$

* Note that we subtract the penalty term from the objective function in this example because we are maximizing rather than minimizing.

† The partial derivative of Max $(x_1 + x_2 - 5, 0)^2$ with respect to x_1 or x_2 equals $2(x_1 + 2_2 - 5)$ if $x_1 + x_2 \geq 5$ and equals zero if $x_1 + x_2 \leq 5$. Therefore it equals 2 Max $(x_1 + x_2 - 5, 0)$. Generally, the partial derivative of Max $(g_i(x) - b_i, 0)^2$ with respect to x_j equals

$$2 \text{ Max } (g_i(x) - b_i, 0) \frac{\partial g_i(x)}{\partial x_j}.$$

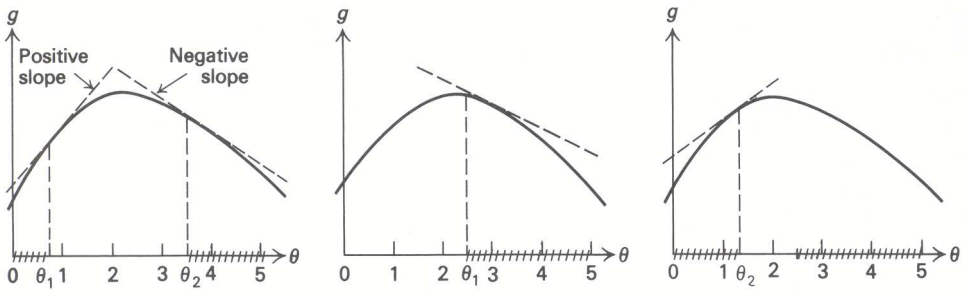

Figure 13.17 Bolzano search for a concave function.

Since the current point $(x_1, x_2, \ldots, x_n)$ and direction $(d_1, d_2, \ldots, d_n)$ are fixed, this problem is a one-dimensional optimization in the variable θ. The direction d used in this method is chosen so that the solution to this one-dimensional problem improves upon the previous point; that is,

$$f(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n) > f(x_1, x_2, \ldots, x_n).$$

Choosing the direction-finding procedure used to implement this algorithm in various ways gives rise to many different algorithms. The Frank-Wolfe algorithm discussed in Section 13.6, for example, determines the direction from a linear program related to the given optimization problem; algorithms for unconstrained optimization presented in the last section select the direction based upon the partial derivatives of the objective function evaluated at the current point x. This section briefly discusses procedures for solving the one-dimensional problem common to this general algorithmic approach; i.e., it considers maximizing a function $g(\theta)$ of a single variable θ.

Search Techniques

Whenever $g(\theta)$ is concave and differentiable, we can eliminate certain points as non-optimal by evaluating the slope $g'(\theta)$ of the function as shown in Fig. 13.17; if $g'(\theta_1) > 0$, then no point $\theta \leq \theta_1$ can maximize g, and if $g'(\theta_2) < 0$, then no point $\theta \geq \theta_2$ can maximize g. This observation is the basis for a method known as *bisection* or *Bolzano search.*

Suppose that the maximum of $g(\theta)$ is known to occur in some interval such as $0 \leq \theta \leq 5$ in Fig. 13.17. Then, evaluating the slope at the midpoint $\theta = 2\frac{1}{2}$ of the interval permits us to eliminate half of the inerval from further consideration—here, $2\frac{1}{2} \leq \theta \leq 5$. By evaluating the slope again at the midpoint $\theta = 1\frac{1}{4}$ of the remaining interval, we can eliminate half of *this* interval. Continuing, we can halve the search interval in which the optimum is known to lie at each step, until we obtain a very small interval. We can then find a point as close as we like to a point that optimizes the given function.

The same type of procedure can be used without evaluating slopes. This extension is important, since derivatives must be evaluated numerically in many applications. Such numerical calculations usually require several function calculations. Instead of evaluating the slope at any point $\theta = \bar{\theta}$, we make function evaluations at two points $\theta = \theta_1$ and $\theta = \theta_2$ close to $\theta = \bar{\theta}$, separated from each other only enough so that we can distinguish between the values $g(\theta_1)$ and $g(\theta_2)$. If $g(\theta_1) < g(\theta_2)$, then every point with $\theta \leq \theta_1$ can be eliminated from further consideration; if $g(\theta_1) > g(\theta_2)$, then points with $\theta \geq \theta_2$ can be eliminated. With this modification; Bolzano search can be implemented by making two function evaluations near the midpoint of any interval, in place of the derivative calculation.

Bolzano's method does not require concavity of the function being maximized. All that it needs is that, if $\theta_1 \leq \theta_2$, then (i) $g(\theta_1) \leq g(\theta_2)$ implies that $g(\theta) \leq g(\theta_1)$ for all $\theta \leq \theta_1$, and (ii) $g(\theta_1) \geq g(\theta_2)$ implies that $g(\theta) \leq g(\theta_2)$ for all $\theta \geq \theta_2$. We call such functions *unimodal,* or single-peaked, since they cannot contain any more than a single local maximum. Any concave function, of course, is unimodal. Figure 13.18 illustrates Bolzano's method, without derivative evaluations, for a unimodal function.

The Bolzano search procedure for unimodal functions can be modified to be more efficient by a method

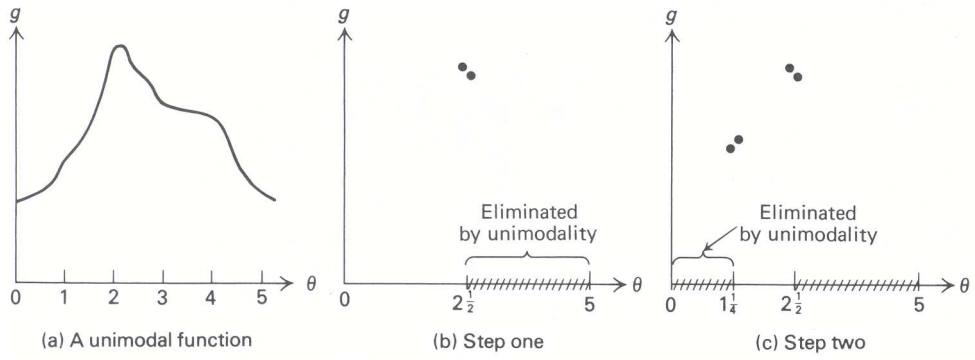

Figure 13.18 Bolzano search for a unimodal function.

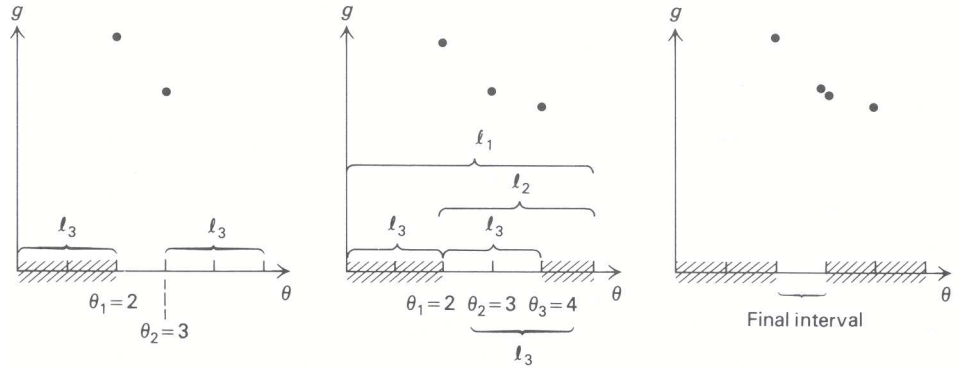

Figure 13.19 Fibonacci search.

known as *Fibonacci search*. This method is designed to reduce the size of the search interval at each step by making only a single function evaluation instead of two evaluations or a derivative calculation, as in Bolzano search.

Figure 13.19 illustrates the method for the previous example. The first two points selected at $\theta_1 = 2$ and $\theta_2 = 3$ eliminate the interval $\theta \leq 2$ from further consideration, by the unimodal property. Note that $\theta_2 = 3$stays within the search interval that remains. By selecting the next point at $\theta_3 = 4$, we eliminate $\theta \geq 4$ from further consideration. Finally, by selecting the last point close to θ_2 at $\theta = 3$, we eliminate $\theta \geq 3$. Consequently, by making four function evaluations, Fibonacci search has reduced the length of the final search interval to 1 unit, whereas four function evaluations (or two, usually more expensive, derivative calculations) in Bolzano search reduced the length of the final search interval to only $1\frac{1}{4}$.

In general, Fibonacci search chooses the function evaluations so that each new point is placed symmetrically in the remaining search interval with respect to the point already in that interval. As Fig. 13.19 illustrates for $k = 1$, the symmetry in placing the function evaluations implies that the length ℓ_k of successive search intervals is given by:

$$\ell_k = \ell_{k+1} + \ell_{k+2}. \tag{14}$$

This expression can be used to determine how many function evaluations are required in order to reduce the final interval to length ℓ_n. By scaling our units of measurement, let us assume that $\ell_n = 1$ (for example, if we want the final interval to have length 0.001, we measure in units of 0.001). Since the final function evaluation just splits the last interval in two, we know that the second-to-last interval has length $\ell_{n-1} = 2$. Then, from Eq. 14, the length of succeeding intervals for function evaluations numbered 3, 4, 5, 6, 7, ..., is given by

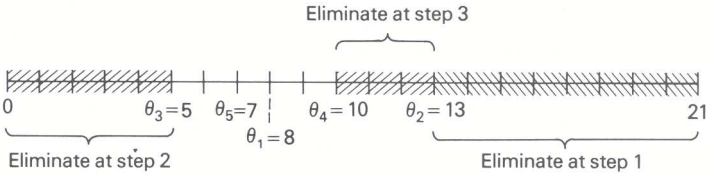

Figure 13.20 A seven-point Fibonacci search.

the ''Fibonacci numbers'':*

$$3 = 1 + 2, \qquad 5 = 2 + 3, \qquad 8 = 3 + 5,$$
$$13 = 5 + 8, \qquad 21 = 8 + 13, \qquad \ldots \tag{15}$$

Consequently, if the initial interval has length 21 (i.e., 21 times larger than the desired length of the final interval), then seven function evaluations are required, with the initial two evaluations being placed at $\theta = 8$ and $\theta = 13$, and each succeeding evaluation being placed symmetrically with respect to the point remaining in the search interval to be investigated further. Figure 13.20 shows a possibility for the first three steps of this application. Fibonacci search is known to be optimal in the sense that, of all search methods guaranteed to reduce the length of the final interval to ℓ_n, it uses the fewest function evaluations. Unfortunately, the length of the final interval must be known in advance, before the location of the initial two function evaluations can be determined. This *a priori* determination of ℓ_n may be inconvenient in practice. Therefore, an approximation to Fibonacci search, called the *method of golden sections,* is used frequently. As the number of function evaluations becomes large, the ratio between succeeding Fibonacci numbers approaches $1/\gamma \approx 0.618$, where $\gamma = (1 + \sqrt{5})/2$ is known as the *golden ratio.* This observation suggests that the first two evaluations be placed symmetrically in the initial interval, 61.8 percent of the distance between the two endpoints, as in golden-section search. Each succeeding point then is placed, as in Fibonacci search, symmetrically with respect to the point remaining in the search interval. Note that this approximation is very close to Fibonacci search even for a procedure with as few as seven points, as in the example shown in Fig. 13.20, since the initial points are at $\frac{13}{21} = 0.619$, or 61.9 percent, of the distance between the two endpoints when applying Fibonacci search. When applied to this example, the first five golden-section points are $\theta_1 = 8.022,\ \theta_2 = 12.978,\ \theta_3 = 4.956,\ \theta_4 = 9.912$, and $\theta_5 = 6.846$, as compared to $\theta_1 = 8,\ \theta_2 = 13,\ \theta_3 = 5,\ \theta_4 = 10$, and $\theta_5 = 7$ from Fibonacci search.

Curve Fitting and Newton's Method

Another technique for one-dimensional optimization replaces the given function to be maximized by an approximation that can be optimized easily, such as a quadratic or cubic function. Figure 13.21, for instance, illustrates a quadratic approximation to our previous example. By evaluating the given function $g(\theta)$ at three points $\theta = \theta_1,\ \theta_2$, and θ_3, we can determine a quadratic function

$$q(\theta) = a\theta^2 + b\theta + c,$$

which agrees with $g(\theta)$ at $\theta_1,\ \theta_2$, and θ_3. In this case, the resulting quadratic approxi- mation is:

$$q(\theta) = -\frac{25}{8}\theta^2 + \frac{65}{4}\theta + 10.$$

By setting the first derivative $q'(\theta)$ of the approximation to 0, we can solve for an approximate optimal

* The method is called Fibonacci search because the numbers in expression (15), which arise in many other applications, are called the Fibonacci numbers.

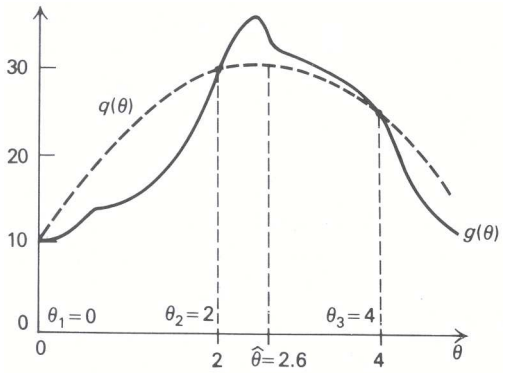

Figure 13.21 Quadratic approximation.

solution $\hat{\theta}$. Here,

$$q'(\theta) = -\frac{50}{8}\theta + \frac{65}{4},$$

and

$$\hat{\theta} = \frac{(65/4)}{(50/8)} = 2.6 \quad \text{with } q(\hat{\theta}) = 31.1.$$

After determining this approximation, we can use the new point $\hat{\theta}$, together with two of the points θ_1, θ_2, and θ_3, to define a new approximation. We select the three points, $\hat{\theta}$ and two others, so that the middle point has the highest value for the objective function, since then the approximating quadratic function has to be concave. Since $g(\hat{\theta}) = g(2.6) > 30$ in this case, we take θ_2, $\hat{\theta}$, and θ_3 as the new points. If $g(\theta) \leq 30$, we would take θ_1, θ_2, and $\hat{\theta}$ as the three points to make the next approximation. By continuing to make quadratic approximations in this way, the points $\hat{\theta}_1$, $\hat{\theta}_2$, ... determined at each step converge to an optimal solution θ^*. [†] In practice, the method may *not* find an optimal solution θ^* and stop in a finite number of steps; therefore, some finite termination rule is used. For example, we might terminate when the function values $g(\hat{\theta}_j)$ and $g(\hat{\theta}_{j+1})$ for two succeeding points become sufficiently small—say, within $\epsilon = 0.00001$ of each other.

Similar types of approximation methods can be devised by fitting with cubic functions, or by using other information such as derivatives of the function for the approximation. *Newton's method* for example, which is possibly the most famous of all the approximating schemes, uses a quadratic approximation based upon first- and second-derivative information at the current point θ_j, instead of function evaluations at three points as in the method just described. The approximation is given by the quadratic function:

$$q(\theta) = g(\theta_j) + g'(\theta_j)(\theta - \theta_j) + \tfrac{1}{2}g''(\theta_j)(\theta - \theta_j)^2. \tag{16}$$

Note that the $q(\theta_j) = g(\theta_j)$ and that the first and second derivatives of $q(\theta)$ agree with the first derivative $g'(\theta_j)$ and the second derivative $g''(\theta_j)$ of $g(\theta)$ at $\theta = \theta_j$.

The next point θ_{j+1} is chosen as the point maximizing $q(\theta)$. Setting the first derivative of $q(\theta)$ in Eq. 16 to zero and solving for θ_{j+1} gives the general formula

$$\theta_{j+1} = \theta_j - \frac{g'(\theta_j)}{g''(\theta_j)}, \tag{17}$$

[†] In the sense that for any given $\epsilon > 0$, infinitely many of the $\hat{\theta}_j$ are within ϵ of θ^*; that is,

$$\theta^* - \epsilon \leq \hat{\theta}_j \leq \theta^* + \epsilon.$$

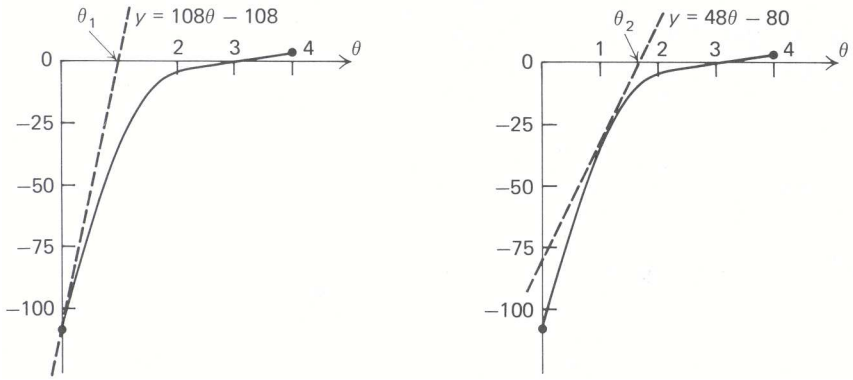

Figure 13.22 Newton's method for solving $h\,\theta = 0$.

for generating successive points by the algorithm.

Letting $h(\theta) = g'(\theta)$ denote the first derivative of $g(\theta)$ suggests an interesting interpretation of expression (17). Since $g'(\hat{\theta}) = 0$ at any point $\hat{\theta}$ maximizing $g(\theta)$, $\hat{\theta}$ must be a solution to the equation $h(\hat{\theta}) = 0$. Newton's method for solving this equation approximates $h(\theta)$ at any potential solution θ_j by a line tangent to the function at $\theta = \theta_j$ given by:

$$\text{Approximation: } y = h(\theta_j) + h'(\theta_j)(\theta - \theta_j).$$

Since $h(\theta_j) = g'(\theta_j)$ and $h'(\theta_j) = g''(\theta_j)$, the solution $y = 0$ to this approximation is the point θ_{j+1} specified in (17). It is because of this association with Newton's well- known method for solving equations $h(\theta) = 0$ that the quadratic-approximation scheme considered here is called Newton's method.

To illustrate Newton's method, let $g(\theta) = (\theta - 3)^4$. Then $h(\theta) = g'(\theta) = 4(\theta - 3)^3$ and $h'(\theta) = g''(\theta) = 12(\theta - 3)^2$. Starting with $\theta_0 = 0$, Newton's method gives:

$$\theta_{j+1} = \theta_j - \frac{h(\theta_j)}{h'(\theta_j)} = \theta_j - \frac{4(\theta_j - 3)^3}{12(\theta_j - 3)^2} = \theta_j - \frac{\theta_j - 3}{3} = \tfrac{2}{3}\theta_j + 1.$$

The points $\theta_1 = 1$, $\theta_2 = 1\frac{2}{3}$, $\theta_3 = 2\frac{1}{9}$, $\theta_3 = 2\frac{11}{27}$, $\theta_4 = 2\frac{49}{81}, \ldots$ that the method generates converge to the optimal solution $\theta^* = 3$. Figure 13.22 shows the first twoapproximations to $h(\theta)$ for this example.

Newton's method is known to converge to an optimal solution θ^* that maximizes $g(\theta)$ as long as $g''(\theta^*) \neq 0$ and the initial point θ_0 is close enough to θ^*. The precise definition of what is meant by ''close enough to θ^*'' depends upon the problem data and will not be developed here. We simply note that, because the rate of convergence to an optimal solution has been shown to be good for Newton's method, it often is used in conjunction with other procedures for one-dimensional optimizations that are used first to find a point θ_0 close to an optimal solution.

EXERCISES

1. a) Over what region is the function $f(x) = x^3$ convex? Over what region is f concave?
   b) Over what region is the function $f(x) = -12x + x^3$ convex (concave)?
   c) Plot the function $f(x) = -12x + x^3$ over the region $-3 \leq x \leq 3$. Identify local minima and local maxima of f over this region. What is the global minimum and what is the global maximum?
2. a) Which of the following functions are convex, which are concave, and which are neither convex nor concave?
   i) $f(x) = |x|$.
   ii) $f(x) = \frac{1}{x}$ over the region $x > 0$.

iii) $f(x) = \log(x)$ over the region $x > 0$.
iv) $f(x) = e^{-x^2}$.
v) $f(x_1, x_2) = x_1 x_2$.
vi) $f(x_1, x_2) = x_1^2 + x_2^2$.

b) Graph the feasible solution region for each of the following constraints:

i) $x_1^2 + x_2^2 \leq 4$.
ii) $x_1^2 + x_2^2 = 4$.
iii) $x_1^2 + x_2^2 \geq 4$.
iv) $x_2 - |x_1| \leq 0$.
v) $x_2 - |x_1| \geq 0$.

Which of these regions is convex?

c) Is the function $f(x_1, x_2) = x_2 - |x_1|$ concave?

3. Because of the significance of convexity in nonlinear programming, it is useful to be able to identify convex sets and convex (or concave) functions easily. Apply the definitions given in this chapter to establish the following properties:

a) If C_1 and C_2 are two convex sets, then the intersection of C_1 and C_2 (that is, points lying in both) is a convex set. For example, since the set C_1 of solutions to the inequality

$$x_1^2 + x_2^2 \leq 4$$

is convex and the set C_2 of points satisfying

$$x_1 \geq 0 \quad \text{and} \quad x_2 \geq 0$$

is convex, then the feasible solution to the system

$$\begin{aligned} x_1^2 + x_2^2 &\leq 4, \\ x_1 \geq 0, \quad x_2 &\geq 0, \end{aligned}$$

which is the intersection of C_1 and C_2, is also convex. Is the intersection of more than two convex sets a convex set?

b) Let $f_1(x), f_2(x), \ldots, f_m(x)$ be convex functions and let $\alpha_1, \alpha_2, \ldots, \alpha_m$ be nonnegative numbers; then the function

$$f(x) = \sum_{j=1}^{m} \alpha_j f_j(x)$$

is convex. For example, since $f_1(x_1, x_2) = x_1^2$ and $f_2(x_1, x_2) = |x_2|$ are both convex functions of x_1 and x_2, the function

$$f(x_1, x_2) = 2x_1^2 + |x_1|$$

is convex.

c) Let $f_1(x)$ and $f_2(x)$ be convex functions; then the function $f(x) = f_1(x) f_2(x)$ need not be convex. [*Hint*. See part (a(v)) of the previous exercise.]

d) Let $g(y)$ be a convex and nondecreasing function [that is, $y_1 \leq y_2$ implies that $g(y_1) \leq g(y_2)$] and let $f(x)$ be a convex function; then the composite function $h(x) = g[f(x)]$ is convex. For example, since the function e^y is convex and nondecreasing, and the function $f(x_1, x_2) = x_1^2 + x_2^2$ is convex, the function

$$h(x_1, x_2) = e^{x_1^2 + x_2^2}$$

is convex.

4. One of the equivalent definitions for convexity of a differentiable function f of single variable x given in this chapter is that the slopes of the function be nondecreasing with respect to the variable x; that is,

$$\left.\frac{df}{dx}\right|_{x=y} \geq \left.\frac{df}{dx}\right|_{x=z} \qquad \text{whenever } y \geq z^*.$$

For a strictly convex function, the condition becomes:

$$\left.\frac{df}{dx}\right|_{x=y} > \left.\frac{df}{dx}\right|_{x=z} \qquad \text{whenever } y > z.$$

Similar conditions can be given for concave functions. If the function f has second derivations, these conditions are equivalent to the very useful second-order criteria shown in Table E13.1.

Table E13.1

Function property	*Condition on the second derivative* $\frac{d^2f}{dx^2}$ *implying the function property*
Convex	$\frac{d^2f}{dx^2} \geq 0$ for all values of x
Concave	$\frac{d^2f}{dx^2} \leq 0$ for all values of x
Strictly convex	$\frac{d^2f}{dx^2} > 0$ for all values of x
Strictly concave	$\frac{d^2f}{dx^2} < 0$ for all values of x

For example, if $f(x) = x^2$, then

$$\frac{d^2 f}{dx^2} = 2,$$

implying that x^2 is strictly convex.

Not only do the second-order conditions imply convexity or concavity, but any convex or concave function must also satisfy the second-order condition. Consequently, since

$$\frac{d^2 f}{dx^2} = 6x$$

can be both positive and negative for the function $f(x) = x^3$, we know that x^3 is neither convex nor concave.

Use these criteria to show which of the following functions are convex, concave, strictly convex, or strictly concave.

a) $4x + 2$ b) e^x

c) $\frac{1}{x}$ for $x > 0$ d) $\frac{1}{x}$ for $x < 0$

e) x^5 f) $\log(x)$ for $x > 0$

g) $\log(x^2)$ for $x > 0$ h) $f(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ x^2 & \text{if } x > 0. \end{cases}$

i) x^4 [*Hint.* Can a function be strictly convex even if its second derivative is zero at some point?]

* $\left.\frac{df}{dx}\right|_{x=y}$ means the derivative of f with respect to x evaluated at $x = y$.

5. Brite-lite Electric Company produces two types of electric lightbulbs; a 100-watt bulb and a 3-way (50–100–150) bulb. The machine that produces the bulbs is fairly old. Due to higher maintenance costs and down time, the contribution per unit for each additional unit produced decreases as the machine is used more intensely. Brite-lite has fit the following functions to per-unit contribution (in \$) for the two types of lightbulbs it produces:

$$f_1(x_1) = 50 - 2x_1, \text{ for } x_1 \text{ units of 100-watt bulbs;}$$
$$f_2(x_2) = 70 - 3x_2, \text{ for } x_2 \text{ units of 3-way bulbs.}$$

The Brite-lite Company is in the unusual position of being able to determine the number of units demanded of a particular type of lightbulb from the amount (in \$) it spends on advertisements for that particular bulb.

The equations are:

$$150 - \frac{1000}{10 + y_1} = \text{Number of units of 100-watt bulbs demanded,}$$
$$250 - \frac{2000}{10 + y_2} = \text{Number of units of 3-way bulbs demanded,}$$

where y_1 is the amount (in \$) spent on advertisements for 100-watt bulbs, and y_2 is the amount (in \$) spent on advertisements for 3-way bulbs.

Brite-lite has an advertising budget of \$1500. Its management is committed to meeting demand. Present production capacity is 125 and 175 units of 100-watt and 3-way bulbs, respectively.

Finally, the production of a unit of lightbulb j consumes a_{ij} units of resource i for $i = 1, 2, \ldots, m$, of which only b_i units are available.

6. Besides its main activity of providing diaper service, Duke-Dee Diapers, a small workshop owned by a large fabrics firm, produces two types of diapers. The diapers are constructed from a special thin cloth and an absorbent material. Although both materials are manufactured by the fabrics company, only limited monthly amounts, b_1 and b_2, respectively, are available because this activity always has been considered as secondary to production of other fabrics. Management has decided to analyze the activity of the workshop in terms of profits, to decide whether or not to continue its diaper-service operation. A limited market survey showed that the following linear functions give good approximations of the demand curves for the two types of diapers:

$$p_j = \beta_j - \alpha_{j1}x_1 - \alpha_{j2}x_2 \qquad (j = 1, 2),$$

where

p_j = Unit price for the diapers,
β_j = Constant,
α_{jk} = Coefficient related to the substitutability of the two types of diapers (if $\alpha_{11} = 1, \alpha_{12} = 0, \alpha_{22} = 1, \alpha_{21} = 0$, there is no substitutability), and
x_k = Units of diapers of type k sold.

Each unit of type j diaper costs c_j $(j = 1, 2)$ to produce and uses a_{ij} $(i = 1, 2;\ j = 1, 2)$ units of resource i.

a) Formulate a model to determine the profits from diaper production.
b) Show how you would transform the model into a separable program.

7. A connoisseur has m bottles of rare wine in his cellar. Bottle j is currently t_j years old. Starting next year, he wishes to drink one bottle each year on his birthday, until his supply is exhausted. Suppose that his utility for drinking bottle j is given by $U_j(t)$ (t is its age when consumed) and that utility is additive. The connoisseur wishes to know in what order to drink the wine to maximize his total utility of consumption.

E

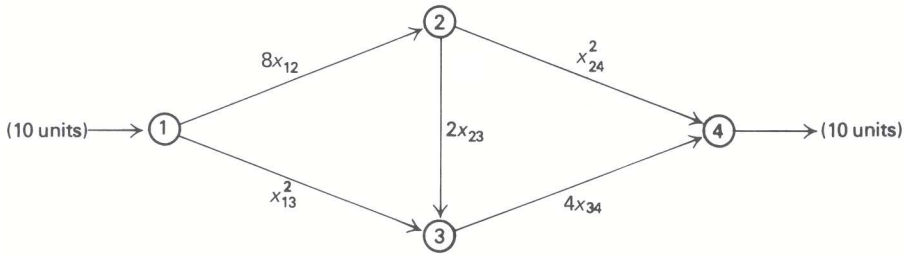

Figure E13.1 Network with arc costs.

a) Formulate his decision problem of which bottle to select each year as an integer linear program. What is the special structure of the formulation? How can this structure help in solving the problem?
b) Suppose that the utility for consuming each bottle of wine is the same, given by a convex function $U(t)$. Show that it is optimal to select the youngest available bottle each year.

8. Consider a directed communications network, where each arc represents the communication link between two geographically distant points. Each communication link between points i and j has a probability of failure of p_{ij}. Then, for any path through the network, the probability of that particular path being operative is the product of the probabilities $(1 - p_{ij})(1 - p_{jk})(1 - p_{kl}) \ldots$ of the communication links connecting the two points of interest.

a) Formulate a network model to determine the path from s to t having the highest probability of being operative.
b) In what class does the problem formulated in (a) fall?
c) How can this formulation be transformed into a network problem?

9. In Section 4 of Chapter 1, we formulated the following nonlinear-programming version of the custom-molder example:

$$\text{Maximize } 60x_1 - 5x_1^2 + 80x_2 - 4x_2^2,$$

subject to:

$$\begin{aligned} 6x_1 + 5x_2 &\leq 60, \\ 10x_1 + 12x_2 &\leq 150, \\ x_1 &\leq 8, \\ x_1 \geq 0, \quad x_2 &\geq 0. \end{aligned}$$

Using the following breakpoints, indicate a separable-programming approximation to this problem:

$$x_1 = 2, \quad x_1 = 4, \quad x_1 = 6,$$

and

$$x_2 = 3, \quad x_2 = 6, \quad x_2 = 10.$$

Do *not* solve the separable program.

10. In the network of Fig. E13.1, we wish to ship 10 units from node 1 to node 4 as cheaply as possible.

The flow on each arc is uncapacitated. The costs on arcs 1–2, 2–3, and 3–4 are linear, with costs per unit of 8, 2, and 4, respectively; the costs on arcs 1–3 and 2–4 are quadratic and are given by x_{13}^2 and x_{24}^2.

a) Suppose that we apply the separable-programming δ-technique discussed in Chapter 9, using the breakpoints

$$x_{13} = 0, \quad x_{13} = 2, \quad x_{13} = 6, \quad x_{13} = 10$$

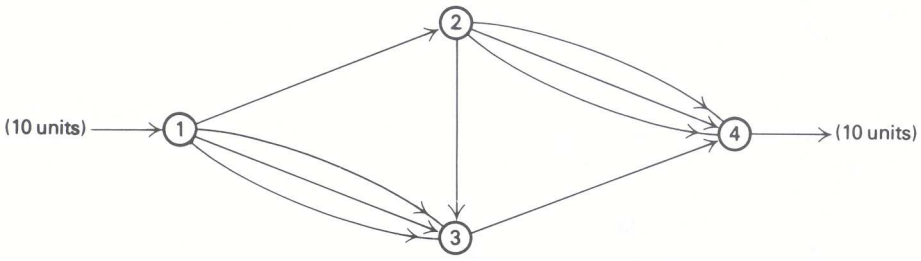

Figure E13.2 Linear approximation with parallel arcs.

and

$$x_{24} = 0, \quad x_{24} = 2, \quad x_{24} = 6, \quad x_{24} = 10.$$

Interpret the resulting linear-programming approximation as a network-flow problem with parallel arcs joining nodes 1 and 3 and nodes 2 and 4, as in Fig. E13.2. Specify the per-unit cost and the arc capacity for each arc in the linear approximation.

b) Solve the separable-programming approximation from part (a), comparing the solution with the optimal solution

$$x_{12} = 5, \quad x_{13} = 5, \quad x_{23} = 3.5, \quad x_{24} = 1.5, \quad x_{34} = 8.5,$$

and minimum cost $= 110.25$, to the original nonlinear problem formulation.

11. a) What is the special form of the linear approximation problem when the Frank-Wolfe algorithm is applied to the network example in the previous exercise?
    b) Complete Table E13.2 for applying the first two steps of the Frank-Wolfe algorithm to this example.

Table E13.2

Arc	*Initial solution*	*Solution to linear approximation*	*Second solution*	*Solution to linear approximation*	*Next solution*
1–2	10				
1–3	0				
2–3	10				
2–4	0				
3–4	10				
Total cost	140	———		———	

12. Solve the following nonlinear program using the λ-method of separable programming described in this chapter.

$$\text{Maximize } 2x_1 - x_1^2 + x_2,$$

subject to:

$$\begin{aligned} x_1^2 + x_2^2 &\leq 4, \\ x_2 &\leq 1.8, \\ x_1,\ x_2 &\geq 0. \end{aligned}$$

Carry out calculations using two decimal places. For each decision variable, use a grid of 5 points (including the extreme values).

13. Two chemicals can be produced at one facility, and each takes one hour per ton to produce. Both exhibit diseconomies of scale: If x_1 tons of the first are produced, the contribution is $6x_1 - (x_1^2/2)$; for the second chemical, the contribution is $\sqrt{50x_2}$ for x_2 tons produced. The facility can work 23 hours per day (one hour is required for maintenance) and, because of raw materials availability, no more than 10 tons of the first chemical and 18 tons of the second can be produced daily. What is the optimal daily production schedule, to maximize contribution?

    Solve by separable programming, using the δ-method described in Chapter 9. For x_1 use a grid: 0, 3, 6, 10, and for x_2 use a grid 0, 2, 5, 18.

14. A young R & D engineer at Carron Chemical Company has synthesized a sensational new fertilizer made of just two interchangeable basic raw materials. The company wants to take advantage of this opportunity and produce as much as possible of the new fertilizer. The company currently has \$40, 000 to buy raw materials at a unit price of \$8000 and \$5000 per unit, respectively. When amounts x_1 and x_2 of the basic raw materials are combined, a quantity q of fertilizer results given by:

$$q = 4x_1 + 2x_2 - 0.5x_1^2 - 0.25x_2^2.$$

    a) Formulate as a nonlinear program.
    b) Apply four iterations of the Frank-Wolfe algorithm; graphically identify the optimal point, using the property that even- and odd-numbered points lie on lines directly toward the optimum. Start from $x^0 = (0, 0)$ and use three decimal places in your computations.
    c) Solve the problem using the algorithm for quadratic programming discussed in Section 13.7.

15. A balloon carrying an x-ray telescope and other scientific equipment must be designed and launched. A rough measure of performance can be expressed in terms of the height reached by the balloon and the weight of the equipment lifted. Clearly, the height itself is a function of the balloon's volume.

    From past experience, it has been concluded that a satisfactory performance function to be maximized is $P = f(\mathrm{V}, \mathrm{W}) = 100V - 0.3V^2 + 80W - 0.2W^2$ where V is the volume, and W the equipment weight.

    The project to be undertaken has a budget constraint of \$1040. The cost associated with the volume V is $2V$, and the cost of the equipment is $4W$. In order to ensure that a reasonable balance is obtained between performance due to the height and that due to the scientific equipment, the designer has to meet the constraint $80W \geq 100V$.

    Find the optimal design in terms of volume and equipment weight, solving by the Frank-Wolfe algorithm.

16. Consider the nonlinear-programming problem:

$$\text{Maximize } 2x_1 - x_1^2 + x_2,$$

    subject to:

$$\begin{aligned} x_1^2 + x_2^2 &\leq 4, \\ x_2 &\leq 1.8, \\ x_1 \geq 0, \quad x_2 &\geq 0. \end{aligned}$$

    a) Carry out two iterations of the generalized programming algorithm on this problem. Let

$$C = \{(x_1, x_2) | x_1 \geq 0, \quad 0 \leq x_2 \leq 1.8\},$$

    leaving only one constraint to handle explicitly. Start with the following two initial candidate solutions:

$$x_1 = 0, \quad x_2 = 1.8, \quad \text{and} \quad x_1 = 2, \quad x_2 = 0.$$

    Is the solution optimal after these two iterations?

b) Carry out two iterations of the MAP algorithm on the same problem. Start with $x_1 = 0$, $x_2 = 0$, as the initial solution, and set the parameters $\delta_1 = \delta_2 = 2$ at iteration 1 and $\delta_1 = \delta_2 = 1$ at iteration 2. Is the solution optimal after these two iterations?

17. An orbital manned scientific lab is placed on an eccentric elliptical orbit described by $x^2+5y^2+x+3y = 10$, the reference system (x, y) being the center of the earth. All radio communications with the ground stations are going to be monitored via a satellite that will be fixed with respect to the reference system. The power required to communicate between the satellite and the lab is proportional to the square of the distance between the two.

    Assuming that the satellite will be positioned in the plane of the ellipse, what should be the position of the satellite so that the maximum power required for transmissions between the lab and the satellite can be minimized? Formulate as a nonlinear program.

18. An investor has \$2 million to invest. He has 5 opportunities for investment, with the following characteristics:

    i) The yield on the first investment is given by a linear function:

    $$r_1 = 3 + 0.000012x_1,$$

    where r_1 = yield per year (%), and x_1 = amount invested (\$).

Minimum required:	\$ 100,000
Maximum allowed:	\$1,000,000
Years to maturity:	6

    ii) The second investment yields:

    $$r_2 = 2 + 0.000018x_2,$$

    where r_2 = yield per year (%), and x_2 = amount invested (\$).

Minimum required:	\$ 200,000
Maximum allowed:	\$1,000,000
Years to maturity:	10

    iii) An investment at 5% per year with interest continuously compounded. (An amount A invested at 5% per year with continuously compounded interest becomes $Ae^{0.05}$ after one year.)

    Years to maturity: 1

    iv) Category 1 of government bonds that yield 6% per year.

    Years to maturity: 4

    v) Category 2 of government bonds that yield 5.5% per year.

    Years to maturity: 3

    The average years to maturity of the entire portfolio must not exceed 5 years.

    a) The objective of the investor is to maximize accumulated earnings at the end of the first year. Formulate a model to determine the amounts to invest in each of the alternatives. Assume all investments are held to maturity.

    b) Identify the special type of nonlinear program obtained in part (a).

19. Since its last production diversification, New Home Appliances, Inc. (NHA), a kitchen equipment manufacturer, has encountered unforeseen difficulty with the production scheduling and pricing of its product line. The linear model they have used for a number of years now no longer seems to be a valid representation of the firm's operations.

In the last decade, the output of the firm has more than tripled and they have become the major supplier of kitchen equipment in the southeastern United States market. After conducting a market survey as well as a production-cost analysis, the consulting firm hired by NHA has concluded that the old linear model failed to optimize the activity of the firm because it did not incorporate the following new characteristics of NHA as a major supplier in the southeastern market;

I. NHA was no longer in a perfectly competitive situation, so that it had to take into account the market demand curve for its products. The consulting firm found that for NHA's 15 principal products, the price elasticity of demand was roughly 1.2; hence, the market demand curve can be expressed by:

$$x_j p_j^{1.2} = 1,600,000 \quad (j = 1, 2, \ldots, 15), \tag{1}$$

where x_j = units of product j sold; and p_j = unit price of product j. For the remaining 25 products, the price is known and can be considered constant for all levels of sales.

II. Since output has increased, the firm has reached the stage where economies of scale prevail; thus, the per-unit production cost c_j decreases according to:

$$c_j = \gamma_j - \delta_j x_j \quad (j = 1, 2, \ldots, 15), \tag{2}$$

where γ_j and δ_j are coefficients determined individually for the 15 principal products. For the remaining 25 products, constant returns to scale is a reasonable assumption (i.e., a linear relationship exists between the amount produced and the cost); consider the production costs per unit as known.

III. Production of each unit of product j consumes a_{ij} units of resource i. Resource utilization is limited by a budget of B dollars, which is available for the purchase of raw materials and labor. The suppliers of sheet steel and aluminum are offering discount prices for larger quantities. Linear regression leads to the following equations that show the relationship between the amount ordered and the unit price of each:

$$\mu_s = \alpha_s - \beta_s b_s,$$
$$\mu_a = \alpha_a - \beta_a b_a,$$

Where μ_s, μ_a are the unit prices for steel and aluminum, respectively; b_s, b_a are the amounts contracted, and $\alpha_s, \alpha_a, \beta_s, \beta_a$ are coefficients. No discounts are available for other resources, because NHA's consumption falls below the level at which such discounts are offered. Unit prices for all other resources are constant and known. Besides steel and aluminum, 51 resources are used.

Formulate a mathematical program that incorporates the above information; the objective is the maximization of contribution (revenues − costs). what type of nonlinear program have you obtained?

20. A rent-a-car company operating in New York City serves the three major airports—Kennedy, La Guardia, and Newark—and has two downtown distribution centers. On Sunday evenings most of the cars are returned to the downtown locations by city residents returning from weekend travels. On Monday morning, most of the cars are needed at the airports and must be ''deadheaded'' by company drivers.

The two downtown distribution centers have a_i $(i = 1, 2)$ excess cars available. The three airports must be supplied with cars at a transportation cost of c_{ij} $(i = 1, 2;\ j = 1, 2, 3)$ for deadheading from distribution center i to airport j. The Monday morning demand r_j for cars at airport j is uncertain and is described by the probability distribution p_j (r_j). If the demand exceeds supply at airport j, the unsatisfied demand is lost, with an average lost contribution per car of u_j.

a) Formulate a mathematical program to minimize total deadheading transportation cost plus expected lost contribution.
b) Suppose now that the manager of fleet allocation is also concerned with supply over demand. All cars in excess of 50 cars above demand must be parked in an overflow lot at a cost of s_j per car. Reformulate the program to include these expected average costs.

21. After the admissions decisions have been made for the graduate engineering school, it is the Scholarship Committee's job to award financial aid. There is never enough money to offer as much scholarship aid as each needy applicant requires.

Each admitted applicant's financial need is determined by comparing an estimate of his sources of revenue with a reasonable school and personal expense budget for the normal academic year. An admittee's need, if any, is the difference between the standard budget and the expected contribution from him and his family. Scholarship offers provide an amount of aid equal to some fraction of each applicant's need. In cases where need is not met in full, the school is able to supply low-cost loans to cover the difference between scholarships and need.

Besides receiving funds from the university, a needy admittee might receive a scholarship from nonuniversity funds. In this case the admittee, if he decides to matriculate, is expected to accept the outside funds. His total scholarship award is then the greater of the university offer or the outside offer, because the university supplements any outside offer up to the level awarded by the scholarship committee. Prior to the deadline for determining a school scholarship-offer policy, the committee has a good estimate of the amount of outside aid that each needy admittee will be offered.

The most important function of the scholarship policy is to enroll the *highest-quality* needy admittees possible. The admissions committee's rank list of all needy admittees is used as the measurement of quality for the potential students.

In using this list, the top $100\alpha\%$ of the needy group ordered by quality is expected to yield at least βT enrollees, where T is the total desired number of enrollees from the needy group. In addition to satisfying the above criteria, the dean wants to minimize the total expected cost of the scholarship program to the university.

As a last point, P_i, the probability that needy admittee i enrolls, is an increasing function of y_i, the fraction of the standard budget B covered by the total scholarship offer. An estimate of this function is given in Fig. E13.3.Here, $x_i B$ is the dollar amount of aid offered admittee i and $n_i B$ is the dollar amount of need for admittee i.

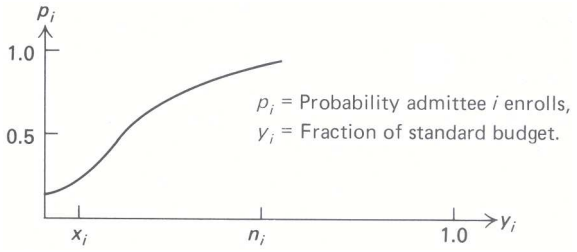

Figure E13.2

a) Formulate a nonlinear-programming model that will have an expected number T of enrolling needy admittees, and minimize the scholarship budget. (Assume that P_i can be approximated by a linear function.)
b) Suggest two different ways of solving the model formulated in (a). [*Hint.* What special form does the objective function have?]
c) Reformulate the model so as to maximize the expected number of enrolling needy students from the top $100\alpha\%$ of the need group, subject to a fixed total scholarship budget. Comment on how to solve this variation of the model.

d) suppose that, in the formulation proposed in (a), the probability P_i that admittee i enrolls is approximated by $P_i = a_i + b_i y_i^{1/2}$. Comment on how to solve this variation of the model.

22. A well-known model to solve the aggregate production-planning problem that permits quadratic cost functions in the model's objective was developed by Holt, Modigliani, Muth, and Simon.* The model allocates manpower, production, and inventories during a prescribed planning horizon, divided into t time periods denoted by $t = 1, 2, \ldots, T$. The decision variables of the model are:

P_t = Production rate for period t;
W_t = Work-force size used in period t;
I_t = Ending inventory at period t.

If d_t is the demand to be satisfied during period t, the constraints of the model are:

$$P_t + I_{t-1} - I_t = d_t \quad (t = 1, 2, \ldots, T).$$

The objective function is to minimize the sum of the cost elements involved in the production process. Holt, Modigliani, Muth, and Simon identified the following cost elements for each time period:

i) Regular payroll cost = $c_1 W_t + c_{13}$;
ii) Hiring and firing cost = $c_2(W_t - W_{t-1} - c_{11})^2$;
iii) Overtime and idle cost = $c_3(P_t - c_4 W_t)^2 + c_5 P_t - c_6 W_t + c_{12} P_t W_t$;
iv) Inventory and back-order cost = $c_7[I_t - (c_8 + c_9 d_t)]^2$.

a) Discuss and interpret the assumption made on the behavior of the cost components. Is it reasonable to assume quadratic functions to characterize costs (ii), (iii), and (iv)?
b) Formulate the overall objective function and the constraints of the model.
c) Suggest a procedure to obtain the optimum solution to the model.

23. In an application of the Holt, Modigliani, Muth, and Simon model (see Exercise 22) to a paint factory, the following decision rules were derived for optimal values of P_t and W_t for a twelve-month period starting with the forthcoming month, t.

$$P_t = \left\{ \begin{array}{l} +0.458 d_t \\ +0.233 d_{t+1} \\ +0.111 d_{t+2} \\ +0.046 d_{t+3} \\ +0.014 d_{t+4} \\ -0.001 d_{t+5} \\ -0.007 d_{t+6} \\ -0.008 d_{t+7} \\ -0.008 d_{t+8} \\ -0.007 d_{t+9} \\ -0.005 d_{t+10} \\ -0.004 d_{t+11} \end{array} \right\} + 1.005 W_{t-1} + 153 - 0.464 I_{t-1}$$

* Holt, C. C., F. Modigliani, J. F. Muth, H. A. Simon, *Planning Production, Inventories, and Work-Force*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1960.

$$W_t = 0.742W_{t-1} + 2.00 - 0.010I_{t-1} + \left\{ \begin{array}{l} +0.0101d_t \\ +0.0088d_{t+1} \\ +0.0071d_{t+2} \\ +0.0055d_{t+3} \\ +0.0042d_{t+4} \\ +0.0031d_{t+5} \\ +0.0022d_{t+6} \\ +0.0016d_{t+7} \\ +0.0011d_{t+8} \\ +0.0008d_{t+9} \\ +0.0005d_{t+10} \\ +0.0004d_{t+11} \end{array} \right\}$$

a) Study the structure of the decision rules. How would you apply them? Are you surprised that the decision rules are linear (as opposed to quadratic)?
b) Note the weights that are given to the demand forecast d_t $(t = 1, 2, \ldots, T)$. Comment on the implication of these weights.
c) How would you obtain the resulting optimum inventory levels, I_t?

24. An important problem in production management is the allocation of a given production quantity (determined by an aggregate model or by subjective managerial inputs) among a group of items. For example, let us assume that we have decided to produce $P = 6000$ units of a given product line consisting of three individual items. The allocation of the total quantity among the three items will be decided by the following mathematical model:

$$\text{Minimize } c = \sum_{i=1}^{3} \left(h_i \frac{Q_i}{2} + S_i \frac{d_i}{Q_i} \right)$$

subject to:

$$\sum_{i=1}^{3} Q_i = P,$$

where

Q_i = Production quantity for item i (in units),
h_i = Inventory holding cost for item i (in \$ /month $\times$ unit),
S_i = Setup cost for item i (in \$),
d_i = Demand for item i (in units/month),
P = Total amount to be produced (in units).

a) Interpret the suggested model. What is the meaning of the objective function? What implicit assumption is the model making?
b) The model can be proved to be equivalent to the following unconstrained minimization problem (by means of Lagrange multiplier theory):

$$\text{Minimize } L = \sum_{i=1}^{3} \left(h_i \frac{Q_i}{2} + S_i \frac{d_i}{Q_i} \right) + \lambda \left(\sum_{i=1}^{3} Q_i - P \right).$$

State the optimality conditions for this unconstrained problem (see Section 13.8). Note that the unknowns are Q_i, $i = 1, 2, 3$, and λ. What is the interpretation of λ?
c) Given the following values for the parameters of the problem, establish a procedure to obtain the optimal values of Q_i.

	Items		
Parameter	*1*	*2*	*3*
h_i	1	1	2
S_i	100	50	400
d_i	20,000	40,000	40,000
$Q = 6000$			

[*Hint.* Perform a search on λ; plot the resulting values of Q and λ. Select the optimum value of λ from the graph corresponding to $Q = 6000$.]

d) Apply the SUMT method to the original model. How do you compare the SUMT procedure with the approach followed in part (c)?

25. When applying the Frank-Wolfe algorithm to the optimization problem

$$\text{Maximize } f(x_1, x_2, \ldots, x_n),$$

subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \qquad (i = 1, 2, \ldots, m), \tag{1}$$
$$x_j \geq 0 \qquad (j = 1, 2, \ldots, n),$$

we replace the given problem by a linear approximation at the current solution $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ given by

$$\text{Maximize} \left[f(x^*) + \frac{\partial f}{\partial x_1} x_1 + \frac{\partial f}{\partial x_2} x_2 + \cdots + \frac{\partial f}{\partial x_n} x_n \right],$$

subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \qquad (i = 1, 2, \ldots, m) \tag{2}$$
$$x_j \geq 0 \qquad (j = 1, 2, \ldots, n).$$

The Partial derivatives $(\partial f / \partial x_j)$ are evaluated at the point x^*. We then perform a one-dimensional optimization along the line segment joining x^* with the solution to the linear approximation.

a) Suppose that x^* solves the linear approximation problem so that the solution does not change after solving the linear approximation problem. Show that there are ''Lagrange multipliers" $\lambda_1, \lambda_2, \ldots, \lambda_m$ satisfying the *Kuhn-Tucker Optimality Conditions* for linearly-constrained nonlinear programs:

$$\text{Primal feasibility} \quad \begin{cases} \sum_{j=1}^{n} a_{ij} x_j^* \le b_i & (i = 1, 2, \ldots, m), \\ x_j^* \ge 0 & (j = 1, 2, \ldots, n), \end{cases}$$

$$\text{Dual feasibility} \quad \begin{cases} \frac{\partial f}{\partial x_j} - \sum_{i=1}^{m} \lambda_i a_{ij} \le 0 & (j = 1, 2, \ldots, n) \\ \lambda_i \ge 0 & (i = 1, 2, \ldots, m), \end{cases}$$

$$\text{Complementary slackness} \quad \begin{cases} \left[\frac{\partial f}{\partial x_j} - \sum_{i=1}^{m} \lambda_i a_{ij} \right] x_j^* = 0 & (j = 1, 2, \ldots, m). \\ \lambda_i \left[\sum_{j=1}^{n} a_{ij} x_j^* - b_i \right] = 0 & (i = 1, 2, \ldots, m). \end{cases}$$

b) What is the form of these Kuhn-Tucker conditions when (1) is a linear program or a quadratic program?

c) Suppose that $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ solves the original optimization problem (1). Show that x^* also solves the linear approximation problem (2) and therefore satisfies the Kuhn-Tucker conditions. [*Hint.* Recall that

$$\lim_{\theta \to 0} \frac{f(x^* + \theta x_j) - f(x^*)}{\theta} = \frac{\partial f}{\partial x_1} x_1 + \frac{\partial f}{\partial x_2} x_2 + \cdots + \frac{\partial f}{\partial x_n} x_n. \Big]$$

d) Suppose that $f(x_1, x_2, \ldots, x_n)$ is a convex function. Show that if $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$ solves the Kuhn-Tucker conditions, then x^* is a global minimum to the original non-linear program (1).

26. When discussing sensitivity analysis of linear programs in Chapter ??, we indicated that the optimal objective value of a linear program is a concave function of the righthand-side values. More generally, consider the optimization problem

$$v(b_1, b_2, \ldots, b_m) = \text{Maximize } f(x),$$

subject to:

$$g_i(x) \le b_i \quad (i = 1, 2, \ldots, m)$$

where $x = (x_1, x_2, \ldots, x_n)$ are the problem variables; for linear programs

$$f(x) = \sum_{j=1}^{n} c_j x_j \quad \text{and} \quad g_i(x) = \sum_{j=1}^{n} a_{ij} x_j \quad \text{for } i = 1, 2, \ldots, m.$$

a) Show that if each $g_i(x)$ is a convex function, then the values of $b_1, b_2, \ldots, b_m$ for which the problem has a feasible solution form a convex set C.

b) Show that if, in addition, $f(x)$ is a concave function, then the optimal objective value $v(b_1, b_2, \ldots, b_m)$ is a concave function of the righthand-side values $b_1, b_2, \ldots, b_m$ on the set C.

27. In many applications, the ratio of two linear functions is to be maximized subject to linear constraints. The linear fractional programming problem is:

$$\text{Maximize } \frac{\sum_{j=1}^{n} c_j x_j + c_0}{\sum_{j=1}^{n} d_j x_j + d_0},$$

subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i \quad (i = 1, 2, \ldots, m), \tag{1}$$

$$x_j \ge 0 \quad (j = 1, 2, \ldots, n).$$

A related linear program is

$$\text{Maximize } \sum_{j=1}^{n} c_j y_j + c_0 y_0,$$

subject to:

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} y_j - b_i y_0 &\le 0 \quad (i = 1, 2, \ldots, m), \\ \sum_{j=1}^{n} d_j y_j + d_0 y_0 &= 1, \\ y_j &\ge 0 \qquad (j = 0, 1, 2, \ldots, n). \end{aligned} \tag{2}$$

a) Assuming that the optimal solution to the linear fractional program occurs in a region where the denominator of the objective function is strictly positive, show that:
   i) If $y_j^* (j = 0, 1, 2, \ldots, n)$ is a finite optimal solution to (2) with $y_0^* > 0$, then $x_j^* = y_j^*/y_0^*$ is a finite optimal solution to (1).
   ii) If $\lambda y_j^* (j = 0, 1, 2, \ldots, n)$ is an unbounded solution to (2) as $x \to \infty$ and $y_0^* > 0$, then $\lambda x_j^* = \lambda y_j^*/y_0^* (j = 1, 2, \ldots, n)$ is an unbounded solution of (1) as $\lambda \to \infty$.

b) Assuming that it is not known whether the optimal solution to the linear fractional program occurs in a region where the denominator is positive or in a region where it is negative, generalize the approach of (a) to solve this problem.

28. A *primal* optimization problem is:

$$\text{Maximize } f(x),$$

subject to:

$$\begin{aligned} g_i(x) &\le 0 \quad (i = 1, 2, \ldots, m), \\ x &\in X \end{aligned}$$

where x is understood to mean $(x_1, x_2, \ldots, x_n)$ and the set X usually means $x_j \ge 0 \, (j = 1, 2, \ldots, n)$ but allows other possibilities as well. The related Lagrangian form of the problem is:

$$L(y) = \underset{x \in X}{\text{Maximize}} \left[f(x) - \sum_{i=1}^{m} y_i g_i(x) \right].$$

The *dual* problem is defined to be:

$$\text{Minimize } L(y),$$

subject to:

$$y_i \ge 0 \quad (i = 1, 2, \ldots, m).$$

Without making any assumptions on the functions $f(x)$ and $g_i(x)$, show the following:

a) (*Weak duality*) If $\overline{x}_j \ (j = 1, 2, \ldots, n)$ is feasible to the primal and $\overline{y}_i \ (i = 1, 2, \ldots, m)$ is feasible to the dual, then $f(\overline{x}) \le L(\overline{y})$.

b) (*Unboundedness*) If the primal (dual) is unbounded, then the dual (primal) is infeasible.
c) (*Optimality*) If $\hat{x}_j$ $(j = 1, 2, \ldots, n)$ is feasible to the primal and $\hat{y}_i$ $(i = 1, 2, \ldots, m)$ is feasible to the dual, and, further, if $f(\hat{x}) = L(\hat{y})$, then $\hat{x}_j$ $(j = 1, 2, \ldots, n)$ solves the primal and $\hat{y}_i$ $(i = 1, 2, \ldots, m)$ solves the dual.

29. Suppose $\hat{x} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n)$ and $\hat{y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m)$ satisfy the following saddlepoint condition:

$$f(x) - \sum_{i=1}^{m} \hat{y}_i g_i(x) \underset{\uparrow}{\leq} f(\hat{x}) - \sum_{i=1}^{m} \hat{y}_i g_i(\hat{x}) \underset{\uparrow}{\leq} f(\hat{x}) - \sum_{i=1}^{m} y_i g_i(\hat{x}),$$

All $(x_1, x_2, \ldots, x_n) \in X$ All $y_i \geq 0$ $(i = 1, 2, \ldots, m)$

a) Show that $\hat{x}$ solves the nonlinear program

$$\text{Maximize } f(x),$$

subject to:

$$g_i(x) \leq 0 \quad (i = 1, 2, \ldots, m),$$
$$x \in X$$

where x refers to $(x_1, x_2, \ldots, x_n)$. [*Hint.* (1) Show that complementary slackness holds; *i.e.*, $\sum_{i=1}^{m} \hat{y}_i g_i(\hat{x}) = 0$, using the righthand inequality with $y_i = 0$ $(i = 1, 2, \ldots, m)$ to show ''$\geq$'', and the signs of y_i and $g_i(x)$ to ''$\leq$''. (2) Use the lefthand inequality and the sign of $g_i(x)$ to complete the proof.]

b) Show that the saddlepoint condition implies a strong duality of the form

Maximize $f(x)$	=	Minimize $L(y)$
subject to:		subject to:
$g_i(x) \leq 0$		$y_i \geq 0$
$x \in X$		

where $L(y)$ is defined in Exercise 28.

30. In linear programming, the duality property states that, whenever the primal (dual) has a finite optimal solution, then so does the dual (primal), and the values of their objective functions are equal. In nonlinear programming, this is not necessarily the case.

a) (*Duality gap*) Consider the nonlinear program:

$$\text{Maximize } \{\text{Min}|(x_1 x_2)^{1/2}; 1|\},$$

subject to:

$$x_1 = 0,$$
$$x_1 \geq 0, \qquad x_2 \geq 0.$$

The optimal objective-function value for the primal problem is clearly 0. The Lagrangian problem is:

$$L(y) = \text{ Maximize } \{\text{Min}|(x_1 x_2)^{1/2}; 1| + y x_1\},$$

subject to:

$$x_1 \geq 0, \qquad x_2 \geq 0.$$

Show that the optimal objective value of the dual problem

$$\underset{y}{\text{Minimize}} \quad L(y)$$

is 1. (Note that y is unrestricted in the dual, since it is associated with an equality constraint in the primal.)

b) (*No finite shadow prices*) Consider the nonlinear program:

$$\text{Minimize } x_1,$$

subject to:

$$x_1^2 \leq 0,$$
$$x_1 \text{ unrestricted.}$$

The optimal solution to this problem is clearly $x_1 = 0$. The Lagrangian problem is:

$$L(y) = \underset{x_1}{\text{Maximize}} \{x_1 - yx_1^2\},$$

where x_1 is unrestricted. Show that the optimal solution to the dual does not exist but that $L(y) \to 0$ as $y \to \infty$.

ACKNOWLEDGMENTS

A number of the exercises in this chapter are based on or inspired by articles in the literature.
Exercise 21: L. S. White, ''Mathematical Programming Models for Determining Freshman Scholarship Offers,'' M.I.T. Sloan School Working Paper No. 379–9g.
Exercises 22 and 23:C. C. Holt, F. Modigliani, J. F. Muth, and H. A. Simon, *Planning Production, Inventories, and Work-Force*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1960.
Exercise 24: P. R. Winters, ''Constrained Inventory Rules for Production Smoothing,'' *Management Science*, 8, No. 4, July 1962.
Exercise 27: A. Charnes and W. W. Cooper, ''Programming with Linear Fractional Functionals,'' *Naval Research Logistics Quarterly*, 9, 1962; and S. P. Bradley and S. C. Frey, Jr. ''Fractional Programming with Homogeneous Functions,'' *Operations Research*, 22, No.2, March-April 1974.
Exercise 30: M. Slater, ''Lagrange Multipliers Revisited: A Contribution to Nonlinear Programming,'' Cowles Commission Paper, Mathematics 403, November 1950; and R. M. van Slyke and R. J. Wets, ''A Duality Theory for Abstract Mathematical Programs with Applications to Optimal Control Theory,'' *Journal of Mathematical Analysis and Applications*,22, No. 3, June 1968.